INOTEK, Vol. 10
ISSN: 2580-3336 (Print) / 2549-7952 (Online)
Url: https://proceeding.unpkediri.ac.id/index.php/inotek/

Evaluasi Sentence-BERT pada Arsitektur Hybrid Chatbot ISP untuk Penanganan Out-of-Vocabulary

[1*]Ivan Yusuf Sholy Khudin, [2]Ahmad Bagus Setiawan
[1,2] Teknik Informatika, Universitas Nusantara PGRI Kediri
E-mail: *[1]ivanyusuf200@gmail.com, [2]ahmadbagus@unp.ac.id
Penulis Koresponden : Ivan Yusuf Sholy Khudin

Abstrak— Pendekatan *rule-based* pada *chatbot* layanan pelanggan sering kali gagal mengenali maksud pengguna ketika terdapat kesalahan ketik atau kata di luar kosakata sistem (*Out-of-Vocabulary*/OOV). Penelitian ini berfokus mengevaluasi kinerja model Sentence-BERT (SBERT) sebagai solusi penanganan OOV pada *chatbot* keluhan jaringan internet. Berbeda dengan pengembangan sistem *chatbot* secara utuh, penelitian ini mengisolasi pengujian khusus pada 300 *dataset* kalimat pelanggan yang tidak baku dan mengandung ambiguitas. Kalimat uji diproses menggunakan SBERT untuk mengekstraksi representasi semantik dan diklasifikasikan ke dalam *intent* spesifik. Hasil evaluasi menunjukkan model mampu mengidentifikasi *intent* dengan akurasi mencapai 88,33%, jauh lebih baik dibandingkan pencocokan kata kunci konvensional, meskipun dengan kompensasi waktu komputasi yang sedikit meningkat pada untuk sisi *server*. Kesimpulannya, implementasi SBERT efektif bertindak sebagai lapisan koreksi otomatis meminimalisir kegagalan respons *chatbot*akibat variasi bahasa natural pelanggan.
Kata Kunci— *Chatbot, Out-of-Vocabulary,* Semantik, *Sentence-BERT*

Abstract— *The rule-based approach in customer service chatbots often fails to recognize user intent when encountering typographical errors or words outside the system's dictionary (Out-of-Vocabulary/OOV). This study focuses on evaluating the performance of the Sentence-BERT (SBERT) model as a solution for handling OOV in an internet network complaint chatbot. Unlike the development of a complete chatbot system, this research isolates its testing specifically on a dataset of 300 non-standard and ambiguous customer sentences. The test sentences are processed using SBERT to extract semantic representations and are then classified into specific intents. The evaluation results indicate that the model can identify intents with an accuracy reaching 88.33%, significantly better compared to conventional keyword matching, despite a slight increase in server-side computational time. In conclusion, the implementation of SBERT effectively acts as an automatic correction layer to minimize chatbot response failures caused by customers' natural language variations.*
Keywords— *Chatbot, Out-of-Vocabulary,* Semantik, *Sentence-BERT*

I. PENDAHULUAN

Dalam industri telekomunikasi, khususnya penyedia layanan internet (*Internet Service Provider*/ISP), kecepatan dan ketepatan respons terhadap keluhan pelanggan merupakan indikator krusial kualitas layanan (*Quality of Service*). Saat terjadi gangguan jaringan massal, *Customer Service* (CS) sering mengalami kelebihan beban kerja akibat lonjakan pesan masuk (*message storm*) [1]. Untuk mengatasi penumpukan antrean pesan, teknologi *chatbot* banyak diadopsi guna mengotomatisasi layanan [2]. Secara umum, sistem *chatbot* pada industri

penyedia layanan masih mengandalkan pendekatan *rule-based* atau *pattern matching*. Pendekatan ini efisien secara komputasi, namun memiliki kelemahan mendasar, sistem gagal mengenali maksud pengguna (*intent*) apabila terdapat kesalahan pengetikan (*typo*), singkatan tidak baku, atau kata di luar kosakata sistem (*Out-of-Vocabulary/OOV)* [3].

