VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

INTI NUSA MANDIRI
DOI: https://doi.org/10.33480/inti.v20i1.5766

PENGEMBANGAN CHATBOT TELEGRAM FAQ LAYANAN ICT MENGGUNAKAN ALGORITMA RANDOM FOREST DAN METODE WORD2VEC

Muhammad Arif Setiyawan[1*]; Erina Divaa Kenoya[2]

Sistem Informasi[1,2]
Universitas Bina Sarana Informatika, Jakarta, Indonesia [1,2]
www.bsi.ac.id [1,2]
muhamadarief251@gmail.com[1*], erinadivaakenoya@gmail.com[2]

(*) Corresponding Author

Abstract— *In today's digital era, chatbots have become an essential tool for businesses to improve interaction with customers. A responsive and efficient chatbot can help customer service agents be happier, improve customer satisfaction, and resolve issues faster. The study aims to create a Telegram-based chatbot that uses the Random Forest algorithm and the Word2Vec method to answer questions about ICT services. The development was carried out by collecting a dataset of questions and answers from FAQs (Frequently Asked Questions) of ICT services. Then, the Random Forest algorithm is used to classify the questions. In addition, the Word2Vec method is used to create vector representations of words in questions and answers. This improves the chatbot's ability to understand complex questions. The test results show that the chatbot gets an accuracy of 91.28%, a precision of 93.56%, a recall of 91.28% and an F1-Score of 91.42% and can provide relevant and accurate answers to user questions. Therefore, the development of this chatbot using the Random Forest algorithm and the Word2Vec method can be an effective solution to improve customer service in the field of ICT services.*

Keywords*: Chatbots*, ICT *Services, Random Forest Algorithms*, *Word2Vec*.

Abstrak— Dalam era digital saat ini, *chatbot* telah menjadi alat penting bagi bisnis untuk meningkatkan interaksi dengan pelanggan. *Chatbot* yang responsif dan efisien dapat membantu agen layanan pelanggan menjadi lebih bahagia, meningkatkan kepuasan pelanggan, dan menyelesaikan masalah dengan lebih cepat. Studi ini bertujuan untuk membuat *chatbot* berbasis Telegram yang menggunakan algoritma *Random Forest* dan metode *Word2Vec* untuk menjawab pertanyaan tentang layanan ICT. Pengembangan dilakukan dengan mengumpulkan *dataset* pertanyaan dan jawaban dari FAQ (*Frequently Asked Questions*) layanan ICT. Kemudian, algoritma *Random Forest* digunakan untuk mengklasifikasikan pertanyaan. Selain itu, metode *Word2Vec* digunakan untuk membuat representasi vektor kata-kata dalam pertanyaan dan jawaban. Ini meningkatkan kemampuan *chatbot* untuk memahami pertanyaan yang kompleks. Hasil pengujian menunjukkan bahwa *chatbot* mendapatkan akurasi sebesar 91.28%, presisi sebesar 93.56%, *recall* sebesar 91.28% dan *F1-Score* sebesar *91.42*% serta dapat memberikan jawaban yang relevan dan akurat untuk pertanyaan pengguna. Oleh karena itu, pengembangan *chatbot* ini menggunakan algoritma *Random Forest* dan metode *Word2Vec* dapat menjadi solusi yang efektif untuk meningkatkan layanan pelanggan di bidang layanan ICT.

Kata kunci: *Chatbot*, Layanan ICT, Algoritma *Random Forest*, *Word2Vec*.

PENDAHULUAN

Istilah ICT (*Information and Communication Technology*) mengacu pada berbagai teknologi yang digunakan untuk mengumpulkan, menyimpan, mengirimkan, dan memanipulasi informasi. Contoh teknologi ini termasuk komputer, perangkat lunak, jaringan,

VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

internet, dan perangkat komunikasi lainnya. Tujuan utama ICT adalah untuk meningkatkan efisiensi dan produktivitas dalam pengolahan informasi dan komunikasi. Dalam layanan ICT, teknologi ini digunakan untuk menyediakan layanan informasi dan komunikasi kepada pengguna. Layanan ini termasuk layanan *help desk*, jaringan komputer, pengelolaan basis data, dan sebagainya. FAQ, singkatan dari "*Frequently Asked Questions*" yang berarti "Pertanyaan yang Sering Diajukan", "layanan ini berisi sebuah list pertanyaan-pertanyaan yang sering diajukan oleh pengunjung ataupun *user* mengenai sebuah topik, beserta jawabannya" (Khoiriyah & Afriati, 2022). FAQ biasanya dirancang untuk memberikan informasi bermanfaat dan menjawab pertanyaan umum tanpa perlu menghubungi layanan pelanggan atau pihak yang berwenang. FAQ sering digunakan di situs web, aplikasi, atau layanan pelanggan untuk memberikan panduan pengguna, menghemat waktu bagi pengguna dan pihak yang menyediakan informasi.

Pelayanan ICT (*Information and Communication Technology*) sangat penting untuk operasi suatu organisasi. Karena banyaknya permintaan layanan ICT, penyedia layanan harus dapat memberikan layanan yang cepat, akurat, dan responsif terhadap kebutuhan pengguna. Salah satu cara untuk meningkatkan efisiensi layanan ICT adalah dengan menggunakan *chatbot*, yang dapat memberikan informasi dan menjawab pertanyaan yang sering diajukan oleh pengguna. *Chatbot* telah menjadi alat yang populer untuk meningkatkan layanan pelanggan di berbagai industri, termasuk industri ICT. Mereka dapat mengurangi waktu tunggu, menjawab pertanyaan pengguna, dan menangani permintaan layanan dengan lebih efisien. Namun, *chatbot* harus dikembangkan dengan metode dan algoritma yang tepat untuk berfungsi dengan baik.

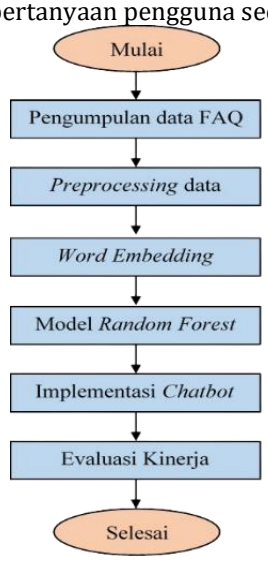
Salah satu algoritma pembelajaran mesin yang dapat digunakan untuk klasifikasi data adalah *Random Forest*. *Chatbot* dapat menggunakan algoritma ini untuk mempelajari pola-pola dalam data FAQ ICT sehingga mereka dapat memberikan jawaban yang lebih akurat. Metode *Word2Vec* digunakan untuk mengkonversi teks ke vektor numerik. *Chatbot* dapat menggunakan representasi vektor ini untuk memahami konteks pertanyaan pengguna dan memberikan jawaban yang lebih sesuai. Berdasarkan penelitian terdahulu yang dilakukan oleh (Puspita & Kalifia, 2024) ini mendapatkan hasil yang cukup memuaskan untuk penggunaan algoritma *Random Forest*. Hasil penelitian menunjukkan bahwa tingkat depresi pada mahasiswa selama pandemi COVID-19 dapat diprediksi secara efektif menggunakan algoritma *Random Forest*. Model yang dikembangkan mampu mengidentifikasi faktor-faktor yang paling berpengaruh terhadap tingkat depresi, memberikan pemahaman mendalam tentang dinamika kesehatan mental mahasiswa selama periode yang menantang ini. Selain itu, berdasarkan penelitian yang dilakukan (Ciptady et al., 2022) ini algoritma yang dipakai yaitu *Random Forest* mendapatkan hasil akurasi yang memuaskan, Akurasi algoritma *Random Forest* dalam memprediksi kualitas kopi pada penelitian ini adalah sebesar 79%. Berdasarkan penelitian yang dilakukan oleh (Af'idah et al., 2021) ini menyimpulkan akurasi yang baik pada penggunaan *Word2Vec.* Kesimpulan dari penelitian ini menunjukkan bahwa ketiga parameter *Word2Vec* memiliki pengaruh terhadap performa model *deep learning* dalam melakukan klasifikasi sentimen. Kombinasi parameter *Word2Vec* yang menghasilkan rata-rata akurasi tertinggi antara lain: arsitketur CBOW, metode evaluasi *Hierarchical Softmax*, dan dimensi bernilai 100.

