Sistem Dialog Multi Entitas Domain Pemesanan Menggunakan Pendekatan Berbasis Aturan

RINGKASAN DISERTASI

Athia Saelan
NIM: 33216308
(Program Studi Doktor Teknik Elektro dan Informatika)

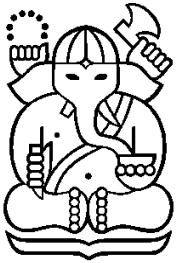
Institut Teknologi Bandung
Desember 2023

Sistem Dialog Multi Entitas Domain Pemesanan Menggunakan Pendekatan Berbasis Aturan

Disertasi ini dipertahankan pada Sidang Tertutup Sekolah Pascasarjana sebagai salah satu syarat untuk memperoleh gelar Doktor Institut Teknologi Bandung

Desember 2023

Athia Saelan
NIM: 33216308
(Program Studi Doktor Teknik Elektro dan Informatika)



Promotor : Dr. Ir. Rinaldi, M.T.
Ko-promotor 1 : Dr. Eng. Ayu Purwarianti, S.T., M.T.
Ko-promotor 2 : Dr. Ir. Rila Mandala, M.Eng.

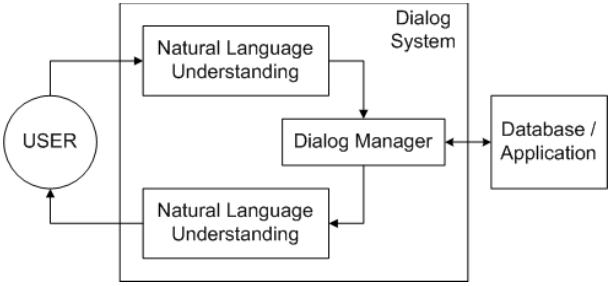
Institut Teknologi Bandung
Desember 2023

Sistem Dialog Multi Entitas Domain Pemesanan Menggunakan Pendekatan Berbasis Aturan

Athia Saelan
NIM: 33216308

1. Latar Belakang

Sistem dialog atau *chatbot* merupakan sistem komputer yang dapat melakukan percakapan dengan manusia. Secara umum, suatu sistem dialog terdiri dari tiga bagian utama, yaitu bagian pemahaman bahasa (*Natural Language Understanding* – NLU), bagian manajemen dialog (*Dialog Manager* – DM), serta bagian pembangkitan bahasa (*Natural Language Generator* – NLG). Arsitektur umum dari sistem dialog dapat dilihat pada Gambar 1 berikut.



Gambar 1. Arsitektur Umum Sistem Dialog

Berdasarkan tujuannya, sistem dialog terdiri dari dua jenis, yaitu sistem dialog *task-oriented* (yang lebih sering disebut sebagai sistem dialog saja) dan sistem dialog *non-task-oriented* (yang biasa disebut *chatbot*). Sistem dialog *task-oriented* adalah sistem dialog yang memiliki tujuan atau tugas tertentu yang harus diselesaikan, misalnya pembelian tiket atau pemesanan makanan. Sedangkan pada sistem dialog *non-task-oriented*, tujuannya hanya untuk bercakap-cakap saja, tanpa tujuan akhir yang spesifik. Penelitian ini fokus pada sistem dialog *task-oriented*.

Sistem dialog *task-oriented* dibangun untuk menyelesaikan suatu tugas. Karena itu, sistem harus bisa melacak sampai mana dialog berjalan, serta mengingat hal-hal yang telah dibahas selama percakapan. Hal ini biasanya disimpan dalam *dialog state*. Saat ini, kebanyakan sistem dialog *task-oriented* menggunakan model *slot-filling*, yaitu mengisi daftar *slot* yang telah didefinisikan selama percakapan berlangsung (Lee & Stent, 2016; Bocklisch dkk, 2017; Balaraman, 2021).

Kekurangan dari model ini adalah sistem hanya fokus pada satu entitas yang sedang dibahas. Sistem mungkin dapat menyimpan banyak entitas, tapi setiap saatnya tetap hanya fokus pada satu entitas saja. Pada kenyataannya, ada banyak kasus percakapan yang melibatkan banyak entitas sekaligus. Misalnya pada dialog pemesanan makanan, seseorang dapat memesan beberapa jenis makanan sekaligus dengan detail yang berbeda-beda. Contoh lainnya, pada perencanaan perjalanan, seseorang bisa saja menyebutkan beberapa tempat yang berbeda sekaligus. Selain itu, pada dialog dengan multi entitas, *user* juga harus dapat mengubah ataupun membatalkan entitas-entitas yang sudah disebutkan sebelumnya sewaktu-waktu. Sebagai contoh, misalnya *user* telah memesan tiga item, dan ingin membatalkan salah satunya, sistem harus bisa menentukan item mana yang akan dibatalkan.

Beberapa penelitian telah dilakukan untuk mengatasi kekurangan tersebut. Penelitian yang membahas mengenai sistem dialog multi entitas antara lain adalah CEDM (Ultes dkk, 2019; Papangelis & Ultes, 2020). Penelitian tersebut merancang struktur *dialog state* untuk menyimpan banyak entitas, serta metode *dialog state update* yang sesuai. Namun penelitian tersebut hanya berfokus pada dialog manager, dan tidak membahas mengenai bagaimana ekstraksi teks yang mengandung banyak entitas dan atribut, serta menghubungkannya dengan *dialog state*.

Sementara itu, beberapa penelitian meneliti metode untuk ekstraksi teks input yang mengandung banyak entitas (Gangadharaiah & Narayanaswamy, 2020; Gubbala & Zhang, 2021). Namun, pada penelitian tersebut, semua nama entitas dan atributnya masih harus disebutkan satu per satu pada teks. Sebagai contoh, "saya pesan pizza keju besar dan pizza pepperoni medium". Penelitian tidak membahas mengenai split entitas, misalnya "saya pesan 2 pizza, 1 keju ukuran besar, 1 pepperoni ukuran medium", entitas yang terpencar, misalnya "saya pesan pizza keju dan pizza pepperoni, dua-duanya ukuran besar", ataupun perujukan ke entitas sebelumnya, misalnya "yang keju ukuran besar ya" yang merujuk ke entitas pizza yang telah dibahas sebelumnya.

Sedangkan pada Frames (El Asri dkk., 2017; Schulz dkk., 2017), sistem dialog yang dibuat memiliki banyak frame dalam *dialog state*-nya, dengan maksud untuk menyimpan semua entitas yang telah dibahas. Namun pada penelitian tersebut, frame lebih dimaksudkan sebagai "*history*" tentang entitas apa saja yang pernah dibahas. Pada setiap *turn*-nya, sistem tersebut tetap fokus pada satu entitas yang sedang dibahas saja, dan tidak menangani banyak entitas sekaligus. Hanya saja, sistem tetap bisa merujuk ke entitas lain yang pernah dibahas sebagai perbandingan, serta bisa kembali lagi ke entitas yang lama bila *user* menginginkannya.

Sejauh ini, penelitian sistem dialog yang telah dilakukan belum ada yang membahas mengenai penanganan dialog multi entitas mulai dari awal sampai akhir. Untuk itu,

penelitian ini membahas mengenai pembangunan sistem dialog multi entitas. Sistem dialog multi entitas yang dibangun harus memiliki kemampuan berikut.

1. Menerima banyak entitas dan atribut entitas dalam satu input, termasuk entitas yang implisit (tidak dituliskan nama entitasnya apa).
2. Menentukan entitas yang disebut oleh *user* merupakan entitas baru atau merujuk ke entitas yang sudah ada di *dialog state*, serta menentukan rujukannya.
3. Melakukan aksi yang berbeda-beda untuk setiap entitas yang disebut.
4. Mengeluarkan output sesuai dengan input terakhir dan *dialog state*.

Ada tiga metode untuk membangun sistem dialog dan aplikasi AI secara umum, yaitu metode berbasis aturan, pembelajaran tradisional, maupun *deep learning*. Untuk pembangunan sistem dialog dengan pendekatan pembelajaran mesin, baik tradisional maupun *deep learning*, membutuhkan data yang cukup banyak. Model pembelajaran tradisional membutuhkan data yang dilabeli untuk setiap komponennnya. Sedangkan model berbasis *deep learning* membutuhkan data yang banyak dalam orde ratusan juta. Selain itu, pelatihan model *deep learning* juga membutuhkan perangkat dengan kemampuan komputasi yang tinggi dan waktu yang lama.

