BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Berdasarkan pada teori dari penelitian saat ini, diperlukan data pendukung yang bisa dijadikan bahan pertimbangan dalam penelitian dengan memanfaatkan penelitian terdahulu yang relevan serta mencakup pembahasan yang dibahas dalam penelitian ini. Dari beberapa referensi penelitian terdahulu yang didapatkan, terdapat perbedaan yang dapat dijadikan perbandingan. Penelitian dengan tema atau judul penelitian yang hampir sama diantaranya dapat dilihat pada **Tabel 2.1** :

Tabel 2.1 Penelitian Terdahulu

1.	Judul Penelitian	Chatbot Development for an Interactive Academic Information Services using the Rasa Open Source Framework (Ruindungan et al., 2021)
	Uraian Penelitian	Penelitian yang dilakukan berpusat pada pengembangan *chatbot* layanan akademik kampus menggunakan platform RASA dengan memanfaatkan sistem pengembangan *chatbot* berbasis GUI bernama RASA X untuk mengolah data FAQ dari layanan akademik kampus sehingga menjadi sebuah *dataset* yang akan dilatih untuk *chatbot*. Hasil pengujian akhirnya sendiri berhasil didapatkan tingkat akurasi rata-rata sebesar 99,5% serta rata-rata tertimbang sebesar 99,55% pada *precision*, *recall*, dan *f1-score* dari keseluruhan *dataset* yang memiliki 12 *intent* dan total 188 sampel kalimat pertanyaan.
2.	Judul Penelitian	Chatbot-based Information Service using RASA Open-Source Framework in Prambanan Temple Tourism Object (Pradana et al., 2022)

	Uraian Penelitian	Penelitian yang dilakukan berpusat pada pengembangan *chatbot* layanan informasi objek wisata menggunakan platform RASA agar dapat menjadi sistem *Question Answering System* secara daring dan dari hasil pengujian akhir berhasil didapatkan akurasi rata-rata sebesar 91% dan berdasarkan *confusion matrix* yang dihasilkan berhasil didapatkan rata-rata tertimbang dari *precision* sebesar 97%, *recall* 94%, dan *f1-score* sebesar 95% dari keseluruhan *dataset* yang memiliki 21 *intent* dan total 291 sampel kalimat pertanyaan.
3.	Judul Penelitian	Aplikasi Chatbot Untuk Pengajuan Proses Pemesanan Surat di Fakultas Teknologi Industri Universitas Islam Indonesia (Sholeh, 2020)
	Uraian Penelitian	Penelitian yang dilakukan berpusat pada pengembangan aplikasi *chatbot* Pengajuan Proses Pemesanan Surat dengan menggunakan platform RASA sebagai sistem *chatbot* yang terimplementasi pada aplikasi berbasis Android yang dibuat menggunakan React Native. Aplikasi yang dirancang juga dapat menyimpan riwayat pesanan surat dengan memanfaatkan REST API. *Chatbot* ini diharapkan dapat dikembangkan lagi agar bisa menangani percakapan kontekstual yang lebih lengkap sehingga dapat menjadi sebuah sistem layanan akademik kampus dan dapat menangani kata-kata *typo* dengan penambahan data *training*.
4.	Judul Penelitian	Developing FB chatbot based on deep learning using RASA framework for university enquiries

		(Windiatmoko et al., 2020)
	Uraian Penelitian	Penelitian yang dilakukan berpusat pada pengembangan *chatbot* layanan akademik kampus menggunakan platform RASA dan juga memanfaatkan platform Facebook Messenger sebagai media penggunaan *chatbot* dalam pengembangan awalnya agar dapat langsung digunakan oleh user, dimana apabila dilakukan penelitian lebih lanjut lagi maka akan dilakukan percobaan implementasi *chatbot* pada platform tersendiri yang lebih mudah di kostumisasi.

Dasar pengembangan *chatbot* menggunakan platform RASA yang telah diteliti pada penelitian yang ada di tabel akan menjadi pegangan dalam melakukan penelitian ini, dan yang akan menjadi pembeda pada penelitian ini terhadap penelitian terdahulu adalah pemanfaatan pada sistem pengembangan *chatbot* berbasis GUI yaitu RASA X, dimana pada penelitian ini akan ditelusuri konsep dari pengujian *chatbot* dengan menggunakan RASA X dimana setiap percakapan baru antara *chatbot* yang diuji dengan pengguna akan masuk pada sistem pengembang *chatbot*, dimana data percakapan baru akan berpotensi untuk menjadi data tambahan untuk data *training* pada *chatbot*, selain itu keseluruhan data percakapan baru antara pengguna dengan *chatbot* yang masuk juga dapat menentukan sebagus apa performa dari *chatbot* menjawab pertanyaan pengguna.

