

Implementasi Chatbot Layanan Pelanggan Berbasis NLP Menggunakan Platform Telegram Pada Penyedia Layanan Internet

Ivan Yusuf Sholy Khudin¹, Ardi Sanjaya², Ahmad Bagus Setiawan³

^{1,2}Teknik Informatika, Fakultas Teknik dan Ilmu Komputer, Universitas Nusantara PGRI Kediri

E-mail: ¹ivanyusuf300@gmail.com, ²dersky@gmail.com, ³ahmadbagus@unpkediri.ac.id

Abstrak – Dalam industri jasa telekomunikasi, kecepatan respon terhadap keluhan pelanggan merupakan indikator vital kualitas layanan (*Quality of Service*). Namun, penanganan keluhan secara manual pada salah satu Internet Service Provider (ISP) di wilayah Kediri seringkali menyebabkan penumpukan antrean pesan (*bottleneck*) dan keterlambatan penanganan gangguan. Penelitian ini menawarkan solusi untuk mengatasi permasalahan tersebut dengan mengembangkan sistem Chatbot cerdas berbasis Natural Language Processing (NLP) yang diintegrasikan dengan platform Telegram. Metode pengembangan sistem dilakukan menggunakan model Waterfall, mulai dari analisis kebutuhan, perancangan pipeline NLP, hingga implementasi kode program menggunakan bahasa Python. Algoritme NLP diterapkan untuk mengklasifikasikan intent (maksud) dari pesan pelanggan yang bervariasi secara otomatis. Hasil pengujian menunjukkan bahwa integrasi chatbot pada Telegram mampu mengenali pola keluhan teknis dan administratif dengan tingkat akurasi yang baik, serta memberikan respon secara real-time. Implementasi sistem ini terbukti secara signifikan meningkatkan efisiensi operasional dukungan teknis dan mengurangi waktu tunggu pelanggan.

Kata Kunci — Chatbot, Internet Service Provider, Natural Language Processing, Python, Telegram.

1. PENDAHULUAN

Penyedia layanan internet (*Internet Service Provider / ISP*) memiliki kewajiban untuk memastikan ketersediaan layanan dan dukungan teknis yang optimal bagi pelanggan sesuai dengan regulasi telekomunikasi yang berlaku. Namun, kendala klasik yang sering dihadapi oleh ISP skala menengah adalah keterbatasan sumber daya manusia (*Customer Service*) dalam menangani lonjakan trafik komplain, terutama saat terjadi gangguan massal (*mass outage*). Berdasarkan observasi pada objek penelitian, proses pemilahan keluhan antara masalah teknis (seperti koneksi lambat, indikator LOS merah) dan masalah administratif masih dilakukan secara manual, yang berpotensi menimbulkan *human error* dan waktu tunggu yang lama bagi pelanggan.

Untuk mengatasi inefisiensi tersebut, pemanfaatan teknologi otomatisasi berupa *Chatbot* telah banyak diteliti di berbagai domain. Pada penelitian terdahulu, telah dikembangkan *chatbot* untuk pelayanan mahasiswa baru yang terbukti efisien dalam memberikan informasi standar [1] [2]. Di sektor pendidikan menengah, implementasi NLP pada *chatbot* juga telah dilakukan dan terbukti mampu meningkatkan akses informasi siswa secara mandiri [3].

Dalam konteks teknologi antarmuka, pemilihan platform Telegram didasarkan pada fitur open API yang handal, ringan, dan mendukung komunikasi dua arah secara *real-time*. Keunggulan ini dikonfirmasi oleh studi lain yang menunjukkan bahwa integrasi NLP pada Telegram mampu memproses bahasa alami dengan akurasi yang baik [4]. Hal ini sejalan dengan temuan yang memanfaatkan Telegram untuk prediksi risiko kesehatan, membuktikan fleksibilitas platform ini untuk sistem cerdas [5].

Di lingkungan Universitas Nusantara PGRI Kediri, pengembangan metode pemrosesan teks juga menjadi fokus penelitian. Implementasi *Natural Language Interface to Database* (NLIDB) pada *chatbot* perusahaan media telah berhasil dilakukan [6]. Selain itu, pengembangan metode pengukuran kemiripan makna menggunakan *Cosine Similarity* juga telah dilakukan dan menjadi dasar penting dalam pencocokan pola teks [7].