Berdasarkan permasalahan diatas maka dibuatlah *chatbot* untuk mengakomodir jawaban dari pertanyaan seputar layanan ICT (*Information and Communication Technology*) yang sering ditanyakan oleh pengguna. *Chatbot* yang dibuat akan menggunakan kombinasi antara algoritma *Random Forest* dan metode *Word2Vec*. Kombinasi tersebut diharapkan dapat meningkatkan performa *chatbot* serta kualitas layanan yang diberikan kepada pengguna dengan memberikan jawaban yang lebih cepat, akurat, dan relevan. Tujuan dari penelitian ini adalah melakukan pengembangan *chatbot* berbasis Telegram untuk menjawab pertanyaan tentang layanan ICT (*Information and Communication Technology*) dengan menggunakan kombinasi antara algoritma *Random Forest* dan metode *Word2Vec*.

BAHAN DAN METODE

Pengembangan chatbot ini dilakukan melalui beberapa tahapan, dimulai dari pengumpulan data *Frequently Asked Questions* (FAQ) terkait layanan ICT sebagai basis data, yang kemudian diproses melalui beberapa tahap penting. Tahapan tersebut meliputi *preprocessing* data, pembuatan representasi kata menggunakan teknik *Word Embedding*, dan penerapan algoritma *Random Forest* untuk mengklasifikasikan pertanyaan yang diajukan. Setelah model *Random Forest* dibangun, *chatbot* diimplementasikan dalam platform

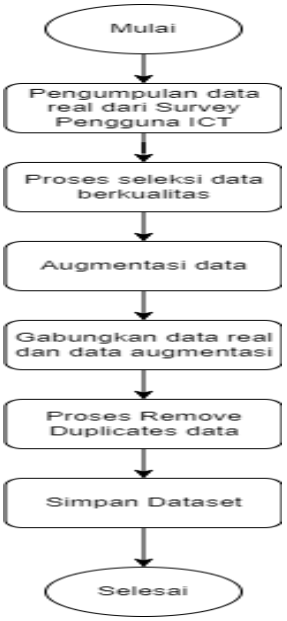
Telegram. Tahap terakhir dari pengembangan ini adalah evaluasi kinerja untuk memastikan bahwa *chatbot* berfungsi dengan baik dalam menjawab pertanyaan pengguna secara efektif.



Sumber: (Hasil Penelitian, 2024)

Gambar 1. Tahapan Penelitian

1. Pengumpulan Dataset



Sumber: (Hasil Penelitian, 2024)

Gambar 2. Proses Pengumpulan Dataset

Data yang digunakan dalam penelitian ini adalah kumpulan pertanyaan dan jawaban yang berkaitan dengan layanan ICT yang dikumpulkan dari hasil survei yang dilakukan terhadap 250 pengguna yang menggunakan layanan ICT, dengan masing masing pengguna dapat menanyakan 2 pertanyaan berbeda. Tidak hanya dari hasil survei, dataset ini juga didapatkan melalui tahapan augmentasi data yang bertujuan untuk mendapatkan banyaknya variasi data pertanyaan yang sudah didapatkan dari hasil survei. *Dataset* ini disusun dalam format Microsoft Excel dan didapatkan *dataset* sejumlah 1717 baris yang terdiri dari 500 baris data *real* dan 1217 baris data hasil augmentasi, dengan dua kolom utama, yaitu kolom *question* dan kolom *answer*. Berikut merupakan flowchart proses pengumpulan data yang dilakukan selama penelitian:

Dari proses Gambar 2. maka mendapatkan sejumlah dataset yang dapat digunakan. Dibawah ini merupakan tabel yang berisi contoh beberapa *dataset* yang akan digunakan:

Tabel 1. Contoh *Dataset* Pertanyaan dan Jawaban

No	Question	Answer
1	Bagaimana cara mengatasi komputer yang lambat?	Bersihkan file sementara, uninstall program yang tidak diperlukan, dan tambahkan RAM jika memungkinkan
2	Apa yang harus dilakukan jika layar komputer tidak menyala?	Periksa sambungan kabel, pastikan monitor menyala, dan coba restart komputer.
3	Bagaimana cara memperbaiki koneksi internet yang lambat?	*Restart router*, periksa penggunaan *bandwidth*, dan pastikan tidak ada perangkat yang menggunakan jaringan secara berlebihan.
4	Apa yang harus dilakukan jika lupa *password*?	Gunakan fitur "Lupa *Password*" pada layanan yang bersangkutan atau hubungi administrator sistem.
5	Mengapa printer tidak bisa mencetak?	Periksa koneksi kabel, pastikan driver printer sudah terinstal, dan periksa apakah kertas dan tinta mencukupi.
6	Bagaimana cara mengatasi virus komputer?	Jalankan *software* antivirus, lakukan pemindaian penuh, dan hapus file yang terinfeksi.
7	Mengapa komputer sering restart sendiri?	Periksa suhu komputer, pastikan tidak ada perangkat keras yang bermasalah, dan scan komputer untuk virus.
8	Apa yang harus dilakukan jika email tidak bisa dikirim?	Periksa koneksi internet, pastikan alamat email tujuan benar, dan cek kuota email jika menggunakan layanan terbatas.
9	Bagaimana cara backup data di komputer?	Gunakan *software* backup, simpan di hard drive eksternal atau cloud storage.
10	Apa yang harus dilakukan jika aplikasi sering *crash?*	*Update* aplikasi ke versi terbaru, periksa kompatibilitas sistem, dan lakukan *reinstall* jika perlu.
11	Bagaimana cara	Matikan komputer, gunakan kompresor udara untuk

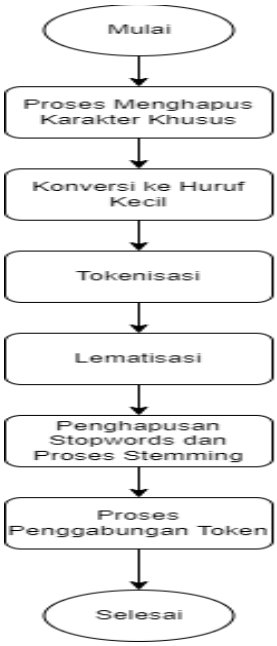
VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

No	Question	Answer
	membersihkan *keyboard* yang kotor?	membersihkan debu, dan lap dengan kain *microfiber*.
12	Mengapa laptop cepat panas?	Bersihkan ventilasi dari debu, gunakan *cooling pad*, dan pastikan tidak ada aplikasi berat yang berjalan bersamaan.
13	Apa yang harus dilakukan jika tidak bisa masuk ke akun?	Cek kembali *username* dan *password*, gunakan fitur "Lupa *Password*", atau hubungi *support*.
14	Bagaimana cara menginstal ulang sistem operasi?	Backup data penting, siapkan media instalasi (CD/USB), dan ikuti panduan instalasi sistem operasi.
15	Mengapa tidak ada suara dari komputer?	Periksa sambungan kabel audio, pastikan *driver audio* terinstal, dan cek volume pengaturan sistem.