Dan *chatbot* akan memiliki fitur Two Stage Fallback yang memungkinkan untuk mengatasi permasalahan kalimat dari pengguna yang ambigu serta permasalahan kata-kata *typo* yang dapat diatasi dengan pengaturan pada model *training chatbot*.

2.2 Landasan Teori

2.2.1 Chatbot

Chatbot adalah perangkat lunak yang diciptakan untuk dapat memenuhi kebutuhan informasi pengguna dari penggunaan sarana pemberian informasi yang sederhana seperti notifikasi/pengingat hingga *chatbot* yang dapat melakukan percakapan natural dengan manusia dan bahkan dapat mencapai tingkat tertinggi dimana *chatbot* dapat menjadi *virtual assistant* yang bisa mengerjakan pekerjaan manusia dan bahkan melakukan pekerjaan yang hanya dapat dilakukan oleh kecerdasan buatan. (Weidauer, 2018). *Chatbot* dapat diimplementasikan pada aplikasi perpesanan, website, *mobile app*, dan platform lainnya.

Hal pertama yang harus dilakukan untuk mengembangkan sebuah *chatbot* adalah memahami apa saja *intent* dari pengguna yang akan dilayani, *intent* merujuk pada informasi atau aksi yang diinginkan pengguna dari *chatbot*. Setiap *intent* dari *input* pengguna yang dipahami oleh *chatbot* akan membuat *chatbot* memberikan respon atau melakukan aksi sesuai dengan *rule* mengenai respon atau aksi yang akan dilakukan *chatbot* untuk suatu *intent. Chatbot* dapat memberikan respon dengan berbagai jenis media seperti teks, audio, foto, video, dan file-file seperti dokumen dan lainnya sesuai *intent* dari pengguna serta media berupa pilihan tombol-tombol perintah yang masing-masing memiliki respon/aksi yang berbeda-beda, sementara untuk aksi yang dapat dilakukan oleh *chatbot* contohnya seperti *chatbot* layanan *booking* dokter yang memungkinkan aksi untuk *booking* dokter sesuai dengan data yang telah di *input* pengguna ke *chatbot.* (Sholeh, 2020).

Chatbot juga dapat memahami banyak jenis media *input* dari pengguna, dimana *chatbot* yang dikembangkan dapat dibagi jenisnya berdasarkan jenis *input* yang dapat dipahami dan *chatbot* dapat dibuat untuk melayani satu atau lebih jenis *input* sesuai dengan kebutuhan pengguna dan kemampuan pengembang *chatbot*. Berikut penjelasan jenis-jenis *chatbot* berdasarkan jenis *input*-nya:

1. *Button-Based Chatbot*

Chatbot ini tidak memerlukan implementasi kecerdasan buatan, karena pengguna hanya perlu memilih tombol perintah yang disediakan di awal interaksi dengan *chatbot* untuk memperoleh respon/aksi yang diinginkan. Tombol perintah juga dapat memungkinkan untuk diisi percabangan ke tombol berikutnya sehingga level *chatbot* nya menjadi *decision tree-based*, yang mirip seperti ketika menggunakan layanan USSD (*Unstructured Supplementary Service Data*) operator seluler.

2. *Text Recognition-Based Chatbot*

Chatbot ini memerlukan implementasi *Natural Language Processing* (NLP) yang merupakan cabang dari *artificial intelligence* (AI) sehingga dapat memahami kalimat yang di *input* oleh pengguna. Ini adalah standar jenis *chatbot* yang paling umum dibuat dan merupakan dasar dari *chatbot* yang kedepannya memungkinkan untuk ditambahkan media *input* selain teks yang dapat dipahami oleh *chatbot*, dengan beberapa contohnya seperti berikut:

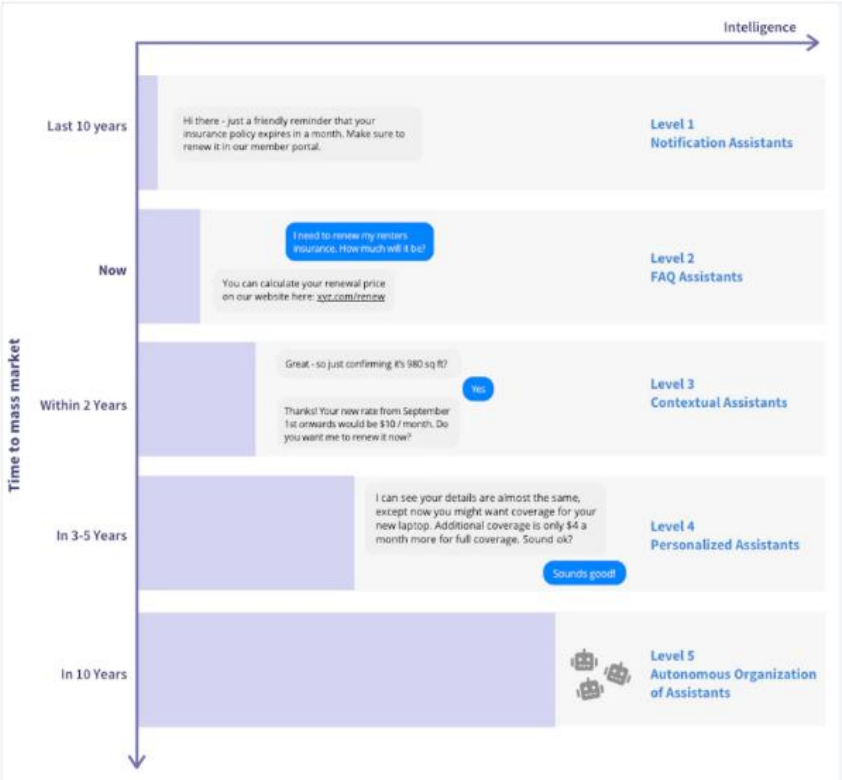
a. Speech Recognition

Speech recognition memungkinkan pengguna untuk mengucapkan perintah kepada *chatbot* dan ucapan pengguna akan diubah menjadi teks yang didapatkan dari kemampuan *chatbot* untuk memahami ucapan dari pengguna, dan kemudian teks akan dipahami oleh *chatbot* dengan *text recognition*. Hal ini sudah digunakan di semua produk *virtual assistant* yang populer seperti *Google Assistant, Siri,* dan *Alexa* dimana produk-produk ini juga dapat memberikan respon dengan media audio/suara yang memiliki ciri khas nya masing-masing.

b. Image Recognition

Chatbot ini memerlukan implementasi *Computer Vision* yang merupakan cabang dari AI untuk dapat mempelajari seluruh fitur dari sebuah gambar yang di *input* pengguna dan mencocokkannya dengan seluruh gambar yang telah dipelajari sebelumnya untuk menemukan maksud/*intent* dari gambar yang di *input*. *Image recognition* yang dibuat dapat dirancang untuk memberikan respon berupa

deskripsi dari gambar yang di *input* atau melakukan pencarian di mesin pencari berdasarkan gambar tersebut yang dapat dicoba pada fitur *Google Lens* yang juga dapat diakses melalui *Google Assistant*.



Gambar 2.1 Level Pengembangan *Chatbot* (Weidauer, 2018)

Pengembangan *chatbot* dapat mengacu pada 5 level seperti pada **Gambar 2.1** untuk memilih level yang sesuai dengan kebutuhan pengguna dan kemampuan pengembang *chatbot*. Semakin tinggi levelnya maka tingkat kecerdasan dari *chatbot* akan meningkat dan pengembangan akan lebih rumit serta membutuhkan waktu lebih lama dikarenakan jangkauan respon/aksi yang dapat diberikan oleh *chatbot* menjadi lebih luas sehingga akan lebih banyak data yang dipelajari. Berikut penjelasan setiap level pengembangan *chatbot*:

1. Level 1: *Notification Assistants*

Chatbot yang dikembangkan pada level ini hanya memiliki fungsi untuk mengirimkan pesan notifikasi kepada pengguna pada waktu tertentu yang dapat diatur oleh pengguna sendiri sebagai pengingat ataupun diatur oleh pengembang *chatbot* sendiri. *Chatbot* ini tidak dapat menerima masukan dari pengguna.

Contoh *chatbot* pada level ini adalah *chatbot* pengingat untuk segera melunasi utang *paylater* pada sebuah aplikasi *e-commerce*.

2. Level 2: *FAQ Assistants*

Chatbot yang dikembangkan pada level ini berfungsi menjawab setiap pertanyaan pengguna yang relevan dengan informasi yang tertuang pada daftar FAQ yang dijadikan sebagai *dataset* yang dibagi menjadi berbagai kelas informasi. Ini adalah *chatbot* yang paling umum dikembangkan dan digunakan pada kebanyakan website resmi sekarang. Beberapa *chatbot* pada level ini juga dikembangkan dengan kemampuan untuk dapat menjalankan sebuah percakapan *multi-step* dengan pengguna dimana *chatbot* memiliki inisiatif untuk salah satunya yaitu menanyakan atau memastikan kepada pengguna apakah pertanyaan yang dikirim sudah benar dan menjalankan *fallback classifier* apabila pertanyaan pengguna tidak dapat diklasifikasi dengan nilai probabilitas diatas *threshold* yang sudah ditentukan.