Meskipun penelitian terdahulu menunjukkan efektivitas NLP pada layanan publik [8], penelitian ini mengembangkan konsep tersebut secara spesifik pada domain jaringan komputer (ISP). Sistem dilatih untuk mengenali istilah-istilah teknis jaringan (*network jargon*) yang sering dikeluhkan pelanggan lokal. Tujuan akhirnya adalah menciptakan sistem asisten virtual yang mampu mengurangi beban kerja admin dan mempercepat *Mean Time to Respond* (MTTR) secara signifikan.

2. METODE PENELITIAN

Penelitian ini menerapkan metode pengembangan sistem menggunakan model *Waterfall*, yang dipilih karena memiliki tahapan yang sistematis dan sesuai untuk pengembangan sistem berbasis kebutuhan yang relatif stabil. Tahapan penelitian meliputi analisis kebutuhan, perancangan sistem, implementasi metode *Natural Language Processing* (NLP), pengujian sistem, serta tahap pemeliharaan.

2.1 Tahapan Pengembangan Sistem (*Model Waterfall*)

Model *Waterfall* digunakan untuk memastikan setiap tahap pengembangan sistem dilakukan secara terstruktur dan terdokumentasi dengan baik. Pemetaan tahapan *Waterfall* pada penelitian ini ditunjukkan sebagai berikut:

2.1.1 Analisis Kebutuhan

Tahap ini dilakukan melalui observasi log percakapan pelanggan dan wawancara dengan tim *Network Operations Center* (NOC) untuk mengidentifikasi jenis keluhan pelanggan, kebutuhan fungsional *chatbot*, serta batasan sistem.

2.1.2 Perancangan Sistem

Pada tahap ini dilakukan perancangan arsitektur *chatbot*, *use case diagram*, serta alur komunikasi antara pelanggan, *chatbot*, *Customer Service* (CS), dan NOC.

2.1.3 Implementasi Sistem

Tahap implementasi meliputi pembangunan *pipeline* NLP, pembuatan *knowledge base*, integrasi dengan Telegram Bot API, serta pengembangan logika klasifikasi *intent*.

2.1.4 Pengujian Sistem

Pengujian dilakukan menggunakan metode Black Box Testing untuk mengevaluasi kesesuaian antara input pengguna dan respon sistem, serta mengukur performa menggunakan *Confusion Matrix*.

2.1.5 Pemeliharaan (*Maintenance*)

Tahap ini mencakup pembaruan *dataset*, penyesuaian *threshold similarity*, serta penambahan *intent* baru berdasarkan pola keluhan pelanggan yang berkembang.

2.2 Tahapan Pengumpulan Data

Basis pengetahuan (*knowledge base*) *chatbot* disusun menggunakan pendekatan *hybrid data acquisition*, yaitu kombinasi antara data empiris lapangan dan validasi oleh pakar teknis jaringan. Pendekatan ini dipilih untuk memastikan bahwa data yang digunakan tidak hanya representatif secara statistik, tetapi juga relevan secara operasional.

2.2.1 Observasi Log Historis

Data diperoleh dari hasil ekstraksi (*dump*) riwayat percakapan antara pelanggan dan petugas *Customer Service* melalui platform *WhatsApp* selama periode tiga bulan. Sebanyak 650 baris data percakapan berhasil dikumpulkan dan digunakan sebagai *dataset* awal. Sebelum digunakan, data mentah terlebih dahulu melalui proses pembersihan untuk menghilangkan informasi sensitif (*Personally Identifiable Information/PII*), seperti nomor telepon dan nama pelanggan.

2.2.2 Wawancara Pakar (*Expert Judgment*)

Validasi dilakukan melalui wawancara mendalam dengan *Divisi Network Operations Center* (NOC). Tahap ini bertujuan untuk menguji ketepatan klasifikasi keluhan serta memastikan sistem mampu membedakan nuansa gangguan teknis yang memiliki kemiripan istilah. Contohnya, membedakan kata kunci “lemot” yang umumnya mengindikasikan *bandwidth contention* dengan istilah “RTO” yang lebih mengarah pada gangguan fisik jaringan atau *packet loss*.

Data hasil observasi kemudian diklasifikasikan dan diberi label (*labelling*) untuk keperluan penyusunan basis pengetahuan dan penentuan kategori *intent*. *Dataset* ini mencakup teks pertanyaan asli dari pelanggan

beserta fitur kata kunci dan kategori *intent* yang relevan. Sampel *dataset* yang digunakan dapat dilihat pada Tabel 1.