Beberapa penelitian sebelumnya berupaya mengintegrasikan *Natural Language Processing* (NLP) untuk mengatasi keterbatasan tersebut. Sebuah penelitian mengembangkan *chatbot* layanan pelanggan menggunakan metode klasifikasi *intent* berbasis *machine learning*, namun sistem masih rentan terhadap ambiguitas kalimat pengguna [4]. Penelitian lain mengevaluasi metode *pattern-based* dan menemukan bahwa akurasinya menurun signifikan saat memproses teks pelanggan yang tidak terstruktur (*noisy data*) [5]. Penelitian lanjutan mengenai penanganan OOV pada teks bahasa Indonesia menunjukkan bahwa pencocokan kata kunci konvensional tidak cukup untuk memahami konteks semantik kalimat secara menyeluruh [6]. Terdapat kesenjangan penelitian dalam memfokuskan penanganan teks *noisy* ini tanpa harus mengubah arsitektur *chatbot* utama yang telah berjalan optimal. Oleh karena itu, diperlukan sebuah mekanisme penanganan alternatif (*fallback layer*) ketika metode *rule-based* tidak menemukan kecocokan pola.

Penelitian ini bertujuan mengevaluasi kinerja model *Sentence-BERT* (SBERT) sebagai solusi penanganan OOV pada *chatbot* layanan keluhan jaringan internet. SBERT merupakan modifikasi dari arsitektur BERT yang menggunakan *siamese network* untuk menghasilkan representasi vektor kalimat (*sentence embeddings*) [7], [9]. Berbeda dengan penelitian sebelumnya yang merancang sistem *chatbot* secara penuh, penelitian ini membatasi lingkup pengujian pada *dataset* kalimat ambigu dan OOV yang gagal diproses oleh sistem *rule-based*.

Hipotesis penelitian ini adalah penerapan algoritma SBERT dapat memetakan *intent* dari kalimat OOV dengan tingkat akurasi yang lebih tinggi dibandingkan pencocokan karakter konvensional, dengan memperhitungkan adanya penambahan beban komputasi di sisi *server*. Hasil penelitian ini diharapkan memberikan dampak praktis berupa efisiensi penyaringan keluhan, penurunan tingkat kegagalan respons, serta meminimalkan kesalahan eskalasi tiket gangguan ke teknisi lapangan.

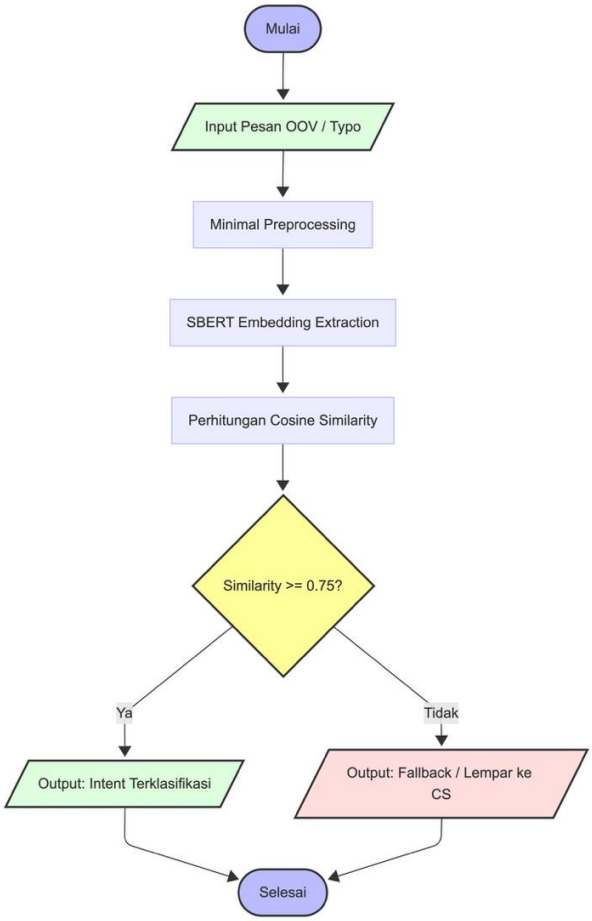
II. METODE

Penelitian ini merupakan penelitian eksperimental kuantitatif yang berfokus pada pengujian performa algoritma NLP. Subjek penelitian adalah sampel data percakapan pelanggan dari PT Gayatri Lintas Nusantara wilayah operasional Kediri Raya, Jawa Timur. Pengambilan sampel dilakukan dengan teknik *purposive sampling*, di mana 300 data kalimat pelanggan diekstraksi secara spesifik dari log percakapan karena teridentifikasi mengandung *typo*, singkatan, dan kalimat ambigu yang gagal diproses oleh mesin *rule-based* utama [7].