3. Level 3: *Contextual Assistants*

Chatbot yang dikembangkan pada level ini dapat bertindak selayaknya seorang *customer service*, dimana masukan dari pengguna dapat ditanggapi oleh *chatbot* dengan memberikan respon berupa pertanyaan balik kepada pengguna untuk melengkapi data yang dibutuhkan oleh *chatbot*. Misalnya ingin dibuat sebuah *chatbot* perbankan yang dapat bertindak sebagai *customer service* yang dapat memandu pengguna *chatbot* yang berkeinginan untuk membuka rekening baru dengan menanyakan satu persatu informasi yang dapat diambil melalui *chatbot* untuk kemudian dari setiap jawaban yang dikirim pengguna dijadikan kumpulan informasi yang dapat digunakan pengguna agar saat mengajukan tahap pembuatan rekening lanjutan yang harus secara luring prosesnya menjadi lebih singkat.

4. Level 4: *Personalized Assistants*

Chatbot yang dikembangkan pada level ini akan bertindak layaknya seorang personalia/asisten pribadi yang dapat memahami kebutuhan serta karakteristik dari penggunanya sehingga *chatbot* dapat menjalankan aksi berupa

memberikan saran dan mengatur keseharian dari pengguna dengan baik.

5. Level 5: *Autonomous Organization of Assistants*

Ini adalah level *chatbot* yang menjadikan *chatbot* layaknya seorang pekerja pada umumnya di setiap bidang industri seperti pemasaran, *human resources* dan bahkan seorang *programmer*. Ini berarti *chatbot* dapat mengenali setiap pengguna dan orang-orang disekitarnya dan dapat bekerja sama dengan manusia serta perangkat yang ada untuk mencapai tujuan utama dari instansi yang menggunakan *chatbot* ini. Level ini adalah wujud mutlak dari *chatbot* terbaik yang diharapkan dapat menjadi nyata dalam belasan tahun yang akan datang.

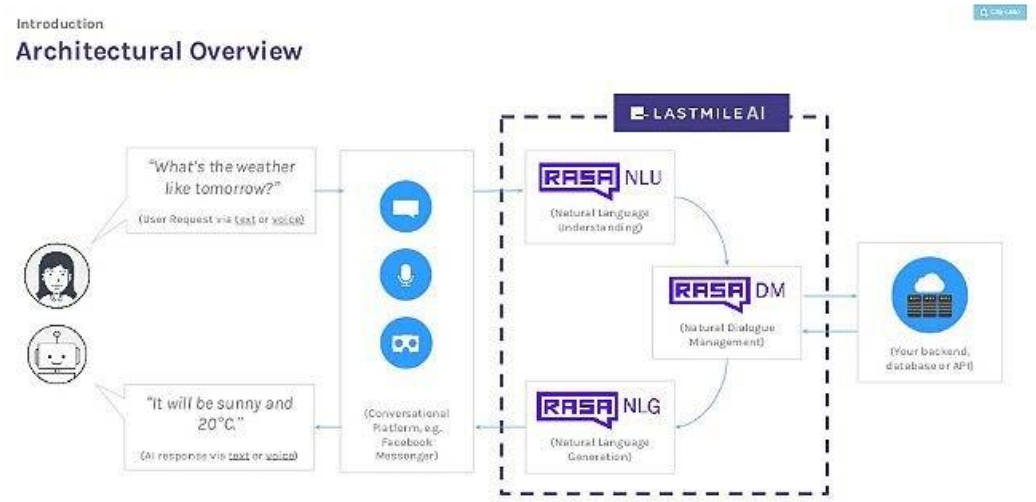
2.2.2 Natural Language Processing (NLP)

NLP merupakan cabang dari *artificial intelligence* yang dapat memproses data berupa teks atau audio dengan menggunakan bahasa natural. Model NLP bekerja dengan mengekstrak fitur-fitur yang terdapat dalam data untuk dijadikan sebagai identitas dari kelas informasi yang telah diatur. Contoh penerapan NLP bisa ditemui dalam penggunaan Search Engine, mesin penerjemah, speech recognition, dan sistem filter spam pada email. (Hanin, 2022)