Untuk ketersediaan data, ada beberapa *dataset* dialog yang tersedia dan dapat digunakan untuk penelitian. Data yang banyak digunakan antara lain data DSTC (Williams dkk., 2013; Henderson dkk., 2014a; Henderson dkk., 2014b), dan MultiWOZ (Budzianowski dkk., 2018; Ye dkk., 2022). Namun data-data tersebut menggunakan model *slot-filling* sehingga tidak cocok untuk sistem dialog multi entitas. Data lain yang mungkin lebih cocok antara lain Frames (El Asri dkk., 2017) dan GraphWOZ (Walker dkk., 2022). Namun seperti yang dibahas sebelumnya, model Frames memang menyimpan banyak entitas pada *dialog state*, tetapi hanya fokus pada satu entitas untuk setiap saatnya sehingga masih kurang cocok. Sedangkan GraphWOZ juga menyimpan dan menerima banyak entitas. Namun model entitasnya hanya berupa nama entitas, bukan entitas dengan atribut berupa *slot-value* seperti yang digunakan pada penelitian ini. Model ini lebih fokus pada hubungan antar entitas.

Karena itu, sistem dialog multi entitas yang dibangun pada penelitian ini menggunakan metode berbasis aturan dengan pertimbangan berikut.

1. Penelitian ini lebih fokus pada model dan alur dialog daripada metode untuk masing-masing komponennya.
2. Belum tersedianya *dataset* dialog yang tepat untuk melatih model sistem dialog multi entitas, terutama dalam Bahasa Indonesia.
3. Keterbatasan kemampuan komputasi dari perangkat yang digunakan.
4. Metode berbasis aturan terbukti cukup efektif untuk domain yang terbatas (Popovski dkk., 2019).

2. Tujuan dan Sasaran Penelitian

Sejauh ini, belum ada penelitian sistem dialog multi entitas secara terintegrasi yang membahas masalah dialog multi entitas dari awal (input teks) hingga akhir (output teks). Selain itu, untuk membangun suatu sistem dialog juga dibutuhkan *dataset* dialog baik untuk pelatihan model maupun pengujian sistem. Karena itu, penelitian ini bertujuan untuk:

1. membangun alur sistem dialog yang dapat menangani dialog multi entitas,
2. mengumpulkan dataset dialog multi entitas.

Sedangkan sasarannya terutama adalah untuk dialog pemesanan makanan, karena di pada dialog tersebut biasanya terdapat beberapa entitas yang dibahas sekaligus, serta sering dirujuk kembali untuk pengubahan detail atau pembatalan.

3. Metode Penelitian

Penelitian ini terdiri dari beberapa tahap, yaitu pengumpulan data percakapan, analisis masalah-masalah yang terjadi pada dialog multi entitas, perancangan solusi, implementasi sistem dialog multi entitas, serta evaluasi.

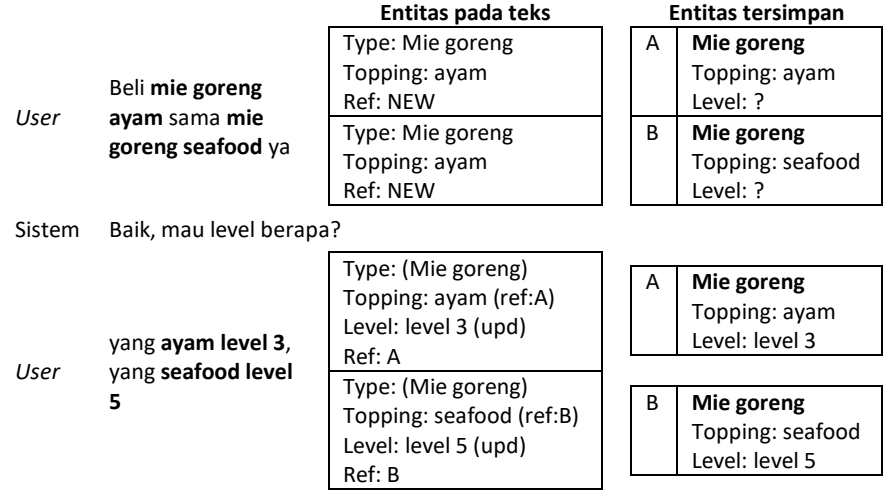
1. Pengumpulan data dilakukan dengan metode Wizard of Oz (WoZ).
2. Analisis masalah dialog multi entitas dilakukan dengan mencoba berbagai kemungkinan penyebutan entitas, perujukan entitas, serta perbedaan proses yang harus dilakukan untuk setiap entitas.
3. Perancangan solusi dan implementasi sistem dilakukan dengan beberapa tahap:
   a. memecah masalah pemrosesan dialog multi entitas menjadi masalah-masalah kecil yang dapat diselesaikan secara terpisah,
   b. membandingkan solusi yang ada untuk setiap masalah pada penelitian terkait,
   c. memilih solusi terbaik yang ada berdasarkan masalah yang dihadapi dan ketersediaan *resource*,
   d. mengimplementasikan algoritma atau aturan untuk setiap masalah yang dihadapi.
4. Evaluasi sistem dialog dilakukan dengan beberapa cara:
   a. menghitung akurasi hasil pemrosesan data percakapan dengan sistem dialog yang dibangun untuk beberapa aspek, dan membandingkannya dengan beberapa model sistem dialog yang lain,
   b. evaluasi model yang digunakan secara kualitatif.

4. Entitas dalam Sistem Dialog Multi Entitas

Sebelum pembahasan lebih lanjut, definisi entitas dalam penelitian ini perlu dijelaskan terlebih dahulu. Suatu entitas adalah suatu objek yang dibahas pada dialog. Entitas didefinisikan dengan *entity-type* dan sekumpulan *slot-value*. Sebagai contoh, *entity-type* misalnya nasi goreng dan roti bakar, sedangkan *slot-value* misalnya besar dan kecil (untuk *slot* ukuran) atau telur dan keju (untuk *slot topping*). Suatu entitas pasti memiliki *entity-type*, tetapi bisa jadi tidak atau belum memiliki *slot-value*. Sebagai contoh, saat *user* memesan nasi goreng, bisa jadi *user* belum menentukan

topping untuk nasi goreng tersebut serta level pedasnya. Namun tidak mungkin *user* memesan topping untuk *entity-type* yang belum ada.

Entitas ada yang baru disebutkan oleh *user*, ada juga yang telah tersimpan di dalam *dialog state* (*memory* dari sistem dialog). Entitas yang disebutkan oleh *user* bisa jadi merupakan entitas baru ataupun merujuk ke entitas yang dibahas sebelumnya (yang tersimpan di *dialog state*). Contohnya dapat dilihat pada Gambar 2.



Gambar 2. Contoh pemrosesan dialog multi entitas

5. Kasus-Kasus pada Dialog Multi Entitas

Ada beberapa hal menarik terkait masalah multi entitas pada dialog. Misalnya untuk cara penyebutan entitas dalam teks, hal-hal berikut yang mungkin tidak ditemukan dalam dialog dengan satu entitas.

1. Penyebutan *slot-value* untuk beberapa entitas sekaligus, misalnya "nasi goreng dan mie gorengnya level 3 ya".
2. Penyebutan lebih dari satu *value* untuk satu *slot*, misalnya "Mau beli mie goreng ayam sama yang seafood ya"
3. Penyebutan *slot-value* yang terpisah-pisah, misalnya "mau nasi goreng seafood sama nasi goreng ayam ya, yang seafood level 3, yang ayam level 5"

Selain penyebutan entitas, hal lain yang menarik adalah perujukan entitas. Entitas yang dirujuk pun bisa jadi berada dalam teks yang sama, atau pada teks sebelumnya. *User* dapat merujuk suatu entitas dengan berbagai cara, misalnya sebagai berikut.

1. Menyebut *entity-type*, contoh: "nasi gorengnya", "es krimnya".
2. Menyebut *entity-type* dan *slot-value*, contoh: "nasi goreng yang pakai telur", "es krim coklatnya".

3. Menyebut *slot-value*, contoh: “yang telur”, “yang pakai ayam”, “yang rasa coklat”.
4. Tidak menyebut keduanya, atau implisit (biasanya merupakan jawaban dari pertanyaan sistem).