Tabel 1. Sampel *Dataset Knowledge Based Chatbot ISP*

No	Teks Pertanyaan (Input)	Fitur Kata Kunci	Label <i>Intent</i> (Output)
1	"Internet mati total sejak sore"	Gangguan	Gangguan Jaringan
2	"Bagaimana cara <i>downgrade</i> paket"	Paket	Info Paket
3	"Bagaimana cara cek <i>invoice</i> "	Tagihan	Cek Tagihan
4	"Modem tidak konek ke internet"	Gangguan	Gangguan Jaringan
5	"Saya butuh bantuan lainnya"	-	Lainnya

2.3 Proses Pelabelan Data

Data hasil observasi kemudian diklasifikasikan dan diberi label (*labelling*) secara manual berdasarkan kategori *intent* yang telah ditentukan. Proses pelabelan dilakukan dengan melibatkan pakar teknis jaringan dari tim NOC untuk meminimalkan bias interpretasi.

Setiap data percakapan diberi label *intent* sesuai dengan maksud utama pertanyaan pelanggan, seperti Gangguan Jaringan, Info Paket, Cek Tagihan, dan Lainnya. Hasil *labelling* ini digunakan sebagai dasar pembentukan *knowledge base chatbot*.

2.4 Pembagian *Dataset*

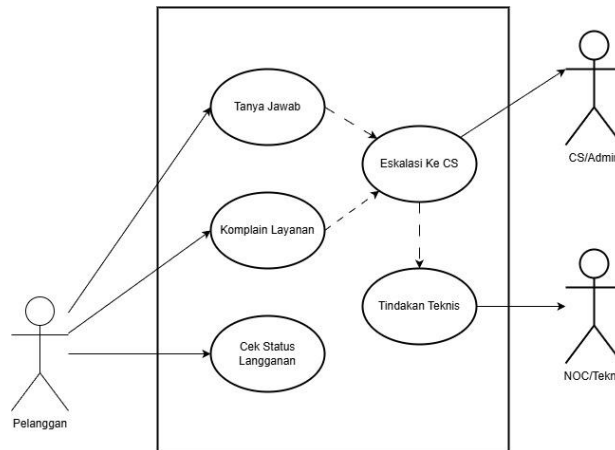
Dataset sebanyak 650 data digunakan sebagai basis pengetahuan (*knowledge base*) untuk menyusun pola pertanyaan dan membangun representasi vektor *intent* menggunakan metode TF-IDF. *Dataset* ini berfungsi sebagai referensi kemiripan (*similarity reference*) dalam proses pencocokan vektor dan klasifikasi *intent*.

Untuk keperluan evaluasi sistem, digunakan 50 data uji yang tidak termasuk dalam basis pengetahuan. Data uji ini digunakan untuk mengukur performa sistem dalam mengklasifikasikan *intent* pengguna dan menghasilkan respon yang sesuai.

Pemisahan antara basis pengetahuan dan data uji dilakukan untuk memastikan bahwa proses evaluasi sistem bersifat objektif dan mencerminkan kondisi penggunaan nyata, tanpa melibatkan data referensi yang sama pada tahap pengujian.

2.5 Arsitektur Sistem

Tahap perancangan sistem meliputi identifikasi komponen *chatbot* yang akan beroperasi pada platform Telegram. Sistem didesain untuk menjembatani komunikasi antara pelanggan ISP dengan server pengolah bahasa alami. Pengguna *Chatbot* Pengguna (Pelanggan) berinteraksi langsung dengan bot melalui aplikasi Telegram. Profil interaksi yang disediakan meliputi Laporan Gangguan, Cek Info Tagihan, Panduan *Restart* Modem, dan Info Paket Baru. Pada gambar 1 dibawah, disajikan *use case diagram* yang memberikan gambaran interaksi antara pelanggan, *Customer Service* (CS) dan *Network Engineer* (NOC). Pelanggan bertindak sebagai pengguna akhir yang mengakses fitur *self-service* pada chatbot, seperti melaporkan gangguan koneksi, mengecek status tagihan, dan meminta panduan *restart* perangkat. CS Bertugas memantau keluhan yang bersifat administratif dan mengambil alih percakapan (*human handover*) jika *chatbot* tidak dapat menjawab pertanyaan non-teknis. NOC memiliki hak akses khusus untuk memperbarui *dataset* jawaban teknis, memantau tiket gangguan kritis yang dieskalasi oleh bot, dan melakukan *maintenance* pada logika NLP sistem. Gambar 1 berikut menunjukkan integrasi alur kerja di mana *chatbot* bertindak sebagai filter L1 (*Level 1 Support*) sebelum masalah diteruskan ke CS atau NOC.