Prosedur eksperimen diawali dengan tahap prapemrosesan teks dasar, yang meliputi *case folding* dan penghapusan karakter khusus, guna mempertahankan struktur kalimat asli dari pengguna. Data uji kemudian dimasukkan ke dalam model SBERT (*pretrained model* bahasa Indonesia) untuk diubah menjadi representasi vektor (*dense vector*). Variabel pengukuran meliputi tingkat akurasi prediksi *intent* dan waktu komputasi (*inference time*). Keputusan klasifikasi *intent*dihitung berdasarkan metrik *Cosine Similarity* antara vektor masukan pengguna dengan vektor *dataset* latih pada *knowledge base* [8].

A. Core Backend dan Arsitektur Subsistem Semantik

Pesan OOV yang tidak terdeteksi oleh penyaring kata kunci utama dikonversi menjadi representasi vektor menggunakan arsitektur SBERT. Proses ini menghasilkan nilai *embedding* yang merepresentasikan makna kalimat secara keseluruhan. Vektor dari pesan pelanggan tersebut selanjutnya dibandingkan dengan vektor *intent* acuan yang tersimpan pada *knowledge base* perusahaan. Alur lengkap proses pemfilteran teks OOV ini diilustrasikan pada Gambar 1.



Gambar 1. Alur Pemrosesan Teks OOV

Pada implementasi perangkat lunak, sistem diatur oleh *core backend* berbasis FastAPI yang meneruskan logika arsitektur layanan pelanggan yang telah dibangun sebelumnya [10]. Modul ini bertindak sebagai gerbang interaksi yang menerima permintaan data via *webhook*, melakukan validasi, dan mengelola logika percabangan (*routing*). Alur pemrosesan dirancang secara hierarkis, lapisan pertama menggunakan pencocokan kata kunci (*Rule Engine*), dan lapisan kedua menggunakan *ML Engine*. Komponen *Semantic Engine* dipanggil sebagai lapisan ketiga apabila kedua lapisan sebelumnya gagal menghasilkan tingkat keyakinan (*confidence score*) yang memenuhi standar. Dokumentasi penempatan logika *routing* pada berkas *core backend* disajikan pada Gambar 2.

```
# Layer 3: Semantic Engine
logger.info(f"MLEngine low confidence, applying Similarity Scoring for: {message}")
if is_ambiguous:
    sem_threshold = 0.98
else:
    sem_threshold = 0.80 if is_short_input else 0.55 # Sedikit diturunkan agar lebih sensitif
sem_res = semantic_engine.detect(message, threshold=sem_threshold)
if isinstance(sem_res, tuple) and len(sem_res) == 3:
    sem_intent, sem_score, sem_reply = sem_res
else:
    sem_intent, sem_score, sem_reply = None, 0.0, None

if sem_intent is not None:
    intent = str(sem_intent)
    engine_source = "semantic_engine"
    confidence = float(sem_score)
    base_reply = str(sem_reply)
    status = route_intent(intent)
else:
    # Final Fallback jika semua engine gagal
    confidence = 0.0
    engine_source = "fallback"
    base_reply = random.choice(FALLBACK_RESPONSES)
```