Selain itu, bisa jadi juga ada satu penyebutan entitas yang merujuk ke banyak entitas pada *dialog state*, seperti contoh pada Gambar 3. Sebaliknya, bisa juga ada beberapa entitas yang disebut yang merujuk ke satu entitas pada *dialog state*, seperti contoh pada Gambar 4.

User	Mau beli **mie goreng ayam** sama **yang seafood** ya
Sistem	Baik, mau level berapa?
User	**level 3**

Gambar 3. Contoh satu penyebutan entitas yang merujuk ke banyak entitas

User	Mau beli nasi goreng ayam 2 ya
Sistem	Baik, mau level?
User	1 **level 3**, 1 lagi **level 5**

Gambar 4. Contoh banyak penyebutan entitas yang merujuk ke satu entitas

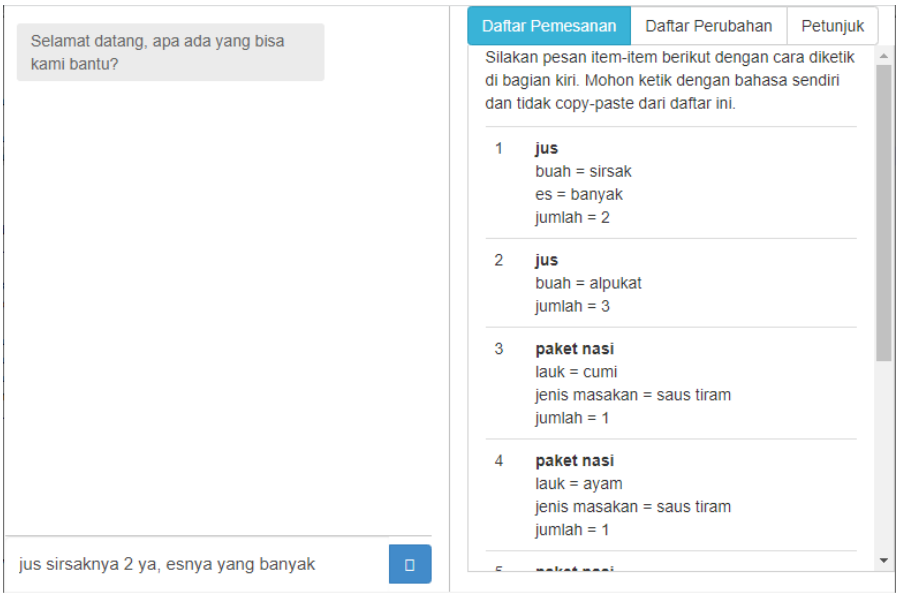
Sedangkan untuk pemrosesan dialog, setiap entitas yang disebut dalam dialog memerlukan pemrosesan yang berbeda. Contohnya pada kalimat “jusnya dibatalkan aja ya, jadinya nasi goreng aja”. Pada kalimat tersebut, pemrosesan untuk entitas “jus” adalah pembatalan. Sedangkan entitas “nasi goreng” memiliki dua kemungkinan pemrosesan tergantung konteksnya. Jika *user* sudah pernah memesan “nasi goreng” sebelumnya, berarti penyebutannya bertujuan untuk konfirmasi bahwa entitas tersebut jadi dibeli. Sedangkan jika entitas “nasi goreng” belum pernah disebut, maka entitas baru “nasi goreng” harus ditambahkan.

6. Pengumpulan Data Percakapan

Pengumpulan data dilakukan dengan metode Wizard of Oz (WoZ). Pada metode ini, seseorang akan berperan sebagai sistem (disebut *wizard*), dan seorang lagi berperan sebagai *user* (disebut partisipan). Data yang dikumpulkan berupa dialog pemesanan makanan. Pengumpulan data ini dilakukan dalam domain pemesanan makanan. Pengumpulan data dilakukan pada halaman *website* yang contohnya dapat dilihat pada Gambar 5.

Proses ini diikuti oleh 20 orang partisipan. Setiap partisipan diminta untuk memesan beberapa *item* yang telah ditentukan oleh sistem. *Item* yang dipilih oleh sistem ini telah diatur agar banyak *item* dengan *entity-type* yang sama dengan detail yang berbeda-beda. Dengan begini, diharapkan partisipan dapat melakukan variasi yang lebih banyak dalam cara penyebutan entitasnya. Selain itu, partisipan juga diminta untuk mengubah beberapa *item* yang telah dipesan sesuai daftar perubahan yang

telah ditentukan oleh sistem. Perubahan ini juga dipilih secara acak dari beberapa *item* pemesanan. Partisipan juga diminta untuk menulis dengan bahasa sendiri, menggunakan bahasa percakapan, dan sebisa mungkin melakukan variasi.

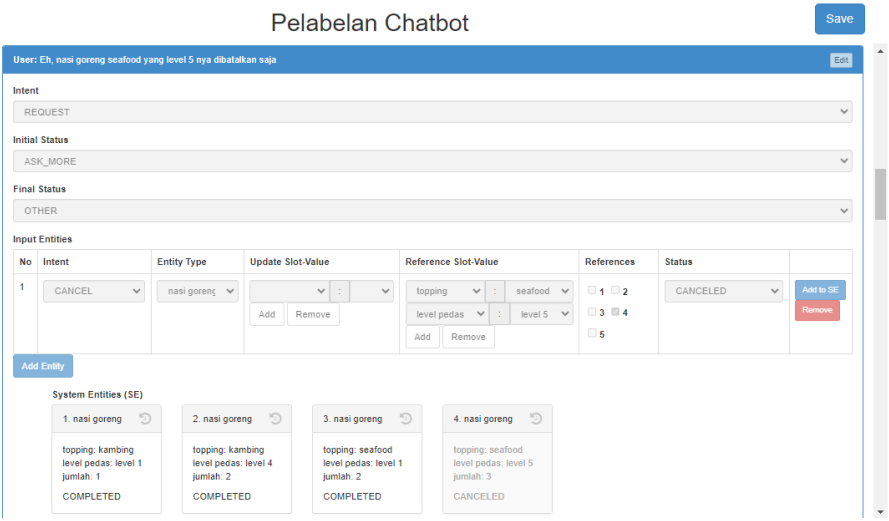


Gambar 5. Contoh tampilan halaman pengumpulan data

Wizard	Selamat datang, apa ada yang bisa kami bantu?
User	Nasi goreng kambing 3, yang 1 level 1, yang 2 level 4, sama nasi goreng seafood 5, yang 2 level 1, 3 lagi level 5
Wizard	Baik, apakah ada lagi?
User	Eh, nasi goreng seafood yang level 5 nya dibatalkan saja
Wizard	Tiga-tiganya dibatalkan?
User	Ya, sama teh dingin tawar ya 3
Wizard	Baik, apakah ada lagi?
User	Eh tehnya tidak jadi, jadinya nasi goreng saja ya
Wizard	Baik, apakah ada lagi?
User	Tidak
Wizard	Baik, jadi pesanan anda: 1 nasi goreng kambing level 1, 2 nasi goreng kambing level 4, dan 2 nasi goreng seafood level 1, apakah sudah betul?
User	Ya, betul
Wizard	Baik, terima kasih

