BAB III
ANALISIS DAN PERANCANGAN

3.1 Analisis

Analisis dilakukan untuk dapat memahami permasalahan yang dihadapi serta merancang kebutuhan sistem chatbot NLP yang memanfaatkan ANN sebagai model yang dioptimasi oleh SGD

3.1.1 Identifikasi Masalah

Dengan mengacu pada sistem informasi penerimaan mahasiswa baru di Ubhinus saat ini, ditemukan beberapa kendala. Salah satunya adalah kurangnya interaksi yang *real-time* antara calon mahasiswa dengan sistem. Informasi yang tersedia cenderung statis dan kurang personal, sehingga calon mahasiswa kesulitan dalam mendapatkan respon jawaban secara cepat atas pertanyaan yang cukup spesifik, terutama saat di luar jam operasional. Hal ini dapat mengakibatkan ketidakpuasan calon mahasiswa terhadap sistem informasi yang ada dan berpotensi mengurangi minat mereka untuk mendaftar. Dengan demikian, diperlukan solusi yang dapat menyelesaikan kendala ini, Salah satunya adalah dengan mengembangkan sistem chatbot berbasis NLP yang dapat memahami dan merespons bahasa manusia secara natural serta memberikan informasi secara *real-time*, relevan, dan personal kepada calon mahasiswa. Serta menggunakan model *Artifical Neural Network* (ANN) dalam mengklasifikasi teks yang diinputkan oleh pengguna, penggunaan algoritma *Stochastic Gradient Descent* (SGD) dalam optimasi pelatihan model dan TF-IDF untuk tahap *prepocessing*.

Tabel 3.1 Analisa *SWOT*

Strengths (Kekuatan)	Weaknesses (Kelemahan)
Sistem yang ada dapat dikembangkan lebih lanjut dengan fitur Chatbot NLP yang menggunakan model ANN dan optimasi SGD dengan *preprocessing* TF-IDF	Sistem saat ini kurang interaktif dan tidak memungkinkan calon mahasiswa untuk mendapatkan jawaban secara langsung terutama saat diluar jam operasional.
Opportunities (Peluang)	Threats (Ancaman)
Pengembangan sistem Chatbot NLP memungkinkan pemanfaatan teknologi terkini dalam bidang kecerdasan buatan serta dapat memberikan respon secara *real-time* kepada calon mahasiswa.	Teknologi terus berkembang, sehingga sistem yang dikembangkan perlu terus diupdate. Pengembangan dan pemeliharaan sistem chatbot membutuhkan biaya.

3.1.2 Pemecahan Masalah

Pemecahan masalah terhadap tantangan sistem informasi penerimaan mahasiswa baru Ubhinus dalam menjawab pertanyaan calon mahasiswa dapat diatasi dengan pengembangan Chatbot NLP yang menggunakan model ANN dan di optimasi menggunakan algoritma SGD. Pengembangan Chatbot ini memanfaatkan teknologi NLP dan ANN untuk dapat mengklasifikasikan teks yang diinputkan oleh pengguna. Sehingga dapat menghasilkan jawaban yang sesuai dengan pertanyaan pengguna serta penggunaan optimasi SGD untuk meminimalisir kesalahan dalam prediksi atau klasifikasi teks di model ANN. Serta penggunaan TF-IDF untuk tahap *preprocessing.*

3.2 Perancangan

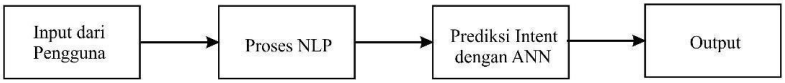
Pada tahap ini, berbagai aspek dan elemen sistem dikembangkan, seperti alur interaksi chatbot, pembuatan model yang akan digunakan, rancangan antarmuka penggunan hingga rancangan pengujian yang akan

diimplementasikan. Proses ini bertujuan untuk memastikan bahwa chatbot mampu bekerja secara efektif dalam mengenali pertanyaan dari pengguna.