Sumber: (Hasil Penelitian, 2024)

2. Proses Preprocessing Data

Sebelum digunakan dalam model *machine learning*, proses *preprocessing* data bertujuan untuk membersihkan dan mempersiapkan data. Berikut merupakan *flowchart* proses *preprocessing* data yang digunakan pada peneliatian ini:

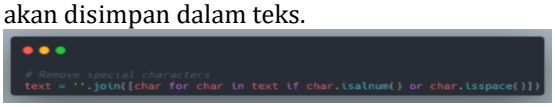


Sumber: (Hasil Penelitian, 2024)

Gambar 3. Proses Preprocessing

Dari *flowchart* diatas, serangkaian proses yang dilalui akan dijelaskan lebih detail pada gambar gambar dibawah ini:

1. Menghapus semua karakter khusus dari teks. Hanya karakter alfanumerik dan spasi yang akan disimpan dalam teks.



Sumber: (Hasil Penelitian, 2024)

Gambar 4. Kode *Remove Special Characters*

Berikut hasil *k*ode *Remove Special Characters*. Tanda tanya (?), tanda hubung (-), tanda baca lain seperti titik (.), koma, atau tanda petik dihilangkan. Hasilnya hanya menyisakan huruf, angka, dan spasi. Teks menjadi lebih bersih dan siap diproses ke tahap berikutnya seperti tokenisasi, lemmatisasi, dan lainnya.

Tabel 2. Hasil Kode *Remove Special Characters*

No	Question	Answer
1	Bagaimana cara mengatasi komputer yang lambat	Bersihkan file sementara uninstall program yang tidak diperlukan dan tambahkan RAM jika memungkinkan
2	Apa yang harus dilakukan jika layar komputer tidak menyala	Periksa sambungan kabel pastikan monitor menyala dan coba restart komputer
3	Bagaimana cara memperbaiki koneksi internet yang lambat	Restart router periksa penggunaan bandwidth dan pastikan tidak ada perangkat yang menggunakan jaringan secara berlebihan

Sumber: (Hasil Penelitian, 2024)

2. Melakukan konversi pada teks menjadi huruf kecil untuk konsistensi data. Kode Convert to *lowercase* ini yaitu untuk melakukan konversi pada teks menjadi huruf kecil untuk konsistensi data.

Sumber: (Hasil Penelitian, 2024)

Gambar 5. *Convert to lowercase*

Berikut hasil proses mengkonversi teks menjadi huruf kecil. Semua teks (baik pada kolom ***Question*** maupun ***Answer***) diubah menjadi huruf kecil. Format ini menciptakan konsistensi dalam teks sehingga siap untuk proses analisis lebih lanjut, seperti tokenisasi atau lemmatisasi.

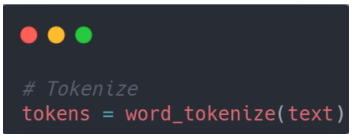
Tabel 3. Hasil *Convert to lowercase*

No	Question	Answer
1	bagaimana cara mengatasi komputer yang lambat	bersihkan file sementara uninstall program yang tidak diperlukan dan tambahkan ram jika memungkinkan

No	Question	Answer
2	apa yang harus dilakukan jika layar komputer tidak menyala	periksa sambungan kabel pastikan monitor menyala dan coba restart komputer
3	bagaimana cara memperbaiki koneksi internet yang lambat	restart router periksa penggunaan bandwidth dan pastikan tidak ada perangkat yang menggunakan jaringan secara berlebihan

Sumber: (Hasil Penelitian, 2024)

3. Memecah teks menjadi token (kata-kata individu) menggunakan fungsi `*word_tokenize*`. Tokenisasi merupakan proses memisahkan teks menjadi unit-unit yang lebih kecil.



Sumber: (Hasil Penelitian, 2024)
Gambar 6. Kode *Word Tokenize*

Berikut hasil proses tokenisasi Setiap kalimat pada *Question* dan *Answer* dipecah menjadi kata-kata (token). Proses ini memungkinkan pemisahan kata-kata secara individual, yang lebih mudah diproses dalam analisis machine learning. Penghilangan Karakter Spasi Berlebihan: Tokenisasi otomatis menghilangkan spasi berlebihan yang tidak relevan antara kata-kata.

Tabel 4. Hasil Tokenisasi

No	Question Tokens	Answer Tokens
1	[bagaimana, cara, mengatasi, komputer, yang, lambat]	[bersihkan, file, sementara, uninstall, program, yang, tidak, diperlukan, dan, tambahkan, ram, jika, memungkinkan]
2	[apa, yang, harus, dilakukan, jika, layar, komputer, tidak, menyala]	[periksa, sambungan, kabel, pastikan, monitor, menyala, dan, coba, restart, komputer]
3	[bagaimana, cara, memperbaiki, koneksi, internet, yang, lambat]	[restart, router, periksa, penggunaan, bandwidth, dan, pastikan, tidak, ada, perangkat, yang, menggunakan, jaringan, secara, berlebihan]

Sumber: (Hasil Penelitian, 2024)

4. Melakukan proses lematisasi pada setiap token. Lematisasi adalah proses mengubah kata-kata menjadi bentuk dasarnya (lemma). Misalnya, kata "*running*" diubah menjadi "*run*". Hanya token yang merupakan kata alfanumerik yang diolah.

Sumber: (Hasil Penelitian, 2024)
Gambar 7. Kode *Lemmatize*

Berikut hasil Lematisasi Kata seperti "menyala" tetap menjadi "menyala", tetapi kata seperti "mencetak" diubah menjadi "cetak", "menginstal" menjadi "instal", dan "terinstal" menjadi "instal". Kata yang memiliki bentuk kata dasar yang lebih sederhana akan dikembalikan ke bentuk lemma-nya. Misalnya, "memperbaiki" menjadi "perbaiki", "mengecek" menjadi "cek".