Gambar 1. Use Case Diagram

2.6 Implementasi

2.6.1 Model Pemrosesan Bahasa Alami (Natural Language Processing - NLP)

Sistem dikembangkan menggunakan pendekatan NLP untuk memproses input teks pelanggan menjadi instruksi terstruktur yang dapat dieksekusi mesin [9]. Agar sistem dapat menentukan jawaban yang tepat dan mengklasifikasikan intent (maksud) pelanggan, alur pemrosesan data dilakukan melalui tahapan berikut:

1. Pemrosesan Teks (Text Preprocessing)

Bertujuan menyederhanakan input agar mudah dianalisis. Tahap awal untuk menstandarisasi input mentah agar beban komputasi lebih ringan. Proses ini meliputi:

- Case Folding & Cleaning*: Mengubah seluruh huruf menjadi kecil (*lowercase*) dan menghapus karakter *noise* (seperti tanda baca dan angka yang tidak relevan).
- Tokenizing*: Memecah kalimat menjadi potongan kata tunggal (token).
- Filtering*: Membuang kata hubung (*stopword*) yang tidak memiliki makna teknis (seperti "yang", "dan", "di").
- Stemming*: Mengembalikan kata berimbuhan ke bentuk dasar menggunakan algoritma Sastrawi (contoh: "mengeluh" menjadi "eluh"), sehingga variasi kata dapat dikenali sebagai entitas yang sama.

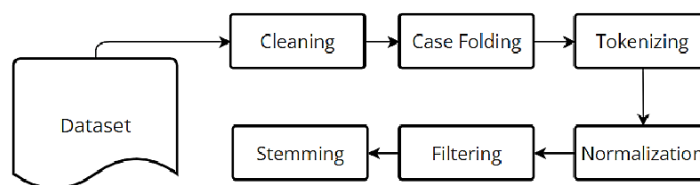
2. Pembobotan Kata (TF-IDF)

Mesin tidak dapat memproses teks secara langsung. Oleh karena itu, hasil *preprocessing* diubah menjadi representasi numerik (vektor) menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF). Metode ini memberikan bobot tinggi pada kata kunci unik (seperti "LOS", "merah", "putus") dan bobot rendah pada kata umum. Hasilnya adalah sebuah vektor yang merepresentasikan "sidik jari" dari pertanyaan pelanggan.

3. Pencocokan Pola (Pattern Matching):

Sistem menggunakan metode perhitungan vektor untuk mengukur tingkat kemiripan (*similarity*) antara pertanyaan pelanggan dengan *intent* yang ada di *database*, memilih respon dengan skor relevansi tertinggi.

Visualisasi dari alur *text preprocessing* NLP ditampilkan pada Gambar 2



Gambar 2. Text Preprocessing

2.6.2 Formulasi Matematis Model NLP

Sistem *chatbot* tidak dibangun menggunakan aturan kondisional sederhana (*rule-based if-else*), melainkan menerapkan pendekatan *Vector Space Model* (VSM) untuk merepresentasikan teks

dalam bentuk numerik. Pendekatan ini memungkinkan sistem melakukan perhitungan kemiripan secara matematis antara pesan pengguna dan basis data *intent*.

1. Pembobotan TF-IDF (*Term Frequency – Inverse Document Frequency*)

Pembobotan TF-IDF digunakan untuk mengukur tingkat kepentingan suatu kata dalam sebuah dokumen terhadap keseluruhan korpus data.

a. *Term Frequency* (TF)

TF mengukur seberapa sering sebuah term t muncul dalam suatu dokumen d :

$$TF(t,d) = \frac{\text{jumlah kemunculan } t \text{ dalam } d}{\text{total kata dalam } d} \quad (1)$$

b. *Inverse Document Frequency* (IDF)

IDF mengukur tingkat kelangkaan term dalam seluruh dokumen:

$$IDF(t) = \log \left(\frac{N}{df(t)} \right) \quad (2)$$

di mana N adalah jumlah total dokumen dan $df(t)$ adalah jumlah dokumen yang mengandung term t .

c. Bobot TF-IDF

$$TF-IDF(t,d) = TF(t,d) \times IDF(t) \quad (3)$$

Pembobotan ini menyebabkan kata umum seperti “yang” atau “dan” memiliki bobot rendah, sementara kata teknis seperti “LOS” atau “RTO” memiliki bobot lebih tinggi.