Gambar 2. Penataan Arsitektur Kode Program

Logika pemrosesan bahasa alami diimplementasikan melalui *class Semantic Engine* yang menggunakan model *paraphrase-multilingual-MiniLM-L12-v2*. Pada tahap inisialisasi lingkungan kerja, sistem melakukan ekstraksi vektor (*encoding*) terhadap seluruh aturan baku untuk disimpan sebagai tensor referensi di dalam memori *server*. Kesiapan lingkungan eksekusi subsistem ini ditunjukkan pada Gambar 3.

```
# Layer 3: Semantic Engine
logger.info(f"MLEngine low confidence, applying Similarity Scoring for: {message}")
if is_ambiguous:
    sem_threshold = 0.98
else:
    sem_threshold = 0.80 if is_short_input else 0.55 # Sedikit diturunkan agar lebih sensitif
sem_res = semantic_engine.detect(message, threshold=sem_threshold)
if isinstance(sem_res, tuple) and len(sem_res) == 3:
    sem_intent, sem_score, sem_reply = sem_res
else:
    sem_intent, sem_score, sem_reply = None, 0.0, None

if sem_intent is not None:
    intent = str(sem_intent)
    engine_source = "semantic_engine"
    confidence = float(sem_score)
    base_reply = str(sem_reply)
    status = route_intent(intent)
else:
    # Final Fallback jika semua engine gagal
    confidence = 0.0
    engine_source = "fallback"
    base_reply = random.choice(FALLBACK_RESPONSES)
```

Gambar 3. Pemuatan Ekstraksi Vektor

Ketika pesan OOV masuk ke dalam fungsi *detect()*, teks masukan diektraksi secara *real-time* menjadi *query embedding*. Komputasi internal kemudian melakukan operasi perkalian matriks menggunakan pustaka utilitas bawaan untuk menghitung nilai kemiripan

kosinus terhadap seluruh tensor referensi. Pipeline transformasi vektor dinamis tersebut akan terekam pada aktivitas log konsol terminal yang dikonfigurasi pada Gambar 4.

```
for row in rows:
    msg = row['message']
    y_true.append(row['intent'])
    r_intent, _, _ = rule_engine.detect_with_response(msg)

    if r_intent == "fallback":
        ml_res = ml_engine.predict(msg, threshold_design=0.5)
        p_intent = ml_res[0] if isinstance(ml_res, tuple) and len(ml_res) > 0 else "fallback"

        if p_intent == "fallback":
            sem_res = semantic_engine.detect(msg, threshold=0.6)
            p_intent = sem_res[0] if isinstance(sem_res, tuple) and len(sem_res) > 0 and sem_res[0] else "fallback"
    else:
        p_intent = r_intent

    y_pred.append(p_intent or "fallback")
```

Gambar 4. Pipeline Inferensi Komputasi Vektor pada Eksekusi Semantic Engine

B. Rumus Cosine Similarity

Untuk menentukan tingkat kemiripan vektor antara pesan masukan pengguna *(A)* dengan pesan acuan *(B)*, digunakan metrik perhitungan *Cosine Similarity*. Perhitungan matematika untuk mendapatkan nilai kosinus di antara dua vektor tersebut disajikan pada Persamaan (1)

$$Cosine\ Similarity\ (A.B) = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_{i-1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

Hasil dari Persamaan (1) berupa nilai skalar yang bergerak di rentang 0 hingga 1. Jika nilai kedekatan semantik memenuhi batas ambang (*threshold*) baku , maka pesan dinilai valid dan sistem akan mengeksekusi jawaban otomatis yang sesuai dengan Standar Operasional Prosedur (SOP) tanpa perlu melempar penolakan respons (*fallback error*).