Tabel 5. Hasil Lematisasi

No	Question Tokens (Lemmatized)	Answer Tokens (Lemmatized)
1	[bagaimana, cara, mengatasi, komputer, yang, lambat]	[bersihkan, file, sementara, uninstall, program, yang, tidak, diperlukan, dan, tambahkan, ram, jika, memungkinkan]
2	[apa, yang, harus, lakukan, jika, layar, komputer, tidak, menyala]	[periksa, sambungan, kabel, pastikan, monitor, menyala, dan, coba, restart, komputer]
3	[bagaimana, cara, memperbaiki, koneksi, internet, yang, lambat]	[restart, router, periksa, penggunaan, bandwidth, dan, pastikan, tidak, ada, perangkat, yang, gunakan, jaringan, secara, berlebihan]

Sumber: (Hasil Penelitian, 2024)

5. Menghapus *stopwords* dan melakukan *stemming* pada sisa token.

Sumber: (Hasil Penelitian, 2024)
Gambar 8. Kode *Remove Stopwords and Stem*

Berikut adalah hasil setelah menghapus *stopwords* dan melakukan *stemming* pada token yang telah dilakukan lematisasi. *Stopwords* adalah

VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

kata-kata yang umum digunakan namun tidak memberikan banyak informasi untuk analisis, seperti "dan", "yang", "di", "untuk", dll. *Stemming* mengubah kata menjadi bentuk dasarnya, misalnya "bersihkan" menjadi "bersih" atau "memperbaiki" menjadi "perbaiki".

Tabel 6. Hasil *Stopwords* dan *Stemming*

No	Question Tokens (Stopwords Removed and Stemmed)	Answer Tokens (Stopwords Removed and Stemmed)
1	[bagaimana, cara, atasi, komputer, lambat]	[bersih, file, sementara, uninstall, program, tidak, perlukan, tambah, ram, memungkinkan]
2	[apa, harus, lakukan, layar, komputer, tidak, nyala]	[periksa, sambung, kabel, pastikan, monitor, nyala, coba, restart, komputer]
3	[bagaimana, cara, perbaiki, koneksi, internet, lambat]	[restart, router, periksa, penggunaan, bandwidth, pastikan, perangkat, gunakan, jaringan, berlebihan]

Sumber: (Hasil Penelitian, 2024)

6. Menggabungkan token yang sudah diproses menjadi satu string.

```
return ' '.join(stemmed_tokens)
```

Sumber: (Hasil Penelitian, 2024)
Gambar 9. Hasil *Join Stemmed*

Berikut hasil penggabungan token yang telah diproses (setelah penghapusan *stopwords* dan stemming) digabungkan kembali menjadi satu string. Hasilnya adalah kalimat yang lebih bersih, dengan kata-kata yang relevan, tanpa karakter khusus, stopwords, atau bentuk kata yang tidak diperlukan.

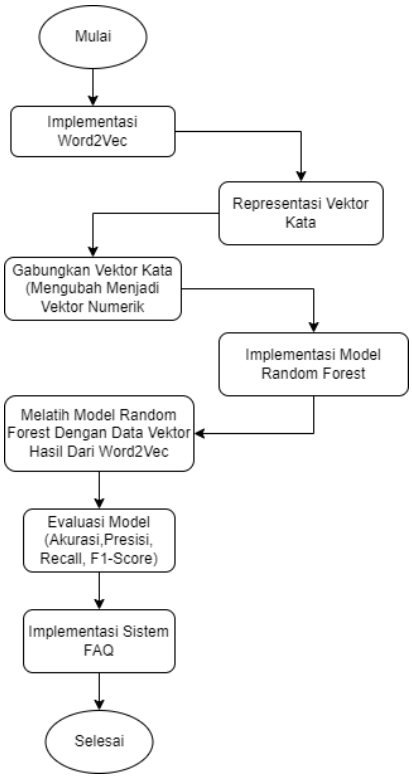
Tabel 7. Hasil Join *Stemmed*

No	Question Tokens (Stopwords Removed and Stemmed)	Processed Answer (Combined Tokens)
1	bagaimana cara atasi komputer lambat	bersih file sementara uninstall program tidak perlukan tambah ram memungkinkan
2	apa harus lakukan layar komputer tidak nyala	periksa sambung kabel pastikan monitor nyala coba restart komputer
3	bagaimana cara perbaiki koneksi internet lambat	restart router periksa penggunaan bandwidth pastikan perangkat gunakan jaringan berlebihan

Sumber: (Hasil Penelitian, 2024)

HASIL DAN PEMBAHASAN

Penelitian ini bertujuan untuk mengembangkan *chatbot* berbasis Telegram yang menggunakan algoritma *Random Forest* dan metode *Word2Vec* untuk menjawab pertanyaan terkait layanan ICT. Pengembangan *chatbot* ini dimulai dengan mengumpulkan *dataset* berisi pertanyaan dan jawaban dari FAQ (*Frequently Asked Questions*) layanan ICT. Algoritma *Random Forest* diterapkan untuk melakukan klasifikasi terhadap pertanyaan yang diterima. Sementara itu, metode *Word2Vec* digunakan untuk membangun representasi vektor dari kata-kata dalam pertanyaan dan jawaban tersebut. Berikut merupakan gambaran *flowchart* dari tahapan implementasi *Word2Vec* dan *Random Forest*:

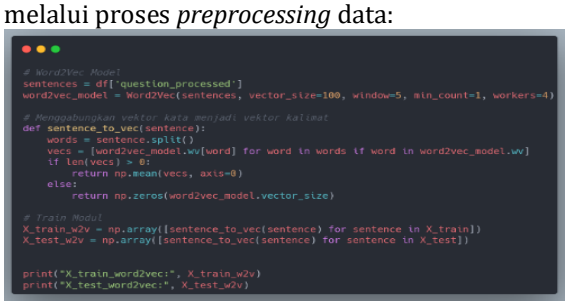


Sumber: (Hasil Penelitian, 2024)
Gambar 10. Implementasi *Word2Vec* dan *Random Forest*

VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri
INTI NUSA MANDIRI
DOI: https://doi.org/10.33480/inti.v20i1.5766

1. Implementasi *Word2Vec* untuk *Word Embedding*

Model *Word2Vec* digunakan untuk merepresentasikan teks dalam bentuk vektor yang dapat diproses oleh model pembelajaran mesin. Dalam penelitian ini, kami menggunakan metode *Continuous Bag of Words (CBOW)*. Berikut merupakan gambar implementasi *Word2Vec* pada data pertanyaan yang sudah melalui proses *preprocessing* data:



Sumber: (Hasil Penelitian, 2024)

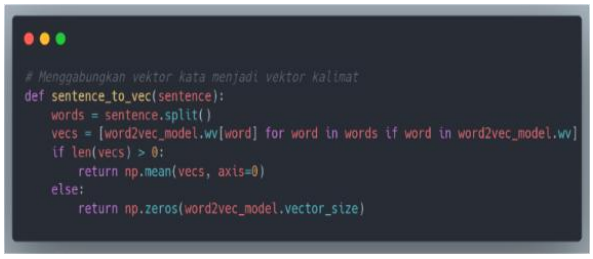
Gambar 11. Kode Implementasi *Word2Vec*

Dari gambar diatas terdapat beberapa proses yang dilakukan pada saat implementasi *Word2Vec*, berikut penjelasannya:

Sumber: (Hasil Penelitian, 2024)

Gambar 12. Kode *Word2Vec* Model

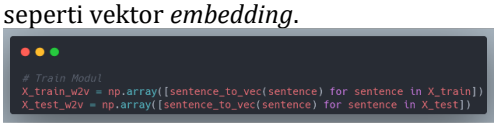
1. Pada gambar diatas terdapat proses pembuatan model *Word2Vec* menggunakan *Word2Vec* dari *library* gensim. Parameter yang digunakan adalah:
   a. `Word2Vec(sentences)`: Membuat model *Word2Vec* dari data teks yang telah diproses (question_processed) dalam *DataFrame* df.
   b. `vector_size=100`: Ukuran vektor *embedding* yang akan dihasilkan untuk setiap kata adalah 100 dimensi.
   c. `window=5`: Ukuran jendela konteks kata yang digunakan untuk pelatihan model adalah 3 kata di kiri dan kanan kata target.
   d. `workers=4`: Jumlah *thread* yang digunakan untuk melatih model adalah 4.
   e. `min_count=1`: Hanya kata yang muncul setidaknya sekali yang akan dipertimbangkan untuk pelatihan model.
   f. `sg=0`: Menggunakan *Continuous Bag of Words* (CBOW) sebagai algoritma pelatihan, bukan *Skip-gram*.