3.2.1 Perancangan Sistem

Dalam mengembangkan sistem ini terdapat beberapa komponen yaitu, perancangan sistem chatbot NLP, perancangan model ANN, perancangan antarmuka dan rancangan pengujian. Perancangan sistem Chatbot NLP dimulai dari membuat mengimport beberapa libary yang dibutuhkan untuk mempermudah penulisan kode dan menghemat waktu sehingga proses pengembangan bisa menjadi lebih efisien. Perancangan antarmuka memiliki fokus untuk meningkatkan pengalaman pengguna dalam menggunakan sistem chatbot ini agar menjadi lebih efisien. Terakhir, rancangan pengujian digunakan untuk memastikan bahwa semua komponen di dalam sistem benar-benar berfungsi sesuai dengan ketentuan yang telah diterapkan melalui rangkaian uji coba dan evaluasi secara menyeluruh.

3.2.1.1 Arsitektur Chatbot



Gambar 3.1 Blok Diagram Chatbot

Diagram blok Chatbot NLP diatas secara garis besar menggambarkan alur kerja dari sebuah Chatbot NLP dengan uraian sebagai berikut.

1. *Input* dari Pengguna

   Bagian ini merepresentasikan titik awal di mana pengguna memberikan input atau pertanyaan kepada chatbot.

2. Proses NLP

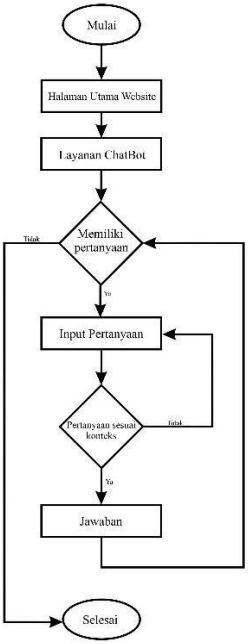
   Proses ini berfungsi agar sistem chatbot dapat memahami input dari pengguna dalam bentuk bahasa alami dan mengubahnya menjadi format yang dapat diproses oleh komputer. Tahapan yang ada di dalam NLP melibatkan beberapa tahap yaitu *Case folding*, tokenisasi, *lemmatization*, Tf-Idf.

3. Prediksi Intent dengan ANN (*Artificial Neural Network*)

   Setelah melalui proses NLP, hasil olahan data akan diberikan kepada jaringan saraf tiruan (ANN). ANN akan menganalisis data tersebut dan mencoba memprediksi "intent" atau maksud dari pengguna. Intent adalah tujuan atau maksud utama dari pesan pengguna. Misalnya, apakah pengguna ingin mengajukan pertanyaan atau mendapatkan informasi tertentu.

4. Output

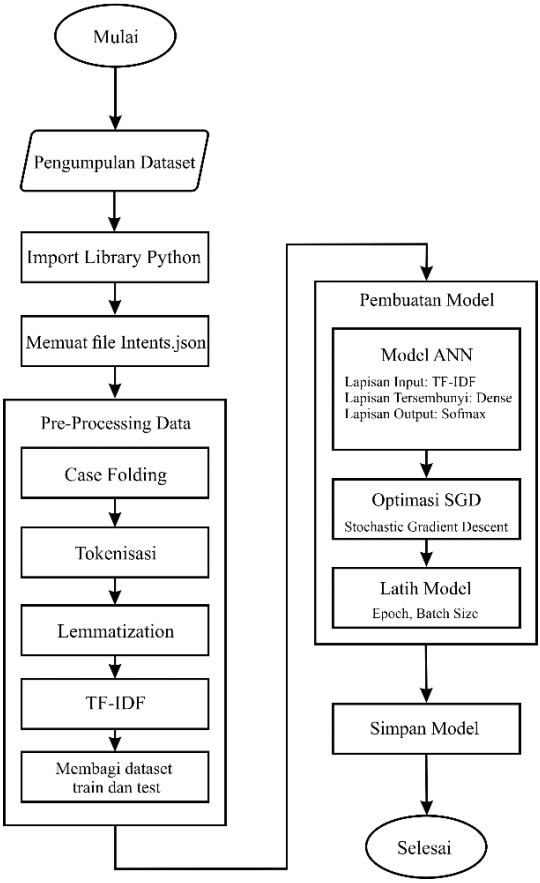
   Bagian ini adalah titik akhir di mana chatbot memberikan respons kepada pengguna berdasarkan prediksi intent.