Penerapan NLP di *chatbot* berfungsi agar *chatbot* dapat memahami input teks bahasa natural dari pengguna, yang kemudian akan dilakukan proses pembelajaran dari input pengguna dan kemudian diberikannya tanggapan atas input pengguna oleh *chatbot*. Untuk mencapai hal itu maka perlu dimanfaatkan dua komponen pada sistem NLP yaitu *Natural Language Understanding* (NLU) dan *Natural Language Generation* (NLG). Dalam NLP, proses awal yang dilakukan adalah mengubah data input pengguna yang masih berupa teks bahasa alami menjadi data terstruktur dimana data mentah telah melalui proses seperti *Tokenization, Featurizing,* dan seterusnya sehingga dapat dimengerti oleh mesin. Kemudian data yang telah diolah akan coba dipahami oleh NLU. Setelah data berhasil diklasifikasi oleh NLU, maka NLG akan bertugas untuk mengubah data terstruktur tadi menjadi teks yang dapat dipahami manusia agar kemudian dari NLG dapat dipilih *rules* yang tepat untuk merespon input dari pengguna. (Kumar, 2018)

2.2.3 RASA Open Source

RASA adalah platform *open source* yang digunakan untuk memproduksi *chatbot* AI dari level *chatbot* notifikasi hingga *chatbot* asisten pribadi berbasis AI dan dapat dihubungkan dengan layanan UI pada media seperti website dan sosial media menggunakan *endpoint* API. (Rasa, n.d.)



Gambar 2.2 Arsitektur RASA (Hanin, 2022)

RASA bekerja dengan melakukan pemahaman terhadap input teks pengguna di bagian RASA NLU, dimana RASA NLU merupakan alat pengolah bahasa alami yang didapatkan dari hasil *model pipeline training* yang menerapkan dasar dari pelatihan model NLP, dimana setelah melalui RASA NLU, maka data input pengguna tadi akan diubah menjadi data terstruktur yang nantinya dapat dibaca di RASA DM dimana mesin akan mengklasifikasi kelas informasi (*intent classification*) dari teks yang dimasukkan pengguna untuk mengetahui jenis informasi apa yang diinginkan dari pertanyaan pengguna. Kemudian dari hasil penentuan *intent* tadi, data terstruktur akan dikirim ke RASA NLG untuk diubah menjadi data teks kembali agar pada RASA NLG dapat dilakukan pemilihan kelas informasi mana yang paling akurat dengan pertanyaan yang diberikan pengguna untuk kemudian berdasarkan *rules* yang ada, *chatbot* akan mengirimkan jawaban atas pertanyaan tersebut. (Hanin, 2022)

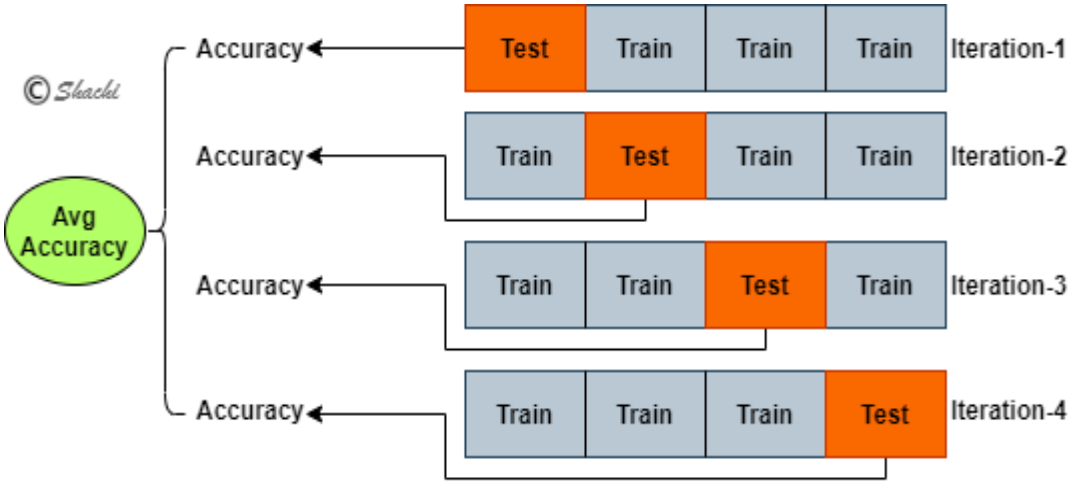
2.2.4 Conversation-Driven Development (CDD)

Conversation-Driven Development (CDD) adalah proses pengembangan *chatbot* dengan bertumpu pada hasil percakapan/*conversation* antara pengguna dengan *chatbot* yang telah dibagikan ke publik, dimana inputan data percakapan pengguna yang belum pernah dipelajari *chatbot* akan masuk ke dalam inventori percakapan yang dapat diamati oleh pengembang untuk dijadikan sebagai bahan pembelajaran baru untuk *chatbot*.