2. Pengukuran Kemiripan (*Vektor Matching*)

Setelah pesan pengguna dan *intent* direpresentasikan sebagai vektor, tingkat kemiripan dihitung menggunakan *Cosine Similarity*:

$$\cos(\theta) = \frac{A \cdot B}{||A|| \times ||B||} \quad (4)$$

Nilai *cosine* berada pada rentang 0–1, di mana nilai mendekati 1 menunjukkan tingkat kemiripan yang tinggi, sedangkan nilai mendekati 0 menunjukkan tidak adanya keterkaitan makna.

2.6.3 Struktur *Data Knowledge Base*

Untuk menjaga fleksibilitas sistem, basis pengetahuan disimpan dalam format *JSON* (*JavaScript Object Notation*). Struktur ini memungkinkan admin NOC untuk memperbarui pola pertanyaan dan respon tanpa harus memodifikasi *source code* utama. Cuplikan struktur *file intents.json* ditunjukkan pada Listing 1:

```
{
  "intents": [
    {
      "tag": "gangguan_los",
      "patterns": [
        "lampu modem merah",
        "pon kedip merah",
        "los nyala terus",
        "internet mati lampu merah"
      ],
      "responses": [
        "Indikator LOS Merah menandakan kabel fiber optik terputus atau redaman tinggi. Mohon periksa apakah kabel patch cord tertekuk atau terlepas."
      ],
      "context_set": "tiket_gangguan"
    },
  ],
}
```

```
{
  "tag": "cek_tagihan",
  "patterns": [
    "tagihan bulan ini berapa",
    "cek invoice",
    "bayar wifi berapa"
  ],
  "responses": [
    "Untuk mengecek tagihan, silakan balas dengan format:
    INFO <spasi> ID_PELANGGAN."
  ]
}
```

2.7 Implementasi Sistem

Sistem *chatbot* dibangun menggunakan bahasa pemrograman *Python* karena dukungan pustaka NLP yang lengkap dan stabil. Implementasi teknis melibatkan penggunaan *Telegram Bot API* sebagai media komunikasi antara pengguna dan server.

Kode program dikembangkan untuk menghubungkan token bot Telegram dengan logika pemrosesan NLP. Selain itu, dilakukan konfigurasi server agar *chatbot* dapat berjalan secara kontinu dan melayani permintaan pelanggan selama 24 jam secara otomatis (*auto-response*).

2.8 Penetapan *Threshold Similarity*

Untuk meningkatkan akurasi respon dan menghindari kesalahan klasifikasi pada pertanyaan ambigu, sistem menerapkan nilai ambang batas (*threshold*) *cosine similarity* sebesar 0,60. Nilai ini ditentukan berdasarkan hasil uji coba awal pada data latih, di mana nilai di bawah *threshold* cenderung menghasilkan respon yang tidak relevan, sedangkan nilai di atas *threshold* menunjukkan kecocokan makna yang stabil.

2.9 Algoritma Klasifikasi *Intent*

Alur kerja klasifikasi *intent* pada chatbot ini dapat dirumuskan dalam bentuk alur sebagai berikut: Terima input teks dari pengguna -> *preprocessing* teks -> Ubah teks menjadi vektor TF-IDF -> Hitung *cosine similarity* dengan seluruh *intent* -> Ambil nilai *similarity* tertinggi -> Jika nilai $\geq$ *threshold*, tetapkan label *intent* -> Jika nilai < *threshold*, kirim respon *fallback* -> Kirim respon ke pengguna melalui Telegram

2.10 Metode Pengujian dan Evaluasi Sistem

Pengujian sistem dilakukan menggunakan metode *Black Box Testing*, dengan fokus pada kesesuaian antara input pengguna dan respon sistem. Evaluasi performa dilakukan menggunakan *Confusion Matrix* untuk menghitung nilai akurasi, presisi, dan *recall*, yang bertujuan mengukur kemampuan sistem dalam mengklasifikasikan *intent* secara tepat.

3 HASIL DAN PEMBAHASAN

3.1 Pengujian Alur Klasifikasi *Intent*

Sebelum dilakukan pengujian performa secara kuantitatif, dilakukan pengujian alur pemrosesan pesan (*end-to-end testing*) untuk memastikan bahwa sistem *chatbot* mampu memproses pertanyaan pelanggan secara utuh, mulai dari input teks mentah hingga menghasilkan respon yang sesuai. Pengujian ini bertujuan untuk memvalidasi bahwa seluruh tahapan pemrosesan bahasa alami yang telah dirancang pada bab metode dapat berjalan dengan benar dan saling terintegrasi.