Gambar 6. Contoh data percakapan yang didapatkan

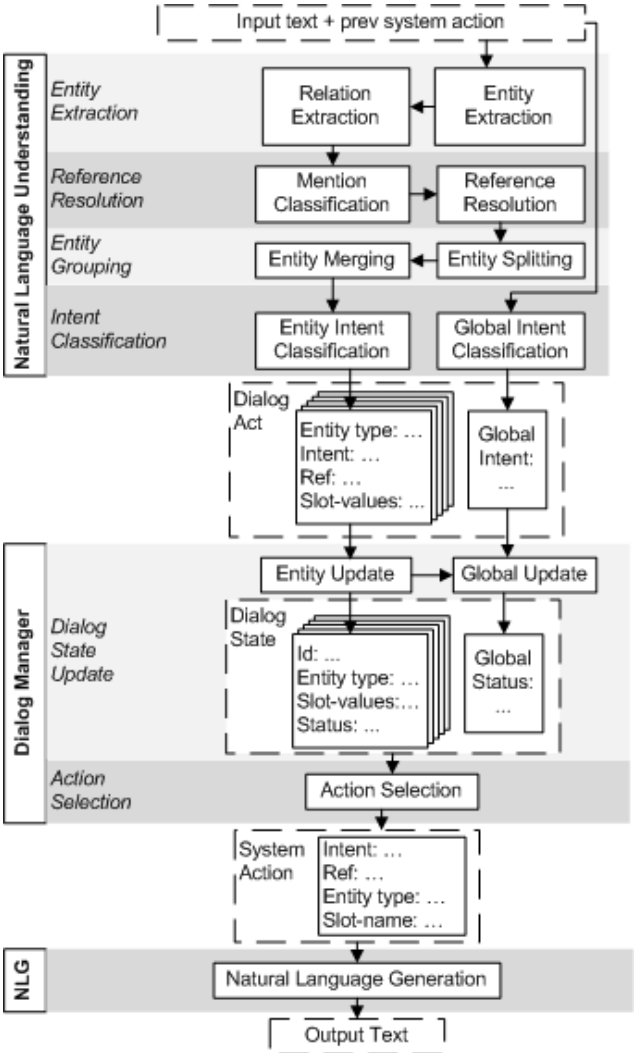
Data yang dikumpulkan terdiri dari 50 percakapan dan 220 teks input. Salah satu contoh percakapan yang didapatkan dapat dilihat pada Gambar 6. Data digunakan untuk proses evaluasi yang dibahas pada bagian 8. Namun sebelum itu, data ini perlu dilabeli terlebih dahulu. Proses pelabelan dilakukan pada halaman pelabelan yang contohnya dapat dilihat pada Gambar 7. Proses ini berfungsi untuk menentukan daftar entitas pada setiap teks *input*. Hal ini diperlukan untuk proses evaluasi sistem, yaitu hasil pemrosesan sistem dianggap benar apabila sama dengan hasil pelabelan. Selain itu, proses pelabelan juga dilakukan secara berurutan, yaitu dari teks *input* pertama dalam proses percakapan hingga teks terakhir. Dengan begini, entitas-entitas yang terdapat pada teks sebelumnya dapat disimpan sebagai konteks untuk teks selanjutnya, misalnya untuk perujukan entitas. Sebagai contoh, jika teks sebelumnya membahas entitas "nasi goreng", kemudian teks selanjutnya disebutkan "level 3", maka kemungkinan "level 3" yang dimaksud adalah untuk entitas "nasi goreng" dan bukan yang lainnya.



Gambar 7. Contoh halaman pelabelan percakapan

7. Rancangan Sistem Dialog Multi Entitas

Sistem dialog yang dibangun terdari dari tiga modul utama mengikuti sistem dialog pada umumnya, yaitu *Natural Language Understanding* (NLU), *Dialog Manager* (DM), dan *Natural Language Generation* (NLG). Rancangan dari sistem dapat dilihat pada Gambar 8. Selain itu ada juga dua struktur penting yang perlu dibahas pada rancangan ini, yaitu *dialog act* dan *dialog state*.



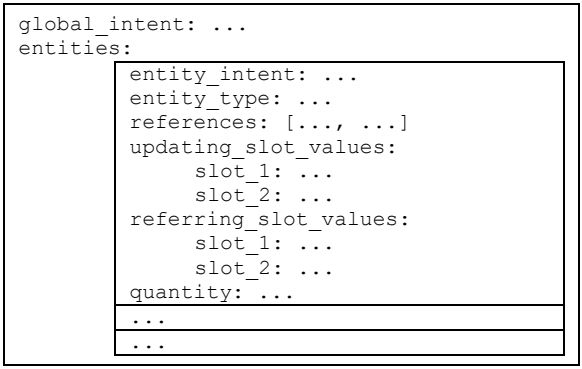
Gambar 8. Rancangan Sistem Dialog Multi Entitas

7.1 Rancangan *Dialog Act*

Dialog act merupakan representasi dari teks input pada sistem dialog. Secara umum *dialog act* terdiri dari dua bagian. Bagian pertama adalah *intent* yang menunjukkan tujuan diucapkannya suatu kalimat pada dialog. Kemudian bagian lainnya berisi informasi-informasi penting terkait tujuan tersebut, yang biasanya direpresentasikan dalam bentuk pasangan *slot* dan *value*.

Pada suatu dialog multi entitas, setiap teks input dapat berisi beberapa entitas sekaligus. Setiap entitas sendiri bisa jadi merupakan entitas baru atau *update* untuk entitas yang sudah ada di *dialog state*. Jika merupakan *update*, sistem harus menentukan entitas mana yang di-*update*. Jadi *dialog act* dalam suatu sistem dialog multi entitas harus dapat merepresentasikan banyak entitas, dengan setiap entitas memiliki *entity-type* dan *slot-values*, dan setiap entitas bisa jadi merupakan entitas baru ataupun merujuk.

Selain itu, setiap entitas juga bisa jadi memiliki tujuan penyebutan yang berbeda . Karena itu, *dialog act* untuk sistem dialog multi entitas juga perlu memiliki *intent* yang terpisah untuk setiap entitas. Namun tidak semua input teks memiliki entitas, sehingga harus ada juga *intent* global. Karena itu, ada dua jenis *intent* yang harus diterapkan, yaitu *intent* global dan *intent* entitas. Bentuk dari *dialog act* yang dirancang dapat dilihat pada Gambar 9.



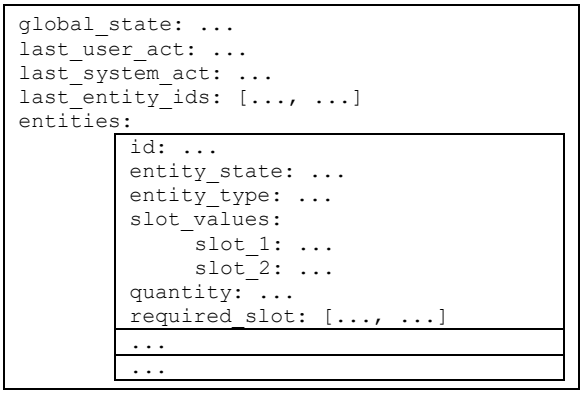
Gambar 9. Rancangan *dialog act*

7.2 Rancangan *Dialog State*

Dialog state berfungsi sebagai penyimpanan sementara mengenai hal-hal yang dibahas selama dialog berlangsung. Perancangan *dialog state* merupakan hal yang penting dalam perancangan sistem dialog multi entitas, karena pada *dialog state* inilah entitas-entitas tersebut akan disimpan. Namun, selain menampung banyak entitas, *dialog state* pada sistem dialog multi entitas harus:

1. Menyatakan status dari setiap entitas (misalnya spesifikasi lengkap, belum lengkap, dibatalkan)
2. Menyatakan status dialog secara keseluruhan (misalnya baru mulai, sedang membahas entitas, konfirmasi)
3. Menyimpan daftar entitas yang terakhir dibahas (untuk mengetahui *entity-type* dari *slot-value*, dan untuk perujukan)

Karena itu, model *dialog state* ysng dirancang adalah seperti pada Gambar 10. Salah satu perbedaan sistem ini dengan sistem *slot-filling* biasa adalah setiap entitas memiliki *dialog state* sendiri. Jadi selain informasi *entity-type* dan *slot-value*, setiap entitas menyimpan status dialog atau aksi terakhir sistem yang terkait dengan entitas tersebut, serta informasi lain yang dinilai perlu disimpan. Selain itu, ada juga *dialog state* global untuk menyimpan status dialog secara global yang tidak terkait dengan entitas.



Gambar 10. Rancangan *Dialog State*

Hal yang penting dari rancangan tersebut adalah adanya *dialog state* setiap entitas selain *dialog state* global. Pada *dialog state* global, disimpan status global, *dialog act* terakhir baik dari *user* maupun sistem, serta dafar entitas yang terakhir dibahas. *Dialog act* serta daftar entitas terakhir ini penting disimpan sebagai konteks untuk *input user* berikutnya. Kemudian pada *dialog state* entitas, pertama yang disimpan adalah id sebagai kode unik untuk mengidentifikasi setiap entitas. Selain itu, informasi-informasi mengenai entitas, seperti status entitas, tipe entitas, daftar *slot*, dan jumlah juga disimpan. Terakhir, disimpan juga daftar *slot* yang wajib, yang jika ada yang belum terisi akan ditanyakan oleh sistem.