Gambar 3.2 Flowchart Chatbot

Pada gambar flowchart diatas menggambarkan alur kerja dari sebuah chatbot pada website. Ketika pengguna mengakses halaman utama website, Pengguna dapat terhubung dengan layanan chatbot. Jika pengguna memiliki pertanyaan, mereka dapat mengajukan pertanyaannya. Ketika pertanyaan yang diajukan oleh pengguna sesuai dengan konteks penerimaan mahasiswa baru maka chatbot akan memproses pertanyaan tersebut dan memberikan respon terhadap pertanyaan yang diajukan pengguna. Apabila tidak ada pertanyaan lagi, maka interaksi dengan chatbot akan selesai. *Flowchart* ini menunjukkan bagaimana chatbot berinteraksi dengan pengguna secara berulang hingga pengguna tidak memiliki pertanyaan lagi.

3.2.1.2 Perancangan NLP



Gambar 3.3 Diagram Alur NLP

Gambar diagram alur ini menggambarkan proses dalam membangun sebuah model pemrosesan bahasa alami NLP. Proses dimulai dari pengumpulan data yang akan digunakan untuk melatih model. Tahap selanjutnya adalah mengimpor *library* python yang akan digunakan, setelah

library python diimpor dataset yang telah disatukan ke dalam file bernama intent yang memiliki format JSON dimuat. Dataset ini kemudian melalui tahap pra-pemrosesan, yang meliputi *Case Folding* (konversi huruf kapital menjadi huruf kecil), tokenisasi (pemecahan teks menjadi bentuk-bentuk kecil), *lemmatization* (mengubah suatu kata ke dalam bentuk dasar), proses TF-IDF memiliki fungsi untuk mengubah kata-kata ke dalam bentuk vektor numerik. Metode ini menentukan nilai bobot untuk setiap kata-kata berdasarkan frekuensi kemunculannya dalam sebuah dokumen tunggal dan juga di seluruh kumpulan dokumen, sehingga kata yang cukup penting memiliki bobot nilai yang lebih tinggi. Setelah itu lanjut ke tahap membagi dataset *train* dan *test*. Dataset dibagi menjadi set pelatihan (train set) untuk mengajarkan model, dan set pengujian (test set) untuk menilai keakuratan kinerja model saat berhadapan dengan data baru yang belum pernah dilatih Tahap selanjutnya setelah *pre-processing* adalah membuat model, Dimana model yang akan digunakan dalam perancangan NLP ini adalah *Artificial Neural Network* (ANN) yang dibuat menggunakan *library* tensorflow dan keras serta akan dilatih menggunakan dataset yang sudah melalui tahap *pre-processing*. Setelah model yang dibuat telah melalui tahap pelatihan, Model akan di optimasi menggunakan algoritma *Stochastic Gradient Descent* (SGD). Setelah tahapan optimasi sudah dilalui oleh model yang telah dibuat, Maka tahapan selanjutnya adalah menyimpan model yang telah dibuat dan di latih. Model yang telah dilatih dan dioptimasi akan di simpan ke dalam file. Model ini dapat digunakan kembali tanpa perlu dilatih lagi mulai awal.

1. Pengumpulan Dataset

   Data yang digunakan dalam penelitian ini diperoleh dari wawancara dengan petugas administrasi pendaftaran mahasiswa baru. Data ini sangat penting karena akan menjadi dasar untuk melatih model NLP yang nantinya diharapkan dapat memberikan jawaban yang akurat terhadap pertanyaan calon mahasiswa. Dengan kata lain, model akan melakukan proses belajar dari data hasil wawancara ini untuk dapat memahami konteks pertanyaan yang sering diajukan dan memberikan respons yang sesuai.

2. *Case Folding*

   Tahap case folding melibatkan pengubahan semua kata dalam teks ke format huruf kecil, seperti 'Siang' menjadi 'siang'. Ini dilakukan terutama untuk memudahkan penolahan data, sehingga model NLP dapat lebih mudah dalam menganalisa pola dan memproses data. Dengan cara mengubah semua kata menjadi huruf kecil, menghilangkan variasi huruf besar kecil yang sebenarnya tidak memiliki perbedaan makna dalam arti sesungguhnya. Hal ini dapat membantu model untuk lebih fokus terhadap makna dari kata itu sendiri. *Case folding* dapat dikembangkan menggunakan *library* dalam bahasa pemrograman Python. Library yang populer untuk dapat melakukan tugas-tugas NLP adalah NLTK (Natural Language Toolkit). *Library* ini menyediakan fungsi dan tools yang memudahkan dalam melakukan *preprocessing* teks.