Pengujian alur dilakukan dengan mengambil salah satu kasus gangguan umum yang sering terjadi di lapangan, yaitu gangguan indikator LOS Merah. Adapun penelusuran alur pemrosesan pada satu siklus interaksi dijelaskan sebagai berikut :

3.1.1 Input Pengguna:

"Min, tolong dong *modem* saya kedip merah terus nih, ga bisa liat *youtube*."

3.1.2 Tahap *Preprocessing (Case Folding & Filtering)*:

Pada tahap ini, sistem melakukan case folding dan cleaning untuk menghilangkan karakter yang tidak relevan serta menyeragamkan format teks. Selanjutnya dilakukan tokenizing dan stopwords removal.

Sistem membersihkan *stopword* dan simbol. *Output*: ['modem', 'kedip', 'merah', 'youtube']

3.1.3 Transformasi Vektor dan Pencocokan Pola

Token hasil *preprocessing* selanjutnya diubah menjadi vektor numerik menggunakan metode TF-IDF. Vektor input ini kemudian dibandingkan dengan vektor representasi setiap *intent* yang tersimpan dalam basis pengetahuan menggunakan metode *Cosine Similarity*.

Hasil perhitungan *similarity* antara vektor input dan vektor intent adalah sebagai berikut:

- Gangguan LOS Merah → *Similarity* = 0,89
- Gangguan Buffering Youtube → *Similarity* = 0,45
- Ganti Password WiFi → *Similarity* = 0,12

Nilai *similarity* tertinggi diperoleh oleh *intent* Gangguan LOS Merah. Nilai ini menunjukkan tingkat kemiripan makna paling besar antara pertanyaan pelanggan dan pola *intent* tersebut.

3.1.4 Klasifikasi (*Vector Matching*):

Sistem menghitung kemiripan vektor input dengan *knowledge base*:

- Match 1: "Indikator LOS Merah" (Score: 0.89)
- Match 2: "Youtube Buffering" (Score: 0.45)
- Match 3: "Ganti Password" (Score: 0.12)
- Decision: Sistem memilih Match 1 karena memiliki skor tertinggi melampaui *threshold* (0.60).

3.1.5 *Output* Sistem: Sistem mengirimkan solusi terkait pengecekan kabel fiber optik (*patch cord*).

3.2 Pengambilan Keputusan dan Respon Sistem

Sistem menerapkan nilai ambang batas (*threshold*) sebesar 0,60. Karena nilai *similarity* tertinggi (0,89) melebihi *threshold*, maka pertanyaan pelanggan diklasifikasikan sebagai Gangguan Jaringan – LOS Merah. Berdasarkan label ini, sistem mengambil respon yang sesuai dari basis pengetahuan. *Output* Sistem diilustrasikan sebagai berikut yang mana hasil tersebut menunjukkan bahwa sistem mampu melakukan klasifikasi intent dan memberikan respon yang relevan secara otomatis.

("Maaf Kak, Indikator LOS Merah menandakan gangguan pada jalur fiber optik. Silakan periksa kondisi kabel *patch cord* atau hubungi teknisi.")

3.3 Hasil Pengujian Sampel

Setelah alur klasifikasi tervalidasi secara fungsional, pengujian dilakukan terhadap beberapa variasi input pesan pelanggan untuk melihat konsistensi respon sistem. Tabel 2 menampilkan sampel hasil pengujian terhadap berbagai jenis pertanyaan.

3.3.1 Hasil Pengujian Sampel (*Sample Testing*)

Tabel 2 memperlihatkan sampel respons sistem terhadap variasi input yang diuji:

No	Input Pesan Pengguna	Kategori Seharusnya	Respon Sistem	Status
1	"Wifi lemot banget min buat main game"	Gangguan Koneksi	Solusi: <i>Refresh Jaringan/Restart</i>	<i>Valid</i>
2	"Kabel putus digigit tikus"	Gangguan Fisik	Solusi: Laporan Teknisi (Tiket)	<i>Valid</i>
3	"Berapa biaya pasang baru?"	Informasi Admin	Info Paket & Harga	<i>Valid</i>
4	"Selamat pagi min"	Sapaan (<i>Greeting</i>)	Sapaan Balik	<i>Valid</i>
5	"Min pinjem duit dong"	<i>Out of Topic</i>	<i>Fallback</i> : "Maaf tidak mengerti"	<i>Valid</i>
6	"Sinyal hilang kena hujan"	<i>Out of Topic</i>	<i>Miss</i> : "Maaf tidak mengerti"	<i>Invalid</i>

Hasil pengujian menunjukkan bahwa sistem mampu membedakan pertanyaan teknis, administratif, dan percakapan di luar konteks dengan cukup baik, meskipun masih ditemukan beberapa kasus kesalahan klasifikasi.