7.3 Modul *Natural Language Understanding*

Modul NLU berfungsi untuk menerjemahkan teks dari *user* menjadi semantiknya, yaitu *dialog act*. Modul ini terdiri dari delapan komponen seperti yang dapat dilihat pada Gambar 8. Bagian pemroesan entitas dimulai dari ekstraksi entitas dan ekstraksi relasi. Setiap entitas yang diekstraksi kemudian akan masuk ke proses klasifikasi mention. Selanjutnya, akan dilakukan pencarian rujukan untuk setiap entitas yang merupakan mention. Setelah itu, dilakukan proses pengelompokan entitas yang terdiri dari *entity merging* dan *entity splitting*. Terakhir, pada *entity intent classification*, sistem memprediksi tujuan pengguna menyebutkan entitas tersebut. Sedangkan untuk pemrosesan *global*, sistem hanya menentukan tujuan

pengguna secara umum di luar entitas pada *global intent classification*. Metode yang digunakan untuk setiap komponen dapat diubah sesuai dengan ketersediaan *resource* dan kebutuhan. Namun, metode yang diimplementasikan pada penelitian ini terangkum dalam Tabel 1.

Tabel 1. Ringkasan Proses NLU

Komponen	Keterangan	Metode
Entity Extraction	Ekstraksi *entity-type*, *slot-value*, dan angka	Pencocokan dengan daftar entitas
Relation Extraction	Ekstraksi relasi antara *slot-value* dengan *entity-type*-nya, antara angka (jumlah) dengan *entity-type* atau *slot-value* yang sesuai	Aturan berdasarkan jarak dan daftar entitas
Mention Classification	Klasifikasi apakah *entity-type* atau *slot-value* merupakan mention, yaitu merujuk ke entitas yang sudah disebut.	Aturan berdasarkan kata yang disebut sebelum/setelah suatu entitas
Reference Resolution	Mencari rujukan dari setiap *mention*.	Aturan dengan mencari entitas yang sama yang disebutkan sebelumnya, baik di teks maupun di *dialog state*
Entity Splitting	Pengelompokan *entity-type* dan *slot-value* dalam entitas-entitas. Setiap penyebutan satu *entity-type* bisa menjadi beberapa entitas.	Aturan dengan mencari *slot-value* yang konflik, yaitu jika ada *slot-value* yang berbeda untuk *slot* yang sama, maka akan dipisahkan menjadi beberapa entitas
Entity Merging	Untuk entitas yang memiliki rujukan di teks, entitas tersebut akan digabungkan dengan rujukannya.	Menggabungkan entitas yang merujuk pada entitas lain di teks.
Intent Classification	Menentukan *user intent*, baik secara global maupun untuk setiap entitas.	Aturan berdasarkan status, keberadaan entitas, dan kata kunci

7.4 Modul *Dialog Manager*

Dialog manager terdiri dari dua bagian utama, yaitu *dialog state update* dan *action selection*. *Dialog state update* adalah proses pembaruan *dialog state* berdasarkan *dialog act* dari pengguna. Sedangkan *action selection* berfungsi untuk menentukan aksi sistem selanjutnya berdasarkan *dialog state*.

Pada *dialog state update*, *dialog state* diperbarui berdasarkan *dialog act*. Karena setiap entitas memiliki *dialog state* sendiri, maka *dialog state update* juga dilakukan per entitas. Selain itu dilakukan juga *dialog state update* global. Prosesnya *update* entitas dimulai dengan menambahkan entitas baru, kemudian status entitas di-*update* satu per satu berdasarkan suatu aturan *update* entitas. Entitas akan terus di-*update* sampai menemukan status yang menunggu *input* dari *user*, atau status final. Jika sudah mencapai status final, maka entitas tersebut dianggap sudah selesai diproses.

Setelah semua entitas ter-*update*, sistem masuk ke proses *update* global, yang mengikuti . Untuk *update* global, status global di-*update* berdasarkan suatu aturan *update* global. *Dialog state* global juga akan di-*update* terus menerus hingga

menemukan status yang menunggu *input* dari *user*, status final, atau status yang menunggu pemrosesan semua entitas selesai.

Setelah *dialog state update* selesai, jika status global belum mencapai status final, sistem perlu menentukan aksi apa yang harus dilakukan agar *user* memasukkan *input* yang sesuai untuk proses selanjutnya. Proses inilah yang dilakukan pada bagian *action selection*. sistem akan memilih aksi selanjutnya dari status *global* dan semua status entitas, dengan urutan pemilihan sebagai berikut.

1. Entitas yang terakhir dibahas dan belum selesai. Jika ada beberapa, dipilih salah satu.
2. Entitas yang belum selesai selain yang terakhir dibahas. Jika ada beberapa, dipilih salah satu.
3. Jika semua entitas sudah selesai, maka yang dipilih adalah status global.

7.5 Modul *Natural Language Generation*

Modul NLG berfungsi untuk mengubah aksi yang dipilih sistem menjadi teks sebagai *output* dari sistem dialog yang akan dibaca oleh *user*. Pada modul ini, tidak ada yang khusus untuk proses sistem dialog multi entitas. Karena itu, metode yang dipilih untuk proses ini adalah metode yang cukup sederhana namun masih banyak digunakan, yaitu *template*. Setiap aksi sistem memiliki beberapa *template* menghasilkan *output* sistem.

8. Hasil dan Pembahasan

Sistem dialog yang telah diimplementasikan kemudian dievaluasi dengan menggunakan data yang telah dikumpulkan, yang dibahas pada bagian 4. Sistem akan dibandingkan dengan beberapa sistem dialog lain yang menggunakan alur yang berbeda, yaitu model *slot-filling* biasa, dan model berbasis *deep learning*. Model dialog *slot-filling* dipilih karena model ini yang masih umum digunakan untuk sistem dialog *task-oriented*. Sistem dialog dengan model *slot-filling* yang akan dibandingkan dibagun dengan framework Rasa (Bocklisch dkk., 2017).

Kemudian model lain yang dibandingkan adalah model berbasis *deep learning*. Model ini dipilih karena merupakan model sistem dialog yang terbaik untuk saat ini. Dengan begitu, dapat diketahui apakah alur sistem dialog yang dibangun dapat lebih baik daripada alur yang dipelajari secara implisit oleh sistem berbasis *deep learning*. Untuk model ini, digunakan ChatGPT (https://chat.openai.com/). Pengujian dengan ChatGPT dilakukan pada *website*-nya dengan metode *prompting*.

Ada banyak aspek yang bisa diukur untuk mengetahui kinerja dari sistem dialog yang dibangun. Agar lebih terstruktur, evaluasi ini dibagi menjadi tiga bagian, yaitu ekstraksi entitas (bagian 8.1), perujukan entitas (bagian 8.2), dan pemrosesan dialog (bagian 8.3).

8.1 Ekstraksi Entitas

Ada tiga aspek yang akan dinilai untuk mengetahui ketepatan dari proses ekstraksi entitas yang telah dilakukan. Ketiga aspek tersebut adalah hasil ekstraksi entitas dari teks, hasil ekstraksi relasi, dan hasil pengelompokan entitas. Untuk mengukur ketepatan dari proses ekstraksi entitas berdasarkan data, digunakan tiga macam pengukuran, yaitu precision, recall, dan nilai F1. Tabel 2 menunjukkan hasil evaluasi ekstraksi entitas berdasarkan ketiga aspek yang telah disebutkan.