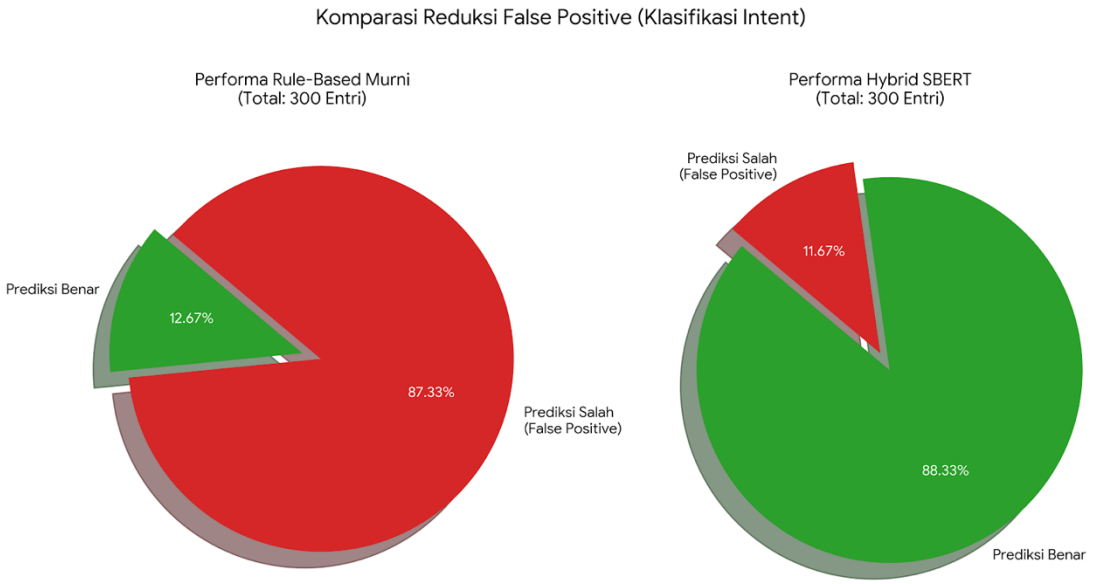
III. HASIL DAN PEMBAHASAN

Data pengujian dalam penelitian ini diproduksi melalui proses isolasi terhadap log aktivitas pesan masuk pada sistem layanan PT Gayatri Lintas Nusantara. Dari total rekaman interaksi, diambil 300 entri data spesifik yang dikategorikan sebagai kegagalan sistematis oleh *engine rule-based* karena tidak memenuhi kriteria pencocokan karakter eksak (*Out-of-Vocabulary*). Eksperimen dilakukan dengan memproses ulang seluruh korpus data tidak terstruktur tersebut ke dalam lapisan arsitektur *Sentence-BERT* (SBERT) untuk dievaluasi kemampuan adaptasi semantiknya. Pengujian performa berfokus pada pengukuran metrik akurasi, reduksi kesalahan interpretasi, dan pengukuran waktu komputasi inferensi *server* yang dirangkum pada Tabel 1.

Tabel 1. Perbandingan Performa Klasifikasi pada Korpus Data OOV

Parameter Evaluasi	Pendekatan Rule Based Murni	Pendekatan SBERT Hybrid	Tingkat Efisiensi Sistem
Prediksi Benar *(True Positive)*	38 Entri	265 Entri	Kenaikan +227 Entri
Prediksi Salah *(False Positive)*	262 Entri	35 Entri	Penurunan -227 Entri
Rata-rata Akurasi	12.67%	88.33%	Peningkatan 75,66%
Waktu Inferensi Rata-rata	0,04 Detik	0,41 Detik	Latensi +0,37 Detik

Hasil eksperimen dari pemrosesan 300 data tersebut menunjukkan peningkatan rata-rata akurasi klasifikasi sebesar 75,66% pasca implementasi pendekatan *hybrid* SBERT, sehingga akurasi mencapai 88,33%. Metode *rule-based* murni menghasilkan tingkat prediksi salah yang tinggi (262 entri/87,33%) karena ketiadaan fleksibilitas sistem dalam menoleransi variasi penulisan. Komparasi visual terhadap lonjakan akurasi dan reduksi drastis nilai *False Positive* dari 262 entri menjadi hanya 35 entri dapat dilihat secara komprehensif pada Gambar 5. Keberhasilan menekan tingkat kesalahan interpretasi ini memberikan dampak praktis yang signifikan terhadap efisiensi penanganan gangguan. Secara operasional, validitas identifikasi awal tersebut berfungsi krusial untuk meminimalkan akumulasi penugasan tiket palsu (*false alarm*) pada tim *Network Operation Center* (NOC).



Gambar 5. Perbandingan Hasil Klasifikasi *Intent* Data OOV

Sebagai validasi kualitatif terhadap hasil statistik sebelumnya, Gambar 6 menyajikan cuplikan log percakapan dari eksekusi sistem terhadap pesan pelanggan. Pada log tersebut,

arsitektur SBERT terbukti mampu memproses kalimat riil yang memiliki tingkat variasi *noise* penulisan tinggi, seperti penggunaan beberapa singkatan tidak baku ("wipi", "bntr2", dan "rto"). Meskipun susunan karakter tersebut gagal dipahami oleh mesin pencocok pola standar, sistem berbasis SBERT berhasil memetakan pesan masukan ke dalam *intent* solusi yang tepat. Keberhasilan klasifikasi ini terjadi karena kalkulasi komputasi vektor (*cosine similarity*) dari kalimat tersebut menghasilkan skor yang melampaui ambang batas minimal konfigurasi sistem (≥0.75).