3.4 Evaluasi Performa Sistem Menggunakan *Confusion Matrix*

Pengujian kuantitatif dilakukan menggunakan 50 data uji yang tidak termasuk dalam basis pengetahuan. Setiap hasil klasifikasi sistem dibandingkan dengan label kebenaran (*ground truth*) dan disusun dalam bentuk *Confusion Matrix*.

kontingensi (*Confusion Matrix*) sebagai berikut:

- True Positive* (TP) = 36 (Sistem membalas keluhan teknis dengan benar).
- True Negative* (TN) = 7 (Sistem menolak *chat* sampah/iseng dengan benar).
- False Positive* (FP) = 3 (*Chat* iseng/salah konteks tapi dijawab serius oleh sistem).
- False Negative* (FN) = 4 (Keluhan teknis valid tapi sistem gagal mengenali/menjawab "tidak mengerti").

Berdasarkan data di atas, perhitungan performa sistem adalah:

$$\text{Akurasi} = \frac{TP + TN}{\text{Total Data}} \times 100\% = \frac{36 + 7}{50} \times 100\% = 86\%$$

$$\text{Presisi} = \frac{TP}{TP + FP} \times 100\% = \frac{36}{36 + 3} \times 100\% = 92,3\%$$

$$\text{Recall} = \frac{TP + TN}{TP + FN} \times 100\% = \frac{36}{36 + 4} \times 100\% = 90\%$$

3.5 Pembahasan Hasil Pengujian

Berdasarkan hasil evaluasi, sistem *chatbot* mencatatkan tingkat akurasi sebesar 86%. Nilai ini menunjukkan bahwa sistem memiliki tingkat keandalan yang cukup tinggi untuk digunakan sebagai *First Layer Support (L1 Support)* pada lingkungan ISP.

Nilai presisi yang tinggi (92,3%) mengindikasikan bahwa ketika sistem memberikan jawaban teknis, respon yang dihasilkan cenderung tepat dan relevan. Hal ini penting untuk menghindari kesalahan penanganan gangguan yang bersifat kritis. Nilai *recall* sebesar 90% menunjukkan bahwa sebagian besar keluhan teknis pelanggan berhasil dikenali oleh sistem.

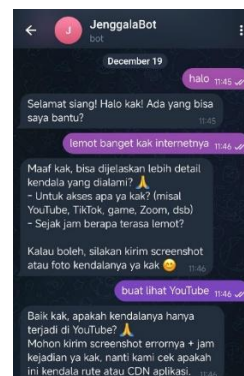
Kesalahan klasifikasi umumnya disebabkan oleh penggunaan istilah non-baku, bahasa slang, atau konteks gangguan yang ambigu, seperti penggunaan kata singkat ("lmt", "ngadat") serta keluhan yang dipengaruhi faktor eksternal.

1.1 Implementasi Antarmuka dan Mekanisme *Slot-Filling*

Selain kemampuan klasifikasi *intent*, sistem juga menerapkan mekanisme *slot-filling* untuk mengekstraksi entitas penting dari pertanyaan pelanggan, seperti jenis aplikasi atau indikator perangkat. Mekanisme ini memungkinkan sistem melengkapi parameter laporan gangguan secara otomatis sebelum diteruskan ke *Customer Service* atau NOC. Visualisasi interaksi sistem pada dua skenario gangguan teknis ditunjukkan pada Gambar 3 dan Gambar 4.



Gambar 3. Gangguan Umum



Gambar 4. Gangguan Aplikasi

4 SIMPULAN

Berdasarkan hasil perancangan dan pengujian yang telah dilakukan, implementasi *Chatbot* Layanan Pelanggan berbasis *Natural Language Processing* (NLP) yang diintegrasikan dengan platform Telegram terbukti efektif berfungsi sebagai asisten virtual Level 1 (*L1 Support*) pada *Internet Service Provider* (ISP). Sistem mampu menangani keluhan pelanggan secara otomatis selama 24 jam tanpa intervensi manual, sehingga meningkatkan ketersediaan layanan dukungan teknis.