Tabel 2. Hasil Evaluasi Ekstraksi Entitas

Aspek		Precision	Recall	F1
Entity Extraction	Rasa	98.36%	72.64%	83.57%
	ChatGPT	99.52%	99.76%	99.64%
	Multi Entitas	97.94%	95.96%	96.94%
Relation Extraction	Rasa	-	-	-
	ChatGPT	98.40%	98.09%	98.25%
	Multi Entitas	95.31%	90.02%	92.59%
Entity Grouping	Rasa	71.59%	39.62%	51.01%
	ChatGPT	91.82%	91.82%	91.82%
	Multi Entitas	79.82%	80.56%	80.18%

Untuk *entity extraction*, hasil dari sistem dialog multi entitas masih berada di bawah ChatGPT, meskipun perbedaannya tidak terlalu signifikan. Sedangkan jika dibandingkan dengan model *slot-filling* Rasa, sistem dialog multi entitas ini dapat dikatakan sudah cukup baik. Kesalahan sebagian besar terjadi akibat adanya entitas yang namanya tidak terdaftar dalam kamus. Hal ini terjadi karena sistem masih menggunakan metode pencocokan teks, sehingga entitas tidak dapat diekstrak jika cara penulisannya berbeda. Contohnya antara lain, sistem tidak bisa mengekstraksi *entity-type* “es krim” pada teks “escream”, serta tidak bisa mendeteksi entitas yang tidak terdaftar pada sistem, misalnya “tahu”, “tempe”, dan “sambel ijo”. Sebaliknya, sistem juga salah mengekstraksi kata-kata yang bukan bagian dari entitas apabila sama dengan daftar, misalnya “teh” dalam “Mie kuahnya delapan porsi Teh”.

Kemudian hasil untuk *relation extraction* juga lumayan baik, masih di bawah ChatGPT, namun tidak terlalu signifikan. Sebagian kesalahan yang terjadi merupakan akibat dari kesalahan pada *entity extraction*. Sedangkan sebagian yang lainnya murni karena kesalahan dari *relation extraction*, sebagian besar terjadi pada ekstraksi relasi angka. Contohnya pada teks “yang ayam 3 level 1, yang 2 lagi level 4”, angka “3” dihubungkan dengan ke *slot-value* “ayam”, padahal seharusnya ke “level 1”. Selain itu, banyak juga angka yang tidak berhasil dihubungkan dengan entitas karena berjarak lebih dari jarak maksimal yang ditentukan. Selain itu, ada juga kesalahan *relation extraction* yang terjadi akibat tidak bisanya mendeteksi *entity-type*. Sebagai contoh pada teks input “3 nasi merah pake ayam saus tiram”. Pada contoh tersebut, *user* tidak menyebut *entity-type*, sehingga sistem tidak dapat

mengekstraksi relasinya. Ada juga 1 *slot-value* yang seharusnya memiliki 2 relasi, misalnya pada "1 jus alpukat sedikit gula dan es". *Slot-value* "sedikit" harusnya bisa mengisi *slot* "jumlah es" dan *slot* "jumlah gula" secara bersamaan. Namun sayangnya kasus yang seperti ini masih belum terdeteksi oleh sistem.

Terakhir untuk *entity grouping*, ketepatannya sudah cukup, walaupun ChatGPT sudah jauh lebih baik. Hal ini terjadi terutama karena bawaan dari kesalahan pada *entity extraction* dan *relation extraction*, karena entitas yang dihasilkan sistem harus sama persis dengan entitas pada pelabelan untuk dianggap benar. Namun nilai ini sudah cukup baik dibandingkan dengan sistem berbasis *slot-filling*. Hal ini terjadi karena sistem berbasis *slot-filling* tidak dapat mengekstraksi lebih dari satu entitas. Namun ada juga teks input yang mengalami kesalahan murni karena *entity grouping*. Salah satu contohnya pada teks "Paket nasinya 2 cumi saus pedas, 3 cumi saus tiram, 3 cumi sambal matah, yang saus tiram pake nasi putih ya, yang lain nasi merah". Pada contoh ini, sistem belum bisa mendeteksi bahwa yang menggunakan "nasi merah" adalah selain "saus tiram", yaitu "saus pedas" dan "sambal matah".

8.2 Perujukan Entitas

Masalah perujukan entitas ditangani oleh bagian *mention classification* dan *reference resolution*. Namun evaluasinya akan digabungkan karena hampir semua kesalahan hanya terjadi pada bagian *mention classification* saja. Hal tersebut dilakukan sebelum *entity grouping*, sehingga hasilnya disebut *text reference*. Selain itu, ada juga *group reference* yang menunjukkan akurasi dari perujukan entitas setelah dilakukan *grouping*. Hasilnya dapat dilihat pada Tabel 3 berikut. Sistem dialog berbasis Rasa tidak dibandingkan pada perujukan entitas ini, karena sistem tersebut belum memiliki modul *reference resolution*. Sedangkan ChatGPT tidak disertakan nilai untuk *text reference*-nya karena ChatGPT langsung mengeluarkan hasil akhir sehingga sulit diketahui prosesnya.

Tabel 3. Hasil Evaluasi Perujukan Entitas

Aspek		Precision	Recall	F1
Text Reference	ChatGPT	-	-	-
	Multi Entitas	91.18%	47.69%	62.63%
Group Reference	ChatGPT	94.97%	94.97%	94.97%
	Multi Entitas	83.03%	83.80%	83.41%

Pada Tabel 3, dapat dilihat bahwa *precision* dari *text reference* cukup tinggi dibandingkan dengan *recall*-nya. Hal ini berarti sebagian besar rujukan yang terdeteksi itu benar, sedangkan banyak rujukan yang tidak terdeteksi oleh sistem. Hal ini terjadi karena kekurangan pada metode *mention classification* yang hanya memperhatikan entitas satu per satu, dan tidak memperhatikan penyebutan entitas yang berkelompok. Misalnya pada kalimat "nasi goreng seafood yang level 5 nya dibatalkan", kata "level 5" akan terdeteksi sebagai mention karena diawali dengan

“yang” dan diakhiri “-nya”. Sedangkan kata “seafood” yang sekelompok dengan “level 5” seharusnya juga merupakan rujukan, tapi tidak terdeteksi oleh sistem. Ada juga *slot-value* yang didahului kata “yang”, tapi diselingi dengan kata lain sehingga tidak terdeteksi, misalnya pada teks “yang toping baso”.

Sedangkan untuk *group reference*, akurasi yang dihasilkan sistem ternyata lebih baik daripada *text reference*. Hal ini terjadi karena entitas yang tidak merujuk dianggap rujukannya adalah “NEW”, dan ini terjadi cukup banyak, yaitu lebih dari 50%, sehingga meningkatkan hasil *group reference*. Namun nilai ini masih berada di bawah nilai ChatGPT. Sebagian besar kesalahan untuk *group reference* terjadi akibat kesalahan *text reference*. Namun, ada juga yang murni dari kesalahan *group reference*, misalnya untuk entitas yang dirujuk secara imlisit, ketika entitas yang dirujuk tidak ada pada *last entity*, maka tidak akan terdeteksi.

8.3 Pemrosesan Dialog

Masalah pemrosesan dialog dilakukan secara per entitas, dan ditangani pada bagian *intent classification* dan *dialog state update*. *Intent classification* menghasilkan *entity intent* dan *global intent*, sedangkan *dialog state update* menghasilkan *entity state* dan *global state*. Hasil evaluasi untuk pemrosesan entitas dapat dilihat pada Tabel 4, sedangkan hasil evaluasi globalnya dapat dilihat pada Tabel 5. Untuk hasil yang berdasarkan *entity*, pengukuran yang digunakan adalah *precission*, *recall*, dan nilai F1. Sedangkan untuk hasil global hanya digunakan akurasi saja, karena hanya ada satu label untuk setiap input teks, sehingga jumlah klasifikasi pasti sama dengan jumlah hasil pelabelan. Pada pengukuran ini, ChatGPT hanya disertakan pada aspek *entity intent*. Pada aspek lainnya, kinerja ChatGPT sulit diukur, karena hasil pemrosesan dari ChatGPT langsung berupa daftar entitas akhir, sehingga tidak diketahui *dialog state*-nya. Sedangkan *entity intent* dapat diperkirakan dari hasil akhir pemrosesan.

Tabel 4. Hasil Evaluasi Pemrosesan Dialog Per Entitas

Aspek		Precision	Recall	F1
Entity Intent	Rasa	87.50%	48.43%	62.35%
	ChatGPT	96.23%	96.23%	96.23%
	Multi Entitas	81.65%	82.41%	82.03%
Entity State	Rasa	93.18%	51.57%	66.40%
	Multi Entitas	87.16%	87.96%	87.56%

Tabel 5. Hasil Evaluasi Pemrosesan Dialog Global

Aspek		Akurasi
Global Intent	Rasa	91.47%
	Multi Entitas	92.73%
Global State	Rasa	79.84%
	Multi Entitas	88.64%

Dari Tabel 4 dan Tabel 5, dapat dilihat bahwa hasil klasifikasi *intent* biasanya lebih baik daripada *dialog state update*. Hal ini terjadi karena kesalahan pada klasifikasi *intent* pasti akan menyebabkan kesalahan pada *dialog state*, sehingga kesalahan pada *dialog state update* tentunya akan menjadi lebih banyak. Untuk klasifikasi *entity intent*, ChatGPT memang paling baik dengan nilai F1 sekitar 96%. Sedangkan pada semua aspek, sistem dialog multi entitas sudah lebih baik daripada Rasa. Untuk yang entitas, karena memang banyak entitas yang tidak terdeteksi oleh Rasa.