Proses pengembangan *chatbot* dengan metode CDD akan membuat proses pelatihan model *chatbot* hingga *deploy chatbot* ke server dilakukan secara berulang-ulang karena di setiap iterasi akan selalu didapatkan data percakapan baru, sehingga diperlukan penetapan jumlah minimum data percakapan pengguna yang baru untuk dipelajari oleh *chatbot*, misalnya *chatbot* hasil dari pembelajaran 1000 sampel kalimat akan dilakukan uji coba ke pengguna terbuka untuk didapatkan 250 sampel kalimat baru dari pengguna tersebut untuk kemudian *chatbot* akan dilatih kembali dengan tambahan kalimat baru. (Rasa, n.d.)

2.2.5 Cross Validation

Cross Validation adalah opsi yang dapat digunakan oleh pengembang *machine learning* yang tidak memiliki *dataset* yang cukup besar untuk melatih model *machine learning* dikarenakan data yang digunakan untuk *train* dan *test* akan dibagi dan digunakan untuk pelatihan model dalam jumlah iterasi lebih dari 1, dimana seluruh data dipastikan akan digunakan sebagai data *train* dan *test* dalam banyak iterasi pelatihan model.



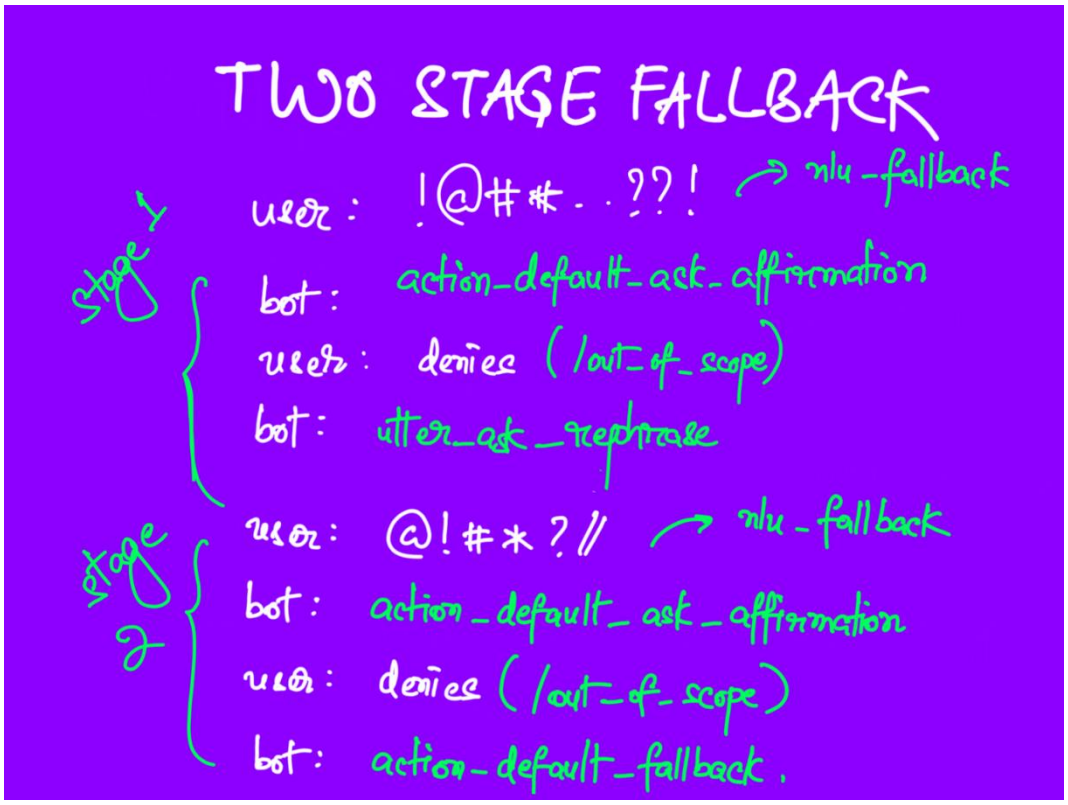
Gambar 2.3 *K-fold Cross-Validation* (*K* = 4). (Kaul, 2020)