Dari sisi performa, penerapan metode NLP berbasis TF-IDF dan *cosine similarity* dengan mekanisme *slot-filling* menunjukkan kemampuan yang baik dalam mengenali variasi bahasa pelanggan. Berdasarkan evaluasi menggunakan *Confusion Matrix* terhadap data uji, sistem mencatatkan tingkat akurasi sebesar 86%, presisi 92,3%, dan *recall* 90%. Hasil ini menunjukkan bahwa sistem memiliki tingkat keandalan yang cukup tinggi dalam mengklasifikasikan keluhan teknis maupun pertanyaan administratif.

Meskipun demikian, sistem masih memiliki keterbatasan dalam menangani penggunaan istilah non-baku, bahasa slang, dan konteks gangguan yang ambigu. Namun secara keseluruhan, implementasi *chatbot* ini mampu berkontribusi signifikan dalam mengurangi antrean pesan (*bottleneck*) saat terjadi gangguan massal dan mempercepat waktu respon layanan, sehingga teknisi jaringan dapat lebih fokus pada penanganan infrastruktur fisik yang bersifat kritis.

5 SARAN

Penelitian ini masih memiliki potensi untuk dikembangkan lebih lanjut guna meningkatkan fungsionalitas dan kecerdasan sistem. Pengembangan lanjutan dapat diarahkan pada integrasi *chatbot* dengan *Application Programming Interface* (API) perangkat jaringan, seperti *MikroTik RouterOS* atau *Optical Line Terminal* (OLT), sehingga sistem mampu melakukan pengecekan parameter teknis jaringan secara otomatis dan *real-time*.

Selain itu, kemampuan pemahaman bahasa dapat ditingkatkan melalui perluasan basis pengetahuan yang mencakup istilah teknis lokal, bahasa slang, serta variasi dialek pengguna. Pengembangan fitur antarmuka juga dapat diarahkan pada penambahan modul *Voice-to-Text* untuk mendukung laporan berbasis suara. Apabila kompleksitas data percakapan meningkat secara signifikan, pendekatan berbasis *deep learning* seperti LSTM atau BERT dapat dipertimbangkan sebagai pengembangan lanjutan untuk meningkatkan pemahaman konteks semantik secara lebih mendalam.

DAFTAR PUSTAKA

- [1] E. M. Kusumaningtyas, A. Barakbah, dan C. W. Murti, “*Chatbot* tentang penerimaan mahasiswa baru,” *Jurnal Informatika*, vol. xx, no. xx, hal. 1–10, 2022.
- [2] T. A. Zuraiyah, D. K. Utami, dan D. Herlambang, “Implementasi *Chatbot* Pada Pendaftaran Mahasiswa Baru Menggunakan *Recurrent Neural Network*,” *J. Ilm. Teknol. dan Rekayasa*, vol. 24, no. 2, hal. 91–101, 2019.
- [3] D. Apriliani, S. F. Handayani, dan I. T. Saputra, “Implementasi *Natural Language Processing* (NLP) Dalam Pengembangan Aplikasi *Chatbot* Pada SMK YPE Nusantara Slawi,” *Techno.Com*, vol. 22, no. 4, hal. 1037–1047, 2023.
- [4] M. Furqan, S. Sriani, dan M. N. Shidqi, “*Chatbot* Telegram Menggunakan *Natural Language Processing*,” *Walisongo J. Inf. Technol.*, vol. 5, no. 1, hal. 15–26, 2023.
- [5] M. R. Muzaki, I. M. Nurfajriana, P. A. Ilahi, A. Ristyawan, dan E. Daniati, “Implementasi Data Mining dengan Algoritma *K-Nearest Neighbors* untuk Memprediksi Risiko Diabetes Menggunakan *Chatbot* Telegram,” *Prosiding SEMNAS*, vol. 8, hal. 310–318, 2024.
- [6] I. A. Rohman, D. Swanjaya, dan A. Sanjaya, “Implementasi NLIDB Pada *Chatbot* CV Owlsoft Media,” *Semin. Nas. Teknol. Sains (STAINS)*, vol. 3, no. 1, hal. 304–309, 2024.
- [7] A. Sanjaya, A. B. Setiawan, U. Mahdiyah, I. N. Farida, dan A. R. Prasetyo, “Pengukuran Kemiripan Makna Menggunakan *Cosine Similarity* dan Basis Data Sinonim Kata,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 10, no. 4, hal. 747-752, 2023.
- [8] Nasution, M. A., Fitri, A., & Rizwinie, K. S. (2024). “Implementasi NLP dalam pembuatan *chatbot* customer service: Studi kasus LARISMA.” *Jurnal Sains, Teknologi & Komputer (SAINTEK)*, 1(1), 13–17.