Sumber: (Hasil Penelitian, 2024)

Gambar 13. Kode Menggabungkan Vektor

2. Pada gambar diatas terdapat proses menggabungkan vektor kata menjadi vektor kalimat menggunakan model *Word2Vec*. Serangkaian proses yang dilakukan pada tahap ini adalah:
   a. `sentence.split()`: Memecah kalimat menjadi kata-kata.
   b. `[word2vec_model.wv[word] for word in words if word in word2vec_model.wv]`: Mendapatkan vektor *embedding* untuk setiap kata dalam kalimat yang ada di dalam model *Word2Vec*.
   c. `np.mean(vecs, axis=0)`: Menghitung rata-rata vektor kata untuk mendapatkan representasi vektor kalimat.
   d. `np.zeros(word2vec_model.vector_size)`: Jika tidak ada kata dalam kalimat yang ada dalam model *Word2Vec*, mengembalikan vektor nol dengan ukuran yang sama seperti vektor *embedding*.



Sumber: (Hasil Penelitian, 2024)

Gambar 14. Kode *Training* Modul

3. Pada gambar diatas terdapat proses pelatihan dan pengujian *dataset* yang sudah melalui tahapan yang dilakukan pada model *Word2Vec* sebelumnya. Berikut penjelasan dari kodenya:
   a. np.array([sentence_to_vec(sentence) for sentence in X_train]): Mengonversi setiap kalimat dalam X_train menjadi vektor kalimat menggunakan fungsi sentence_to_vec, kemudian mengubahnya menjadi array NumPy.
   b. np.array([sentence_to_vec(sentence) for sentence in X_test]): Melakukan hal yang sama untuk X_test.

Kode ini digunakan untuk menghasilkan representasi vektor dari kalimat-kalimat yang ada dalam *dataset* pelatihan (X_train) dan *dataset* pengujian (X_test) berdasarkan model *Word2Vec* yang telah dilatih. Representasi vektor kalimat ini

VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

kemudian dapat digunakan sebagai input untuk model pembelajaran mesin lainnya.

2. Implementasi Algoritma *Random Forest* untuk Klasifikasi

Algoritma *Random Forest* digunakan untuk mengklasifikasikan pertanyaan ke dalam kategori jawaban yang sesuai. Model ini dipilih karena kemampuannya dalam menangani data teks yang kompleks dan memberikan performa yang baik dalam berbagai aplikasi.

```
# Train RandomForestClassifier
rf_classifier = RandomForestClassifier(n_estimators=150, max_features='log2', random_state=42, min_samples_leaf=1,
min_samples_split=5, bootstrap=False)
rf_classifier.fit(X_train_combined, y_train)
```

Sumber: (Hasil Penelitian, 2024)

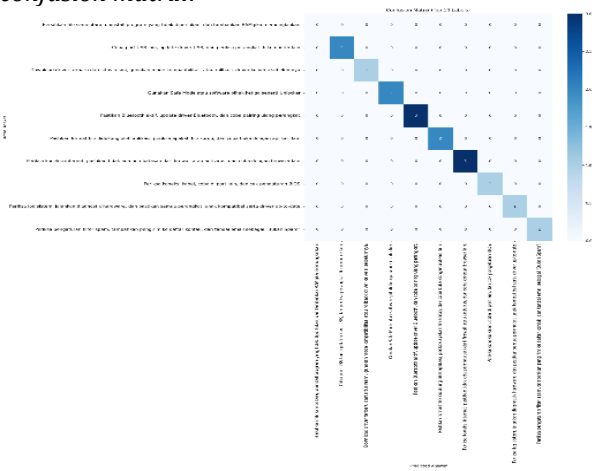
Gambar 15. Kode Implementasi Algoritma *Random Forest*

Pada gambar diatas menunjukkan bagaimana algoritma *Random Forest* dapat digunakan untuk membuat model klasifikasi pada data yang telah diberikan. Setiap baris kode dijelaskan di bawah ini:

1. `rf_classifier=*RandomForestClassifier(n_estimators=150,max_features='log2',random_state=42,min_samples_leaf=1,min_samples_split=5,bootstrap=False)*` : Pada baris ini, kita mengatur parameter n_*estimators* ke 150 untuk menentukan berapa banyak pohon keputusan (*decision trees*) yang akan digunakan dalam hutan acak. Di sini, 150 pohon akan digunakan. Model mungkin lebih lambat untuk dilatih tetapi lebih akurat dengan lebih banyak pohon. Parameter kedua yang diatur yaitu *max_features* menjadi log2 untuk menentukan jumlah fitur maksimum yang akan dipertimbangkan untuk membagi setiap *node*. Dalam hal ini, jumlah maksimum fitur yang dipertimbangkan adalah logaritma basis 2 dari jumlah total fitur. Parameter selanjutnya yang diatur adalah random_state menjadi angka 42. Pada parameter ini menetapkan *seed* yang akan digunakan untuk pengacakan sehingga hasil eksperimen dapat direproduksi. Lalu ada parameter *min_samples_leaf* yang akan ditetapkan menjadi angka 1, kegunaannya untuk menentukan jumlah sampel minimun yang harus ada di daun dari setiap pohon keputusan. Dalam hal ini, setiap daun harus memiliki setidaknya satu sampel. Parameter kelima yang diatur yaitu *min_samples_split* menjadi angka 5, ini digunakan untuk menentukan jumlah sampel minimum yang diperlukan untuk membagi *node*. Karena diatur menjadi angka 5 maka node hanya akan dibagi jika ada minimal 5 sampel. Untuk parameter terakhir ada *bootstrap* yang akan diatur menjadi *False.* Parameter ini menentukan penggunaan *bootstrap sampling* saat membuat keputusan. Diatur menjadi *False* agar seluruh *dataset* akan digunakan untuk membangun setiap pohon (tanpa pengambilan sampel ulang).
2. `*rf_classifier.fit(X_train_combined, y_train)*`: Dalam baris ini, kami melatih model dengan data pelatihan *(X_train_combined)* dan labelnya *(y_train)*. Kami menggunakan metode fit dari objek *Random Forest Classifier*. Model akan dibuat sebagai hasil dari proses pelatihan ini dan dapat digunakan untuk melakukan prediksi.
3. *y_pred = rf_classifier.predict(X_test_combined)*: Kami menggunakan model yang telah dilatih untuk melakukan prediksi pada data *test (X_test_combined)*. Hasil prediksi akan disimpan pada variabel y_pred yang akan digunakan untuk mengukur metrik-metrik seperti akurasi, presisi, *recall* dan *F1-Score*.