Dataset akan dibagi menjadi beberapa bagian (*fold*) sebanyak *K* terlebih dahulu, dan kemudian pelatihan dan pengujian model akan dilakukan sebanyak *K* iterasi. Dalam setiap iterasinya akan diambil 1 bagian data untuk dijadikan data *test*, dimana tiap iterasi pasti akan memiliki data *test* yang berbeda. Ini dapat memastikan setiap data yang ada pada *dataset* dapat terpakai dengan efektif untuk pelatihan dan pengujian serta dapat memperoleh nilai akurasi yang lebih baik dikarenakan didapat dari hasil pemerataan tiap iterasi pelatihan dan pengujian model. (Kaul, 2020)

2.2.6 Two Stage Fallback

Two Stage Fallback adalah fitur yang disediakan oleh RASA untuk menyelesaikan permasalahan input teks pengguna yang tidak dapat diklasifikasi dengan baik oleh *chatbot* dimana nilai *confidence* dari klasifikasi tertinggi terhadap suatu kelas informasi tidak mencapai batas minimum yang diatur.

Hal-hal seperti pesan pengguna yang tidak dapat dimengerti sangat lumrah terjadi pada *chatbot*, bahkan ini juga dapat terjadi dalam percakapan antar manusia. Namun tentunya percakapan dengan *chatbot* harus terus berjalan dengan cara mengklarifikasi kepada pengguna mengenai informasi apa yang diinginkan, dan hal inilah yang coba diwujudkan dengan penggunaan Two Stage Fallback. (Wochinger, 2019).

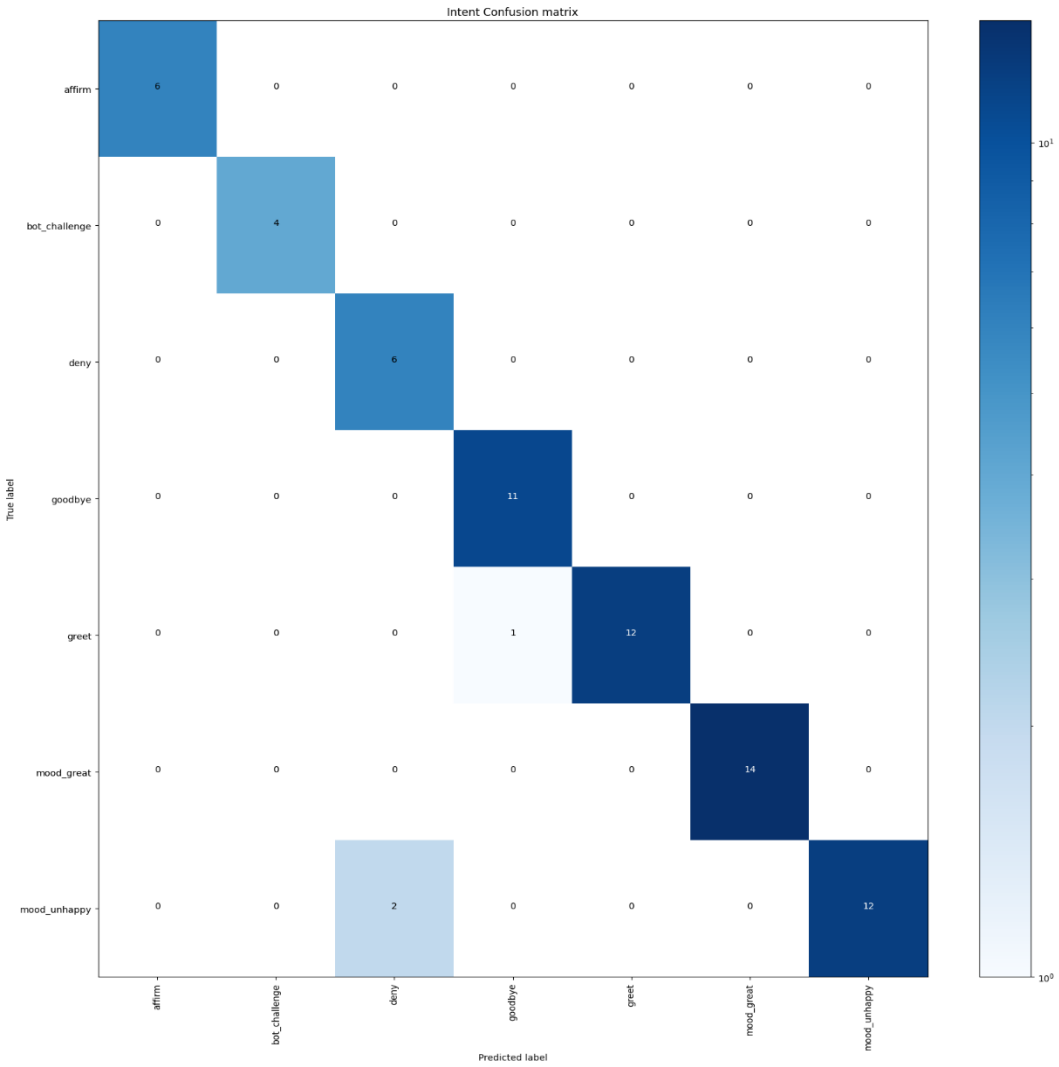


Gambar 2.4 Mekanisme Two Stage Fallback. (Karajgi, 2021)