Kesalahan di klasifikasi *intent* entitas sendiri sebagian besar terjadi akibat kesalahan-kesalahan pada komponen sebelumnya. Selain itu, ada yang mengalami kesalahan pada ekstraksi entitas, ekstraksi relasi, ekstraksi mention, atau pembentukan grup. Kesalahan yang murni dari klasifikasi *intent* adalah pada kalimat "eskrimnya ganti aja jadi roti bakar kacang". Hal ini terjadi karena kurang lengkapnya aturan untuk perubahan entitas, terutama apabila ada perubahan *entity-type*.

Sedangkan untuk *intent* global, hasilnya sudah lebih baik daripada *intent* entitas. Hal ini terjadi karena sebagian besar *input* terkait dengan entitas. Untuk kesalahannya semua terkait dengan kesalahan pada komponen sebelumnya, terutama ekstraksi relasi, yaitu terdapat *slot-value* yang tidak diketahui *entity-type*-nya. Hal ini menyebabkan *entity group* tidak dapat terbentuk, sehingga dianggap tidak ada entitas dan mempengaruhi klasifikasi *global intent.*

Kemudian untuk kesalahan pada *dialog state update*, sebagian besar kesalahan terjadi karena kesalahan klasifikasi *intent*. Selain itu, untuk *update entitas*, penyebab lainnya adalah kesalahan ekstraksi entitas, sehingga *slot-value* dari entitas tersebut dianggap belum lengkap. Kesalahan lainnya terjadi akibat kegagalan perujukan entitas, sehingga yang diubah adalah status dari entitas yang salah. Sedangkan untuk *global state*, kesalahannya sebagian besar terjadi akibat kesalahan pada *entity state*-nya. Jika ada entitas yang belum diangap belum selesai diproses, *global state* akan terus berada pada status memproses entitas.

8.4 Kelebihan dan Kekurangan Sistem Dialog Multi Entitas

Secara umum, hasil pemrosesan sistem dialog multi entitas yang diusulkan sudah lebih baik daripada sistem dialog berbasis *slot-filling*. Namun, jika dibandingkan dengan proses yang memanfaatkan ChatGPT yang berbasis *deep learning*, seringkali proses berbasis *deep learning* memberikan hasil yang lebih baik.

Ada beberapa kelebihan dari sistem dialog multi entitas dibandingkan yang berbasis *slot-filling* biasa, antara lain sebagai berikut.

1. Sistem yang diusulkan dapat mengekstraksi banyak entitas sekaligus.
2. Sistem yang diusulkan dapat menangani pemisahan entitas apabila ada lebih dari satu *slot-value* yang disebut untuk *slot* yang sama.
3. Sistem yang diusulkan dapat merujuk entitas yang telah disebut sebelumnya.

4. Sistem yang diusulkan dapat mengganti nilai dari suatu *slot* pada entitas yang dirujuk, atau membatalkannya.

Meskipun begitu, ternyata sistem yang memanfaatkan model *deep learning* seperti ChatGPT pun mampu menangani hal-hal tersebut, bahkan lebih baik, terutama untuk proses pengelompokan entitas dan perujukan entitas. Hal ini terjadi karena model *deep learning* dibangun dengan data yang sangat banyak, yang kemungkinan mencakup berbagai hal termasuk pengelompokan entitas dan perujukan entitas. Namun ada beberapa kelebihan dari sistem dialog multi entitas berbasis aturan yang diusulkan jika dibandingkan dengan model *deep learning*, antara lain sebagai berikut.

1. Sistem yang diusulkan tidak membutuhkan data yang banyak untuk melatih model.
2. Sistem yang diusulkan tidak membutuhkan perangkat dengan kemampuan komputasi yang tinggi untuk melatih model.
3. Sistem yang diusulkan tidak membutuhkan waktu yang lama untuk proses pelatihan model.

Walaupun sudah lebih baik, ada beberapa hal yang ternyata masih belum berhasil ditangani oleh ChatGPT, terutama terkait penalaran dan perhitungan jumlah. Namun, model yang digunakan pada ChatGPT ini memang masih model umum. Untuk membuatnya menjadi lebih spesifik, mungkin perlu dicoba proses fine tuning pada model ChatGPT dengan data dialog multi entitas.

9. Penerapan Model untuk Domain Lain

Sistem dialog multi entitas ini diimplementasikan dalam domain pemesanan makanan. Namun sistem ini juga dapat diimplementasikan pada domain pemesanan lainnya. Untuk mengimplementasikan sistem ini dalam domain pemesanan lain, ada beberapa hal yang harus disesuaikan, antara lain sebagai berikut.

1. Daftar entitas, yang terdiri dari daftar *entity-type*, daftar *slot* untuk setiap *entity-type*, dan daftar *value* untuk setiap *slot*.
2. Kamus berisi daftar kata yang mungkin digunakan untuk masing-masing kata di *entity-type*, *slot-name*, dan *slot-value*.

Selain itu, ada beberapa hal pada domain pemesanan juga yang belum diimplementasikan pada penelitian ini, misalnya menanyakan alternatif atau rekomendasi, menanyakan detail, perhitungan harga, pembayaran, serta pengiriman. Kemampuan-kemampuan ini secara umum bisa ditambahkan dengan cara menyesuaikan beberapa hal, antara lain sebagai berikut.

1. Menambah alur dialog untuk hal yang ingin ditambahkan pada aturan *update* entitas maupun global.
2. Menambah *intent* pada daftar *intent* entitas dan global jika dibutuhkan, termasuk menambah aturan *intent classification* untuk *intent* tersebut.
3. Menambah *template* NLG untuk setiap status sistem yang baru ditambahkan.

4. Menambahkan fungsi pada NLG jika dibuthkan penulisan khusus terkait data tertentu.
5. Menambahkan fungsi untuk menjalankan *task* yang perlu dilakukan.

Sedangkan jika sistem dialog multi entitas ini ingin diimplementasikan untuk domain lain, hal yang harus ditambahkan kurang lebih sama, yaitu daftar entitas dan kamus, alur dialog dalam bentuk FSM, daftar *intent* dan aturannya, daftar *template* NLG dan fungsi NLG jika dibutuhkan, serta fungsi-fungsi untuk menjalankan *task* sesuai dengan kebutuhan domain yang diimplementasikan.

10. Kesimpulan

1. Pada penelitian ini telah dibangun *framework* sistem dialog multi entitas berbasis aturan pada domain pemesanan. Sistem yang diajukan untuk sistem dialog multi entitas terdiri dari tiga modul utama, yaitu NLU, DM, dan NLG. Modul NLU terdiri dari beberapa komponen untuk mengekstraksi entitas-entitas yang terdapat dalam teks *input*, yaitu ekstraksi *entity-type* dan *slot-value*, ekstraksi relasi, dan penggabungan entitas. Setelah entitas terekstraksi, dilakukan pemrosesan dialog secara terpisah antar entitas, dan juga pemrosesan global. Pemrosesan ini dimulai dari klasifikasi *intent* pada NLU, dan *dialog state update* pada DM. Kemudian pada DM juga dilakukan *action selection* untuk memilih entitas mana yang akan dibahas berikutnya oleh sistem. Terakhir, NLG mengubah aksi sistem menjadi teks *output*.
2. Pada penelitian ini juga dikumpulkan *dataset* dialog multi entitas dengan metode WoZ. *Dataset* yang telah dikumpulkan dan dianotasi berisi 50 percakapan dengan 220 teks *input*. Data ini digunakan untuk mengevaluasi kinerja sistem dialog multi entitas yang dibangun.
3. Secara umum, alur sistem yang diajukan sudah dapat menyelesaikan masalah-masalah pada dialog multi entitas. Meskipun begitu, masih ada beberapa kekurangan untuk masing-masing komponen sistem. Sistem sudah bisa melakukan ekstraksi entitas dengan nilai F1 sebesar 80,2% dan perujukan entitas dengan nilai F1 sebesar 83,4%. Sedangkan keberhasilan sistem melakukan klasifikasi *intent* untuk setiap entitas dengan nilai F1 82%, dan menentukan final state untuk setiap entitas dengan nilai F1 sebesar 87,6%. Beberapa kekuarangan yang masih perlu diperbaiki antara lain masih banyaknya kesalahan ekstraksi entitas. Selain itu juga proses perujukan entitas yang seringkali belum dapat mendeteksi mention dengan baik. Kesalahan-kesalahan tersebut juga dapat menyebabkan kesalahan pada komponen selanjutnya.
4. Kinerja sistem yang dibangun secara umum sudah lebih baik daripada model berbasis *slot-filling*, namun masih perlu ditingkatkan jika dibandingkan dengan model berbasis *deep learning*. Sistem berbasis *deep learning* dapat melakukan beberapa *task* dengan lebih baik, padahal sistem tidak dilatih dengan data khusus untuk domain pemesanan ataupun untuk dialog multi entitas. Namun, jika mempertimbangkan ketersediaan data dan kemampuan komputasi perangkat