3. Tokenisasi

   Proses tokenisasi melibatkan pembagian kalimat menjadi elemen-elemen paling dasar, yaitu token, yang bisa berupa kata-kata atau frasa tertentu atau bahkan karakter tunggal. Sebagai ilustrasi, dalam kalimat "Saya ingin bertanya tentang pendaftaran" akan diurai menjadi bentuk token: "Saya", "ingin", "bertanya", "tentang", "pendaftaran". Dalam proses tokenisasi library yang digunakan untuk mengimplemetasikannya adalah *Natural Language Toolkit* (NLTK) yang berasal dari bahasa pemrograman Python.

4. *Lemmatization*

   Lemmatization adalah proses mengembalikan setiap kata ke bentuk dasarnya agar dapat mengkategorikan kata yang berbeda dengan makna yang serupa. Proses ini bertujuan untuk mengembalikan kata-kata dengan mengubahnya menjadi bentuk dasar. Lemma adalah bentuk dasar dari sebuah kata tanpa imbuhan

seperti awalan atau akhiran. Misalnya, kata "daftar" akan tetap menjadi "daftar", tetapi kata "pendaftaran" akan diubah menjadi "daftar". Untuk dapat melakukan *lemmatization* dalam tahap *preprocessing*, Salah satu *library* yang cukup populer dan dapat digunakan adalah NLTK (Natural Language Toolkit).

5. *Term Frequency* (TF) - *Inverse Document Frequency* (IDF)

   TF-IDF merupakan metode yang dapat digunakan untuk dapat mengubah mengubah input tekstual menjadi vektor yang terdiri dari angka. Tujuan TF-IDF adalah untuk menentukan seberapa signifikan sebuah kata dalam suatu berkas. Metode ini mengalokasikan nilai bobot yang lebih tinggi kepada istilah-istilah yang khas dan krusial dalam dokumen tersebut, sedangkan istilah-istilah yang umum dan sering ditemukan di banyak dokumen (seperti "dan", "yang", "itu") akan diberi nilai bobot rendah. TF-IDF terdiri dari dua komponen utama yaitu:

   a. *Term Frequency* (TF) berfungsi untuk menentukan frekuensi kemunculan suatu kata dalam sebuah dokumen. Semakin sering kata tersebut muncul, semakin besar juga nilai bobot yang dihasilkannya.

   b. IDF (Inverse Document Frequency) berfungsi menentukan seberapa tidak umumnya sebuah kata dalam suatu kumpulan dokumen. Kata-kata yang jarang muncul di seluruh dokumen akan mendapatkan nilai yang lebih tinggi. Sementara itu, kata-kata yang sering muncul di banyak dokumen cenderung dianggap kurang informatif dan diberi bobot yang lebih kecil.

      Penghitungan nilai akhir TF-IDF dilakukan dengan mengalikan bobot nilai dari TF dan IDF. Hasilnya akan berupa vektor numerik di mana setiap nilai bobot merepresentasikan seberapa "penting" kata di dalam dokumen tersebut.

6. Pembuatan Model

Model yang akan digunakan pada penelitian ini adalah model *Artifical Neural Network* (ANN) yang digunakan untuk mengklasifikasikan teks, Model ini dikembangkan bahasa pemrograman Python dan *library* Keras yang merupakan bagian dari *framework Tensorflow.* Dalam model ini setiap baris menciptakan lapisan dengan jumlah neuron tertentu yang saling terhubung. Terdapat Fungsi aktivasi *Rectified Linear Unit* (RELU) untuk membantu model belajar pola non-linear dalam data serta *lapisan Dropout* untuk mencegah model menghafal data latih terlalu kuat dan membuatnya lebih fleksibel terhadap data baru.