Berdasarkan **Gambar 2.4**, Two Stage Fallback akan bekerja apabila klasifikasi kelas pesan yang dimasukkan pengguna tidak mencapai batas minimum nilai *confidence* yang diinginkan. *Chatbot* akan diinstruksikan untuk mengklarifikasi maksud dari pesan yang disampaikan oleh pengguna dengan menyediakan beberapa kelas informasi yang sekiranya dimaksud oleh pengguna, dan apabila tidak terdapat informasi yang diinginkan, maka *chatbot* akan meminta pengguna untuk menginput kembali pesan baru yang lebih jelas dan tidak ambigu. Namun apabila setelah pesan terbaru dimasukkan dan masih belum mencapai nilai *confidence* yang diinginkan, maka *chatbot* akan mengulang proses klarifikasi sebelumnya dengan menyediakan beberapa kelas informasi dengan nilai *confidence* tertinggi berdasarkan pesan pengguna. Namun apabila masih gagal maka *chatbot* akan menyampaikan pesan terakhir yang bisa berupa arahan untuk beralih ke tenaga manusia atau saran yang sekiranya dapat membantu pengguna. (Karajgi, 2021)

2.2.7 Parameter Evaluasi

Dalam melakukan pengujian model *chatbot*, maka diperlukan media untuk menampilkan hasil evaluasi dengan format yang jelas dan mudah dimengerti, salah satunya adalah dengan penggunaan *confusion matrix*. *Confusion matrix* adalah tabel yang menggambarkan hasil klasifikasi dari tiap kelas. (Hanin, 2022)



Gambar 2.5 Contoh Confusion Matrix di RASA

Berdasarkan contoh *confusion matrix* pada **Gambar 2.5** dapat dilihat bahwa jumlah hasil klasifikasi yang tepat di setiap kelasnya dapat dilihat pada kotak-kotak yang membentuk garis diagonal, dimana semakin pekat warnanya maka semakin banyak datanya, dan data-data yang berada pada area kotak luar

garis diagonal menandakan bahwa ada data yang tidak terklasifikasi dengan benar. Tampilan hasil evaluasi seperti ini akan sangat membantu dalam menentukan kelas-kelas informasi apa saja yang bermasalah dalam klasifikasinya dimana nilai *precision* dan *recall* nya tentu juga akan bermasalah.

Precision adalah nilai keakuratan hasil klasifikasi tiap kalimat pesan yang diklasifikasikan terhadap satu kelas informasi, dimana setiap pesan yang terklasifikasi terhadap satu kelas informasi tentunya akan ada yang bernilai benar (*True Positive*) dan salah (*False Positive*), sehingga rumus mencari *precision* dapat disimpulkan sebagai TP / (TP + FP).

Recall adalah nilai keakuratan hasil klasifikasi tiap kalimat pesan yang berasal dari satu kelas informasi. Nilai *recall* akan berkurang apabila terdapat kalimat dari kelas tersebut yang terklasifikasi ke kelas lain (*False Negative*), sehingga dapat disimpulkan rumus *recall* sebagai TP / (TP + FN).

Untuk menambah variasi nilai akurasi klasifikasi maka dibuatlah nilai kombinasi antara *precision* dan *recall*, yaitu *f1-score*, dengan rumus yaitu $\frac{(2*P)*R}{(P+R)}$. (P = *Precision*, R = *Recall*)

Penghitungan nilai rata-rata *precision, recall dan f1-score* untuk keseluruhan kelas dapat dilakukan dengan menggunakan *macro-average* dan *weighted-average*. Terdapat satu perbedaan antara *macro-average* dan *weighted-average*, dimana *macro-average* tidak memperhatikan jumlah sampel kalimat pada tiap kelas dalam penghitungan nilai rata-ratanya, sedangkan *weighted-average* menghitung nilai rata-rata dengan mentimbang jumlah sampel kalimat pada tiap kelas. (Hanin, 2022)