message	intent	status	confidence	engine
sudah dicoba colok ulang tapi tetep tidak keluar sinyal mya	gangguan_lemot_umum	TO_NOC	0.56431913	semantic_engine
kira* bisa d pasang kapan kak	pasang_baru	AUTO_RESPONSE	0.746686	semantic_engine
kak internetnyaa kenapaa lampunyaa berwarna merahh	kabel_putus	TO_NOC	0.81621563	semantic_engine
ini untuk internetnya bs nya kapan y?	gangguan_umum	TO_NOC	0.73312306	semantic_engine
wifiku muter	gangguan_lemot_umum	TO_NOC	0.8824794	semantic_engine
ini kok sinyal nya merah kenapa ya? kan belum waktunya baya…	kabel_putus	TO_NOC	0.67693114	semantic_engine
internet saya barusan kok lambat ya	gangguan_umum	TO_NOC	0.6920353	semantic_engine

Gambar 6. Cuplikan Log Percakapan Pengujian *Chatbot* pada Teks OOV

Evaluasi terhadap 300 data ini secara empiris mengonfirmasi bahwa pemrosesan berbasis semantik memiliki performa yang jauh lebih unggul dibandingkan pendekatan pencocokan kata kunci konvensional dalam mengenali bahasa alami yang tidak baku. Penurunan rasio *False Positive* dari 262 entri menjadi 35 entri memberikan dampak langsung pada efisiensi operasional penyedia layanan. Secara manajerial, akurasi filtrasi ini berfungsi menekan akumulasi *false alarm*pada sistem tiket eskalasi perusahaan, sehingga tim *Network Operation Center* (NOC) tidak terbebani oleh antrean penugasan penanganan gangguan yang tidak valid. Arsitektur SBERT terbukti berhasil mengatasi kendala performa saat memproses teks tidak terstruktur (*noisy data*) melalui mekanisme ekstraksi representasi vektor yang mampu menangkap kedekatan makna kalimat secara komprehensif.

Meskipun demikian, pencatatan waktu pemrosesan di sisi *server* menunjukkan waktu inferensi rata-rata mencapai 0,41 detik. Hal ini mengindikasikan adanya penambahan latensi sebesar 0,37 detik dibandingkan metode *rule-based* murni. Peningkatan durasi komputasi ini merupakan *trade-off* yang rasional akibat proses perkalian matriks berdimensi tinggi pada model *Transformer* saat mengekstraksi nilai *embedding* kalimat. Meskipun terdapat peningkatan beban *server* yang berpotensi menjadi batas kinerja (*bottleneck*) saat terjadi lonjakan permintaan (*message storm*), rentang waktu tersebut masih berada di bawah 1 detik sehingga tergolong wajar untuk interaksi sistem-pengguna. Isu latensi ini merupakan ranah optimasi lanjutan, di mana implementasi teknik kompresi model seperti kuantisasi bobot (*weight quantization*) dapat disarankan untuk studi berikutnya.

IV. KESIMPULAN

Penelitian ini menyimpulkan bahwa implementasi model *Sentence-BERT* (SBERT) efektif memitigasi kegagalan respons *chatbot* akibat variasi pesan *Out-of-Vocabulary* (OOV) pada layanan pelanggan ISP. Berdasarkan pengujian terhadap 300 data kalimat tidak baku yang telah diuraikan pada tahap eksperimen, penerapan arsitektur semantik ini terbukti secara linier meningkatkan rata-rata akurasi klasifikasi *intent* hingga mencapai 88,33%. Capaian teknis ini

mengonfirmasi hipotesis bahwa perhitungan metrik kedekatan vektor mampu mengatasi kelemahan fleksibilitas pada metode pencocokan konvensional berbasis aturan (*rule-based*).