3. Evaluasi Kinerja Model

Evaluasi model dilakukan dengan mengukur metrik-metrik seperti akurasi, presisi, *recall* dan *F1-Score.* Tahapan pengukurannya akan menggunakan *confusion matrix* untuk mendapatkan visualisasi nilai metrik yang akan digunakan untuk menghitung akurasi, presisi, *recall* dan *F1-Score.* Berikut merupakan gambar yang menunjukan *confusion matrix*:



Sumber: (Hasil Penelitian, 2024)

Gambar 16. Visualisasi *Confusion Matrix*

Pada gambar diatas matriks menampilkan 10 label teratas untuk menjaga kejelasan visual dengan

sebagian besar prediksi berada di diagonal utama yang menunjukkan bahwa prediksinya akurat. Nilai prediksi maksimum untuk beberapa label mencapai angka 3 dengan jumlah angka di luar diagonal menunjukkan hampir tidak ada misklasifikasi yang signifikan. Hal tersebut menunjukkan bahwa model dapat mengklasifikasikan jawaban dengan cukup baik untuk set data yang dianalisis. Karena dataset yang digunakan jumlahnya terbilang sangat banyak, maka untuk penghitungan akurasi, presisi, *recall* dan *F1-Score* akan menggunakan bantuan *library scikit-learn* seperti pada gambar dibawah ini:

```
# Evaluasi model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='micro')
recall = recall_score(y_test, y_pred, average='micro')
f1_score = f1_score(y_test, y_pred, average='micro')

print(f"Akurasi: {accuracy*100:.2f}%")
print(f"Presisi: {precision*100:.2f}%")
print(f"Recall: {recall*100:.2f}%")
print(f"F1-Score: {f1_score*100:.2f}%")
```

Sumber: (Hasil Penelitian, 2024)
Gambar 17. Kode Evaluasi Kinerja Model

Pada gambar tersebut, hasil dari masing-masing metrik evaluasi dihitung menggunakan fungsi-fungsi dari *library scikit-learn*, seperti `*accuracy_score*`,`*precision_score*`,`*recall_score*`, dan `*f1_score*`, dengan menggunakan data prediksi (`*y_pred*`) dan data target yang sebenarnya (`*y_test*`). Hasil evaluasi diperoleh dengan menguji model yang telah dilatih dengan data pengujian kemudian dicetak dalam bentuk persentase dengan dua angka desimal. Hasil evaluasi kinerja model adalah sebagai berikut:

Tabel 8. Metrik Evaluasi Kinerja Model

Metrik	Nilai
Akurasi	91.28%
Presisi	93.56%
Recall	91.28%
F1-Score	91.42%

Sumber: (Hasil Penelitian, 2024)

Pada tabel metrik evaluasi kinerja model diatas menunjukkan bahwa *chatbot* yang dikembangkan memiliki kinerja yang baik dengan akurasi sebesar 91.28%. *F1-Score* sebesar 91.42%, presisi sebesar 93.56%, dan *recall* sebesar 91.28%. Ini menunjukkan bahwa *chatbot* mampu mengenali pertanyaan dengan baik dan memberikan jawaban yang tepat. Beberapa komponen yang mempengaruhi kinerja model adalah sebagai berikut:

1. Kualitas *dataset*: *Dataset* yang digunakan memiliki variasi pertanyaan yang cukup besar dan terstruktur dengan baik, yang memungkinkan model untuk belajar dengan baik.
2. Data *Preprocessing*: Proses *preprocessing* membantu membersihkan data dan meningkatkan kualitas input untuk model.
3. Algoritma dan metode yang digunakan: Kombinasi algoritma *Random Forest* dan metode *Word2Vec* terbukti efektif dalam menangani teks dan memberikan representasi yang baik.

4. Implementasi Model Pada Chatbot

Setelah model dilatih dan dievaluasi, langkah berikutnya adalah menerapkannya sebagai *chatbot* di *platform* Telegram. Proses implementasinya mencakup:

1. Integrasi dengan API Telegram: Pada tahap integrasi ini menggunakan *library* dari python yaitu **python-telegram-bot==13.7**. Untuk mendapatkan beberapa hal yang dibutuhkan untuk pengoperasian *chatbot* seperti gambar dibawah ini:

```
from telegram import Update, Bot
from telegram.ext import Updater, CommandHandler, MessageHandler, Filters,
CallbackContext, ConversationHandler
```

Sumber: (Hasil Penelitian, 2024)
Gambar 18. Kebutuhan Library Telegram

Tahap ini juga terdapat proses untuk menghubungkan *code* python dengan bot yang sudah dibuat pada *BotFather*. *BotFather* ini merupakan bot resmi yang disediakan oleh *platform* telegram untuk membuat dan mengelola bot baru. Saat proses pembuatan bot baru *BotFather* akan menghasilkan token akses yang nantinya dapat digunakan untuk melakukan akses HTTP API dari bot yang sudah dibuat. Sehingga *code python* yang sudah dibuat dapat terhubung langsung dengan bot telegram untuk menjalankan perintah perintah seperti gambar *code* dibawah ini:

Sumber: (Hasil Penelitian, 2024)
Gambar 19. Perintah untuk *Chatbot* Telegram

VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

Sumber: (Hasil Penelitian, 2024)
Gambar 20. Perintah Untuk *Chatbot* Telegram

Sumber: (Hasil Penelitian, 2024)
Gambar 21. Perintah Untuk *Chatbot* Telegram

2. Digambar 19, 20 dan 21 ini terdapat fungsi *handler* Telegram Bot yang digunakan untuk mengirim perintah langsung pada *chatbot* seperti:
   a. Fungsi *handle_inactivity*, fungsi ini akan diaktifkan saat pengguna *chatbot* tidak aktif selama dua menit dan akan mengirimkan pesan untuk menginformasikan bahwa pengguna tidak aktif dan meminta mereka untuk mengakhiri sesi.
   b. Fungsi *start* yang digunakan untuk memulai bot dengan menggunakan perintah "/*start*" dan akan memunculkan pesan balasan yang memberitahukan pada pengguna bahwa bot siap untuk digunakan.
   c. Fungsi *unknown* yang digunakan untuk menangani perintah yang tidak dikenali oleh *chatbot* dan akan menampilkan pesan balasan kepada pengguna bahwa perintah yang diajukan tidak dikenali.
   d. Fungsi *reply to message* yang memiliki banyak kegunaan seperti untuk melakukan pengecekan apakah bot sudah berhasil dijalankan, mengambil teks pesan yang diketik pengguna pada *chatbot* telegram, melakukan *preprocessing* terlebih dahulu dari teks pesan lalu mengubahnya menjadi vektor untuk dapat menghasilkan prediksi jawaban dari pertanyaan yang diajukan dan mengirimnya pada *chatbot* yang sedang diakses oleh pengguna. Di fungsi ini juga terdapat pengecekan ketika bot belum diinisiasi maka akan meminta pengguna untuk melakukan perintah "*start*" terlebih dahulu.
   e. Fungsi *stop,* fungsi ini digunakan untuk mengakhiri sesi penggunaan *chatbot* dan menghentikan *timer* yang sudah diatur untuk validasi ketika pengguna tidak mengajukan pertanyaan kembali selama dua menit.
   f. Fungsi *set_timer*, merupakan fungsi yang digunakan untuk menetapkan timer pada pengguna dalam mengakses *chatbot*.
   g. Fungsi *reset_timer*, fungsi ini digunakan untuk melakukan *reset timer* yang sudah ditetapkan pada setiap pengguna *chatbot*.
   h. Fungsi *show_continue_prompt*, fungsi ini digunakan untuk menampilkan pertanyaan konfirmasi seperti "Apakah Anda ingin melanjutkan bertanya atau berhenti?" beserta button yang bertuliskan "Lanjut Bertanya" dan "Berhenti" ketika chatbot sudah menampilkan hasil jawaban dari pertanyaan yang diajukan oleh pengguna.
   i. Fungsi *button*, merupakan fungsi yang berguna untuk menetapkan aksi yang akan dilakukan oleh kedua button pada fungsi *show_continue_prompt*.
   j. Fungsi *main*, merupakan fungsi utama yang digunakan untuk menjalankan semua *handler* perintah yang sudah dibuat. Pada fungsi ini berisikan beberapa perintah sebagai berikut:
      1) *Updater* adalah objek yang mengelola *polling* dan menangani pembaruan.
      2) *Dispatcher.add_handler* berfungsi untuk menambah perintah /*start*, *reply to message*, dan perintah *unknown* yang sudah dibuat.
      3) start_polling() digunakan untuk memulai *polling* agar mendapatkan pembaruan dari Telegram.
      4) *Idle* berfungsi untuk menjaga bot tetap berjalan hingga dihentikan secara manual.
3. Penerapan Model dalam *Chatbot*: Pada tahapan ini memulai penerapan model pada bot yang telah dilatih untuk memproses pertanyaan dan memberikan prediksi jawaban yang sesuai dengan model *random forest*. Berikut