Fungsi aktivasi yang ada di dalam model ANN ini berfungsi untuk memungkinkan model belajar pola non-linear dalam data, Menggunakan fungsi aktivasi *Rectified Linear Unit* (ReLU) pada sebagian besar neuron. Fungsi ReLU mendefinisikan output neuron sebagai nilai maksimum antara 0 dan nilai inputnya. Fungsi Dropout di dalam model ANN adalah untuk mencegah model menghafal data latih terlalu kuat (overfitting) dan membuatnya lebih generalisasi terhadap data baru, Dengan cara menerapkan teknik *dropout. Dropout* secara acak menonaktifkan sebagian neuron pada setiap iterasi pelatihan. Hal ini memaksa model untuk belajar fitur-fitur yang lebih dalam dan dapat mengurangi risiko *overfitting.*

Proses pelatihan model ANN melibatkan penyesuaian bobot secara berulang-ulang untuk meminimalkan kesalahan prediksi pada data latih. Proses ini biasanya dilakukan menggunakan algoritma optimasi seperti *Stochastic Gradient Descent* (SGD).

7. Optimasi Model

Untuk dapat mencapai kinerja yang optimal, Model ANN yang telah dibangun perlu melalui proses optimasi. Optimasi adalah proses penyesuaian parameter model (bobot dan bias) secara berulang

untuk meminimalkan fungsi *loss*. Fungsi *loss* berfungsi untuk mengukur seberapa besar kesalahan prediksi model terhadap data sebenarnya. Dengan menggunakan optimasi *Stochastic Gradient Descent* (SGD) akan digunakan, khususnya varian dengan parameter *Nesterov*, SGD adalah algoritma optimasi yang cukup populer dalam pembelajaran mesin. Prinsip kerjanya adalah dengan memperbarui parameter model secara bertahap berdasarkan gradien dari fungsi *loss* terhadap parameter tersebut. Gradien menunjukkan arah peningkatan tercepat dari fungsi loss, sehingga dengan bergerak ke arah yang berlawanan dari gradien, Hal ini dapat meminimalkan fungsi *loss*. Karena algoritma ini secara berulang-ulag memperbarui bobot dan kecenderungan pada setiap lapisan untuk dapat meminimalkan fungsi *loss*. Dalam penelitian ini, Penulis menggunakan algoritma *Stochastic Gradient Descent* (SGD) dengan parameter *Nesterov* sebagai *optimizer*.

8. *Deployment*

Setelah model *Artificial Neural Network* (ANN) selesai dilatih dan dioptimasi menggunakan algoritma *Stochastic Gradient Descent* (SGD), langkah krusial selanjutnya adalah Simpan Model ke dalam sebuah file. Proses ini sangat penting karena memungkinkan arsitektur model dan bobot-bobot yang telah dipelajari selama pelatihan untuk disimpan secara permanen di penyimpanan lokal (seperti hard drive atau Google Drive).

Pada penelitian ini, model disimpan dalam format *.keras*. File ini berisi semua informasi yang dibutuhkan, mulai dari struktur jaringan (jumlah lapisan dan neuron) hingga bobot-bobot koneksi antar neuron yang telah disesuaikan melalui proses pelatihan. Dengan menyimpan model, kita dapat memuatnya kembali kapan saja untuk digunakan, tanpa perlu mengulang seluruh proses pelatihan yang memakan waktu dan sumber daya komputasi.

3.2.2 Perancangan Data

Perancangan Data adalah proses mengumpulkan Untuk mengidentifikasi pertanyaan umum dari calon mahasiswa kepada petugas pendaftaran, penelitian ini menggunakan data yang bersumber dari hasil wawancara dengan staf administrasi pendaftaran mahasiswa baru terkait pertanyaan yang sering diajukan oleh calon mahasiswa serta jawaban dari pertanyaan yang diajukan. Hasil dari wawancara tersebut di olah lalu dikembangkan lagi dan akan disimpan dalam bentuk file yang berformat JSON yang didalamnya memiliki struktur data yang terdiri dari tiga komponen yaitu tag, *patterns*, dan *responses*, yang masing-masing memiliki fungsi penting dalam proses klasifikasi dan respon chatbot. Fungsi tag digunakan sebagai penanda atau label utama yang mewakili maksud (intent) dari sebuah pernyataan dari pengguna. Setiap intent memiliki satu tag dengan nama berbeda yang digunakan oleh model agar dapat mengenali kategori percakapan, seperti "sapaan", "daftar", atau "selesai". Berikutnya fungsi *patterns* merupakan beberapa contoh kalimat pertanyaan atau frasa yang biasa digunakan oleh pengguna untuk menyampaikan maksud tertentu. *Patterns* ini yang menjadi data latih bagi model agar dipelajari variasi kalimat dengan beberapa pertanyaan dengan penyampaian yang berbeda, sehingga model dapat mengklasifikasikan input secara akurat meskipun bentuk kalimatnya berbeda-beda. Berikutnya fungsi *responses* adalah daftar tanggapan yang akan diberikan oleh chatbot ketika berhasil mengenali sebuah intent atau kalimat yang di input oleh pengguna. Agar mendapatkan urutan terkait pertanyaan yang sering diajukan serta dapat digunakan untuk melatih model yang akan digunakan.