yang digunakan, maka sistem dialog multi entitas berbasis aturan ini masih dapat digunakan.

5. Untuk aspek praktis, hasil dari penelitian ini dapat dimanfaatkan untuk otomatisasi pemesanan melalui aplikasi pesan instan. Misalnya untuk pemesanan makanan, barang, tiket, ataupun hotel. Selain itu, jika digabungkan dengan sistem ASR dan TTS, sistem ini seharusnya juga dapat dimanfaatkan untuk pemesanan secara otomatis melalui telepon atau mesin penjawab otomatis.

11. Tindak Lanjut

Secara umum, masih diperlukan model yang lebih fleksibel untuk memodelkan dialog multi entitas, misalnya untuk pemisahan entitas, kasus ambigu, serta penalaran dan perhitungan jumlah. Selain itu, masih banyak masalah dalam dialog multi entitas yang belum dibahas pada penelitian ini. Beberapa fenomena yang dapat diteliti lebih lanjut antara lain adalah dialog dengan entitas yang lebih bebas yang bervariasi, misalnya memiliki tipe entitas yang tidak terbatas, *slot* dengan banyak *value*, serta menangani hubungan antar entitas. Selain itu, *dataset* untuk dialog multi entitas dapat dibuat lebih banyak lagi, sehingga bisa digunakan untuk metode-metode berbasis statistik, misalnya dengan *deep learning*.

Daftar Referensi

Balaraman, V., Sheikhalishahi, S., dan Magnini, B. (2021). Recent Neural Methods on Dialogue State Tracking for Task-Oriented Dialogue Systems: A Survey. *Proceedings of the 22nd annual meeting of the special interest group on discourse and dialogue*, pages 239–251.

Bocklisch, T., Faulkner, J., Pawlowski, N., dan Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. arXiv preprint arXiv:1712.05181.

Budzianowski, P., Wen, T. H., Tseng, B. H., Casanueva, I., Ultes, S., Ramadan, O., & Gašić, M. (2018): Multiwoz--a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *2018 Conference on Empirical Methods in Natural Language Processing*, 5016–5026.

El Asri, L., Schulz, H., Sharma, S., Zumer, J., Harris, J., Fine, E., dan Suleman, K. (2017): Frames: A corpus for adding memory to goal-oriented dialogue systems. *18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219.

Gangadharaiah, R. dan Narayanaswamy, B. (2020). Recursive Template-based Frame Generation for Task Oriented Dialog. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2059–2064.

Gubbala, P. dan Zhang, X. (2021). A Sequence to Sequence Model for Extracting Multiple Product Name Entities from Dialog. arXiv preprint arXiv:2110.14843.

Henderson, M., Thomson, B. dan Williams, J.D. (2014a): The Second Dialog state Tracking Challenge, *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 263-272.

Henderson, M., Thomson, B. dan Williams, J.D. (2014b): The third dialog state tracking challenge, *Spoken Language Technology Workshop (SLT)*, IEEE, 324-329.

Lee, S. dan Stent, A. (2016). Task Lineages: Dialog State Tracking for Flexible Interaction. *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 11–21, Los Angeles, USA.

Papangelis, A. dan Ultes, S. (2020). Towards meaningful, grounded conversations with intelligent agents. arXiv preprint arXiv:2006.15768.

Schulz, H., Zumer, J., El Asri, L., dan Sharma, S. (2017): A Frame Trackin Model for Memory-Enhanced Dialogue Systems. *2nd Workshop on Representation Learning for NLP*, 219–227.

Ultes, S., Budzianowski, P., Casanueva, I., Rojas-Barahona, L., Tseng, B. H., Wu, Y. C., Young, S., dan Gašić, M. (2019). Addressing objects and their relations: The conversational *entity* dialogue model. *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 273–283.

Walker, N. T., Ultes, S., & Lison, P. (2022): GraphWOZ: Dialogue Management with Conversational Knowledge Graphs. *arXiv preprint arXiv:2211.12852*.

Williams, J., Raux, A., Ramachandran, D., dan Black, A. (2013): The *dialog state* tracking challenge, *SIGDIAL 2013 Conference*, 404-413.

Ye, F., Manotumruksa, J., & Yilmaz, E. (2022): Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 351–360.

Riwayat Hidup

Nama	: Athia Saelan, S.T., M.T.
Telepon	: 08156226105 / 081910109010
Email	: athiasaelan@gmail.com

Penulis lahir di Bandung pada tanggal 7 Oktober 1989. Penulis merupakan putri pertama dari empat bersaudara dari pasangan Bapak Muchsin Saelan dan Ibu Evi Sofia. Penulis lulus dari SMAN 3 Bandung pada tahun 2008, kemudian mendapatkan gelar sarjana dari Program Studi Teknik Informatika ITB pada tahun 2012. Penulis sempat bekerja sebagai *programmer* di suatu perusahaan IT di Bandung, sebelum mendapatkan beasiswa PMDSU pada tahun 2015 untuk melanjutkan studi S2 dan S3 di ITB. Penulis memperoleh gelar magister dari Program Studi Magister Informatikan ITB pada tahun 2017. Penulis menikah dengan seorang suami bernama Ahmad Syaugi pada tahun 2018, dan dikaruniai seorang anak bernama Muhammad Ali Umar pada tahun 2019.

Daftar Publikasi Terkait Penelitian

Saelan, A. , Purwarianti, A. , dan Widyantoro, D. H. (2018): Answering Comparison in Indonesian Question Answering System with Database, *International Journal on Electrical Engineering and Informatics*, 10(4), 783 – 798.

Ucapan Terima Kasih

Puji syukur kepada Allah SWT yang telah memberikan segala rezeki dan kemudahan serta mengizinkan penulis untuk menyelesaikan disertasi ini. Penulis juga mengucapkan terima kasih yang sebanyak-banyaknya kepada para pembimbing, Pak Rinaldi, Bu Ayu, Pak Rila, serta Pak Dwi yang telah mendahului kita (semoga Allah merahmati beliau), yang telah membantu proses penyusunan disertasi ini dari awal sampai akhir, dan mempermudah proses penyelesaian disertasi ini. Tak lupa juga pada para penguji dan *reviewer* yang telah memberikan banyak usulan dan masukan mulai dari awal disusunnya disertasi ini, Pak Bambang, Bu masayu, Bu Ade, Bu Ulfa, Bu Putri, Bu Dessi, serta Pak Iping yang telah mendahului kita (semoga Allah merahmati beliau). Terima kasih juga untuk Kemenristekdikti yang telah memberikan dukungan finansial pada awal masa studi doktor lewat beasiswa PMDSU, Pak Umar sebagai kaprodi, dan Bu Nur yang telah mempermudah proses-proses administrasi, serta teman-teman penulis yang sudah dimintai tolong pada proses pengumpulan data. Terakhir, penulis juga mengucapkan terima kasih yang banyak kepada keluarga penulis, orangtua yang selalu memberikan dukungan dan doa, suami yang selalu menyemangati agar proses ini bisa selesai dan telah banyak membantu, serta anak yang turut mendukung penyelesaian disertasi ini.