INTI NUSA MANDIRI
DOI: https://doi.org/10.33480/inti.v20i1.5766

merupakan gambar *code* yang menunjukkan penerapan tersebut:

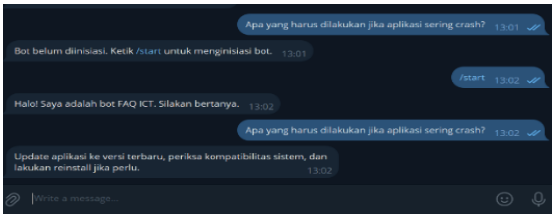
Sumber: (Hasil Penelitian, 2024)
Gambar 22. Penerapan model pada *chatbot*

Terlihat pada gambar 22. Penerapan model pada *chatbot* model *random forest* diterapkan untuk menghasilkan prediksi berdasarkan vektor fitur "*combined_vec*". Hasilnya adalah perkiraan *classifier*, yang dalam hal ini akan menghasilkan jawaban dari pertanyaan yang telah diprediksi oleh model. Setelah hasil prediksi dari model sudah didapatkan maka akan langsung dikirimkan sebagai pesan balasan ke pengguna yang sedang mengakses *chatbot* di telegram.

4. Pengujian Bot: Pada tahap ini bot yang sudah diintegrasikan dengan *coding* python akan dilakukan proses pengujian. Pengujian yang dilakukan yaitu dengan melakukan tes terhadap fungsi *handler* yang telah dibuat dan memasukkan berbagai pertanyaan pada *chatbot* untuk memastikan bahwa *chatbot* dapat memberikan jawaban sesuai dengan *dataset* yang digunakan. Berikut merupakan gambar yang menunjukkan pengujian *chatbot* pada telegram:

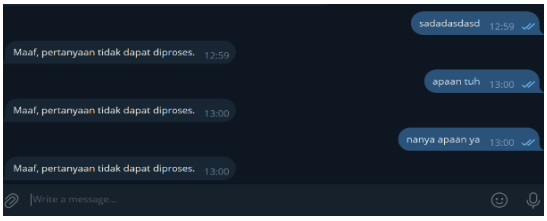
Sumber: (Hasil Penelitian, 2024)
Gambar 23. *Chatbot* Telegram

Pada gambar diatas menunjukkan proses pengujian *chatbot* dengan memasukkan lima pertanyaan untuk membuktikan apakah *chatbot* dapat menjawab semua pertanyaan yang diajukan dengan benar dan tepat sesuai daftar pertanyaan yang terdapat pada *dataset*. Untuk melakukan pengujian terhadap *handler* yang sudah ada akan ditunjukkan dalam gambar berikut:

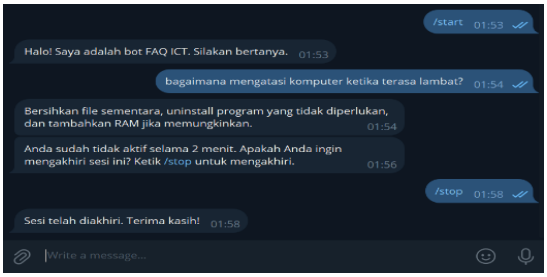


Sumber: (Hasil Penelitian, 2024)
Gambar 24. *Handler Start* dan *Reply Message Chatbot* Telegram

Sumber: (Hasil Penelitian, 2024)
Gambar 25. Pertanyaan Konfirmasi dari *Chatbot*



Sumber: (Hasil Penelitian, 2024)
Gambar 26. *Handler Unknown Chatbot* Telegram



Sumber: (Hasil Penelitian, 2024)
Gambar 27. *Handler Stop Chatbot* Telegram

Pada gambar 24, 25, 26, dan 27 ini menunjukkan proses pengujian *handler chatbot. Handler* yang pertama masuk kedalam tahapan *testing* adalah *reply message* dan *start*, ketika *chatbot* dimulai dari awal pengguna tidak dapat langsung memasukkan pertanyaan terlebih dahulu sebelum *handler start* diketik atau ditekan dan akan memunculkan pesan balasan "Bot belum diinisiasi. Ketik /start untuk menginisiasi bot." pada pengguna *chatbot*. Kemudian *handler* selanjutnya yang diuji yaitu *handler* konfirmasi pertanyaan, handler ini akan berjalan setelah pengguna mendapatkan jawaban dari pertanyaan sebelumnya dan akan menampilkan pertanyaan konfirmasi "Apakah ada

VOL. 20. NO. 1 AGUSTUS 2025
P-ISSN: 0216-6933 | E-ISSN: 2685-807X
Diterbitkan Oleh: LPPM Universitas Nusa Mandiri

yang ingin ditanyakan lagi?" jika pengguna memilih "Lanjut Bertanya" maka pengguna dapat bertanya kembali pada bot namun ketika pengguna memilih "Berhenti" maka bot akan mengeksekusi *handler stop* untuk memberhentikan bot. Untuk pengujian *handler* selanjutnya yaitu *handler unknown. Handler* ini akan menampilkan pesan balasan "Maaf, pertanyaan tidak dapat diproses." pada pengguna ketika pertanyaan yang dimasukkan tidak terdapat dalam *dataset*. Setelah *handler unknown* diuji, terdapat *handler* terakhir yang akan diuji yaitu *handler stop, handler* ini akan muncul bersamaan dengan pesan balasan "Anda sudah tidak aktif selama 2 menit. Apakah Anda ingin mengakhiri sesi ini? Ketik /stop untuk mengakhiri." yang dikirimkan pada pengguna *chatbot* untuk memberitahukan pengguna saat sedang tidak aktif bertanya kembali pada *chatbot* selama dua menit. Selain muncul bersamaan dengan pesan tersebut, *handler* ini dapat juga dieksekusi dengan mengetik perintah "/stop" untuk mengakhiri sesi penggunaan *chatbot* yang akan menampilkan pesan balasan "Sesi telah diakhiri. Terima kasih!" pada pengguna. Dari hasil beberapa pengujian diatas menunjukkan bahwa *chatbot* yang dikembangkan memiliki kemampuan untuk menjawab pertanyaan FAQ dengan tepat dan *handler* yang sudah dibuat untuk melakukan perintah pada *chatbot* berjalan dengan baik.