3.2.3 Perancangan *User Interface*

Pada tahap ini, *User* akan langsung dihadapkan tampilan sistem chatbot. Pada tampilan ini terdapat pesan selamat datang dari sistem yang mengundang pengguna untuk mengajukan pertanyaan. Sistem pun memberikan respons berupa panduan yang akan ditampilkan secara lengkap. Terdapat juga kotak teks untuk pengguna mengetikkan pertanyaan dan tombol "Kirim" untuk mengirimkan pesan.

Gambar 3.4 Desain User Interface

3.3 Rancangan Pengujian

Pengujian sistem dirancang untuk memastikan bahwa sistem yang telah dibangun dapat beroperasi sebagaimana mestinya. dan memenuhi spesifikasi yang telah ditentukan. Penelitian ini menguji kinerja model menggunakan metode pengujian uji akurasi dan *Confusion Matrix*. *Confusion Matrix* adalah metode pengujian yang berfungsi untuk dapat menilai seberapa baik model klasifikasi bekerja, termasuk model chatbot yang telah dibangun, Hal ini dilakukan dengan membandingkan hasil prediksi model dengan label sesungguhnya dari data uji dalam bentuk tabel.

3.3.1 *Confusion Matrix*

Confusion Matrix menyajikan representasi visual yang jelas mengenai efektivitas model kita dalam mengategorikan teks input secara akurat. Pada tahap pengujian menggunakan *confusion matrix*, langkah pertama adalah melakukan prediksi terhadap sejumlah data uji, di mana data ini berisi sampel input beserta label kelas (intent) yang benar. Setiap input akan diproses oleh model untuk dapat menghasilkan prediksi intent,

yang kemudian akan dibandingkan dengan label yang sebenarnya. Hasil prediksi dan label sebenarnya ini diolah menjadi *confusion matrix*, yaitu sebuah tabel yang menunjukkan seberapa sering model berhasil memprediksi intent dengan benar (benar positif/benar negatif) dan seberapa sering model salah memprediksi (salah positif/salah negatif) untuk setiap kelas.

Confusion matrix memberikan informasi detail tentang performa model dalam bentuk empat nilai utama yaitu *True Positive* (TP) adalah prediksi benar yang terkonfirmasi. *False Positive* (FP) merupakan prediksi salah yang tidak seharusnya ada, dan *False Negative* (FN) adalah data yang gagal diprediksi dengan tepat padahal seharusnya ada, dan *true negative* (TN) untuk prediksi salah yang terhindar.

3.3.2 *Classification Report*

Classification report merupakan alat evaluasi penting dalam pengujian model klasifikasi, yang memberi pemahaman menyeluruh terkait kinerja model dalam membedakan masing-masing kelas (intent). Laporan ini menyajikan analisis hasil prediksi model pada data uji. Di dalamnya, terdapat beberapa metrik penting seperti *presisi, recall, f1-score*, dan support yang disajikan untuk tiap kategori secara terpisah, sehingga memungkinkan evaluasi performa model yang lebih mendalam dan terperinci dibandingkan hanya menggunakan satu metrik global.

Pada tahap pengujian, data uji yang terdiri dari input dan label kelas yang benar diproses oleh model untuk menghasilkan prediksi. Perbandingan antara hasil prediksi dan label aktual dilakukan, dan dari hasil perbandingan ini, *classification report* disusun untuk setiap kelas. Metrik *Precision* menunjukkan ketepatan model saat mengidentifikasi sebuah kelas, yaitu jumlah prediksi yang akurat dari keseluruhan prediksi yang dibuat untuk kelas yang dimaksud. *Recall* mengukur seberapa baik model dalam menemukan semua contoh dari suatu kelas, yaitu berapa banyak data sebenarnya dari kelas tersebut yang berhasil diprediksi dengan benar.