Kontribusi praktis dari penelitian ini adalah tersedianya lapisan penanganan alternatif (*fallback layer*) yang mengoptimalkan otomatisasi layanan keluhan. Reduksi tingkat interpretasi salah (*False Positive*) secara langsung memperbaiki alur manajemen gangguan, mencegah eskalasi tiket palsu (*false alarm*), dan mengefisiensikan alokasi penugasan tim teknis lapangan. Meskipun proses ekstraksi vektor menambah latensi komputasi *server* sebesar 0,37 detik, durasi tersebut masih berada pada batas wajar interaksi sistem-pengguna. Kendala penambahan beban komputasi ini membuka peluang pengembangan lebih lanjut, khususnya pada penerapan metode kompresi model untuk penelitian mendatang.

UCAPAN TERIMAKASIH

Penulis menyampaikan terima kasih yang sebesar-besarnya kepada pihak manajemen operasional PT Gayatri Lintas Nusantara atas izin akses korpus data operasional dan PT Jenggala Lintas Nusantara sebagai perantara yang telah memfasilitasi kebutuhan data historis layanan pelanggan serta memberikan dukungan operasional selama penelitian ini berlangsung.

DAFTAR PUSTAKA

[1] A. E. N. Nasser, F. Fattah, and L. B. Ilmawan, "Analisis Quality Of Service Layanan Internet Service Provider pada Esports MOBA," *LINIER: Literatur Informatika dan Komputer*, vol. 1, no. 3, pp. 248–255, Dec. 2024, doi: 10.33096/linier.v1i3.2502.

[2] S. R. M. Ibrahim and D. Handayani, "Implementasi Chatbot Berbasis Aturan untuk Layanan Customer Service E-commerce pada Platform WhatsApp," *Journal of Information System Research (JOSH)*, vol. 7, no. 1, pp. 39–46, Oct. 2025, doi: 10.47065/josh.v7i1.8443.

[3] J. Khan, K. Ahmad, S. K. Jagatheesaperumal, and K. A. Sohn, "Textual variations in social media text processing applications: challenges, solutions, and trends," *Artif. Intell. Rev.*, vol. 58, no. 3, Mar. 2025, doi: 10.1007/s10462-024-11071-z.

[4] M. Akbar Nasution *et al.*, "Implementasi NLP Dalam Pembuatan Chatbot Customer Service Publisher Jurnal Studi Kasus LARISMA," Jan. 2024. doi: 10.56495/saintek.v1i1.451.

[5] T. Astuti Rahayu, A. Kharisma Hidayah, H. Witriyono, and R. Guntur Alam, "Implementasi Metode Pattern-Based Natural Language Processing (NLP) Pada Chatbot," *JSAI: Journal Scientific and Applied Informatics*, vol. 7, no. 3, 2024, doi: 10.36085/jsai.v7i3.6739

[6] M. Abdul, H. Fathuddin, E. Prakarsa Mandyartha, and A. L. Nurlaili, "Penerapan Sentence-Bert dan Cosine Similarity untuk Pencarian Semantik Dokumen Skripsi dalam Format PDF," *R2J*, vol. 8, no. 1, 2025, doi: 10.38035/rrj.v8i1.

[7] H. Hartatik and A. Syafrianto, "PENERAPAN MODEL SENTENCE-BERT UNTUK SISTEM REKOMENDASI BUKU BERBASIS KONTEN DI PERPUSTAKAAN DIGITAL," *Jurnal Dialektika Informatika (Detika)*, vol. 6, no. 1, pp. 12–19, Nov. 2025, doi: 10.24176/detika.v6i1.15916.

[8] I. Hakiki, F. Atqiya, and A. Suryan, "Implementasi Model Sentence-Bert untuk Deteksi Plagiarisme Pada Karya Tulis Ilmiah Psikologi Berbahasa Indonesia," *Cerdika: Jurnal Ilmiah Indonesia*, vol. 5, no. 12, 2025, doi: 10.59141/cerdika.v5i12.2889.

[9] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Aug. 2019, doi: 10.48550/arXiv.1908.10084.

[10] I. Yusuf, S. Khudin, A. Sanjaya, and A. B. Setiawan, "Implementasi Chatbot Layanan Pelanggan Berbasis NLP Menggunakan Platform Telegram Pada Penyedia Layanan Internet," Kediri, Jan. 2026. doi: 10.29407/vbgfc739.