KESIMPULAN

Dengan menggunakan algoritma *Random Forest* dan metode *Word2Vec*, pengembangan *chatbot* Telegram untuk layanan FAQ ICT berhasil, di mana *chatbot* mampu memberikan jawaban yang akurat dan relevan terhadap pertanyaan yang sering diajukan pengguna. Kombinasi algoritma *Random Forest* dan *Word2Vec* terbukti efektif dalam meningkatkan akurasi, presisi, *recall*, dan *F1-Score*, menghasilkan akurasi sebesar 91.28%, presisi 93.56%, *recall* 91.28%, dan *F1-Score* 91.42%. Algoritma *Random Forest* menawarkan kemampuan klasifikasi yang kuat, sedangkan metode *Word2Vec* menyediakan representasi vektor kata yang lebih baik. Proses *preprocessing* data, seperti lemmatisasi dan *stemming* dalam bahasa Indonesia, konversi teks menjadi huruf kecil, penghapusan karakter khusus, dan *stopwords*, sangat meningkatkan kinerja model dan kualitas *dataset* yang digunakan dalam penelitian. Model yang diterapkan di platform Telegram memungkinkan *chatbot* berinteraksi dengan pengguna secara cepat dan efisien, sehingga meningkatkan efisiensi layanan ICT.

REFERENSI

Af'idah, D. I., Dairoh, D., Handayani, S. F., & Pratiwi, R. W. (2021). Pengaruh Parameter Word2Vec terhadap Performa Deep Learning pada Klasifikasi Sentimen. *Jurnal Informatika: Jurnal Pengembangan IT*, *6*(3), 156–161. https://doi.org/10.30591/jpit.v6i3.3016

Alvi Hasanah, N., Nanik Suciati, & Diana Purwitasari. (2021). Pemantauan Perhatian Publik terhadap Pandemi COVID-19 melalui Klasifikasi Teks dengan Deep Learning. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(1), 193–202. https://doi.org/10.29207/resti.v5i1.2927

Andhika, L. D., Cahyani, D. R., Saputra, D., Herawati, T., Khoiruddinsyah, M., & Saputra, D. D. (2023). Analisis Sentimen Kosumen KFC Berdasarkan Pendekatan Naive Bayes dan Ada Boost Berbasis Data Twitter. *Jurnal INSAN - Journal of Information System Management Innovation*, *3*(1), 55–61. https://doi.org/10.31294/jinsan.v3i1.2219

Angeline, G., Wibawa, A. P., & Pujianto, U. (2022). Klasifikasi Dialek Bahasa Jawa Menggunakan Metode Naives Bayes. *Jurnal Mnemonic*, *5*(2), 103–110. https://doi.org/10.36040/mnemonic.v5i2.4748

Ciptady, K., Harahap, M., Jonvin, J., Ndruru, Y., & Ibadurrahman, I. (2022). Prediksi Kualitas Kopi Dengan Algoritma Random Forest Melalui Pendekatan Data Science. *Data Sciences Indonesia (DSI)*, *2*(1). https://doi.org/10.47709/dsi.v2i1.1708

Dewi, A. R., Diana, S., Fakhrezi, M. A., Awang, N., Ma'arif, H., & Saputra, D. D. (2023). Sentimen Analisis Terhadap Puan Maharani Sebagai Kandidat Calon Presiden 2024 Berdasarkan Opini Twitter Menggunakan Metode Naive Bayes Dan Adaboost. *JSiI (Jurnal Sistem Informasi)*, *10*(1), 75–80. https://doi.org/10.30656/jsii.v10i1.5785

Herlambang, H. P., Saputra, F., Prasetiyo, M. H., Puspitasari, D., & Nurlaela, D. (2023). Perbandingan Klasifikasi Tingkat Penjualan Buah di Supermarket dengan Pendekatan Algoritma Decision Tree, Naive Bayes dan K-Nearest Neighbor. *Jurnal INSAN - Journal of Information System Management Innovation*, *3*(1), 21–28. https://doi.org/10.31294/jinsan.v3i1.2097

Joergensen E Munthe, C., Astuti Hasibuan, N., & Hutabarat, H. (2022). Penerapan Algoritma

Text Mining Dan TF-RF Dalam Menentukan Promo Produk Pada Marketplace. *Resolusi : Rekayasa Teknik Informatika Dan Informasi*, *2*(3), 110–115. https://doi.org/10.30865/resolusi.v2i3.309

Khoiriyah, R., & Afriati, D. (2022). Analisis Proses Bisnis Dan Layanan Penerbit Buku Online Andamari Creative. *Jurnal Simasi: Jurnal Ilmiah Sistem Informasi*, *2*(1), 13–21.

Narendra, L. W. (2022). Topic Modeling in Conversational Dialogs for Naming Intent Labels Using LDA. *JTECS : Jurnal Sistem Telekomunikasi Elektronika Sistem Kontrol Power Sistem Dan Komputer*, *2*(1), 65. https://doi.org/10.32503/jtecs.v2i1.1820

Pratiwi, R. W., H, S. F., Dairoh, D., Af'idah, D. I., A, Q. R., & F, A. G. (2021). Analisis Sentimen Pada Review Skincare Female Daily Menggunakan Metode Support Vector Machine (SVM). *Journal of Informatics, Information System, Software Engineering and Applications (INISTA)*, *4*(1), 40–46. https://doi.org/10.20895/inista.v4i1.387

Puspita, A., & Kalifia, A. D. (2024). *Gudang Jurnal Multidisiplin Ilmu Analisi Depresi Selama Pandemi Covid-19 Menggunakan Algoritma Random Forest*. *2*, 336–338.

Putra, F., Tahiyat, H. F., Ihsan, R. M., Rahmaddeni, R., & Efrizoni, L. (2024). Penerapan Algoritma K-Nearest Neighbor Menggunakan Wrapper Sebagai Preprocessing untuk Penentuan Keterangan Berat Badan Manusia. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *4*(1), 273–281. https://doi.org/10.57152/malcom.v4i1.1085

Sinambela, D. P., Naparin, H., Zulfadhilah, M., & Hidayah, N. (2023). Implementasi Algoritma Decision Tree dan Random Forest dalam Prediksi Perdarahan Pascasalin. *Jurnal Informasi Dan Teknologi*, *5*(3), 58–64. https://doi.org/10.60083/jidt.v5i3.393

Tuasamu, Z., M. Lewaru, N. A. I., Idris, M. R., Syafaat, A. B. N., Faradilla, F., Fadlan, M., Nadiva, P., & Efendi, R. (2023). Analisis Sistem Informasi Akuntansi Siklus Pendapatan Menggunakan DFD Dan Flowchart Pada Bisnis Porobico. *Jurnal Bisnis Manajemen*, *1*(2), 495–510.