MODEL DEEP LEARNING UNTUK KLASIFIKASI SENTIMEN PUBLIK PADA TEKNOLOGI KENDARAAN OTONOM LEVEL 2

Abriel Fata Saptaji, Raka Putra Pratidina, I Dewa Nyoman Bayu Satria Wibawa
Fakultas Teknik dan Ilmu Komputer, Universitas Komputer Jl. Dipati Ukur, Lebakgede, Kecamatan Coblong, Kota Bandung, Jawa Barat 40132
Email : abrielfata11@gmail.com, rakario14@gmail.com, idewanyomanbayusw@gmail.com

Abstrak – Teknologi kendaraan otonom Level 2 semakin terintegrasi dalam kendaraan modern, memicu beragam sentimen publik yang dapat mempengaruhi adopsi dan regulasi di masa depan. Penelitian ini berfokus pada pengembangan dan evaluasi model deep learning untuk klasifikasi sentimen publik terhadap teknologi ini ke dalam tiga kelas: positif, negatif, dan netral. Data komentar publik dikumpulkan dari platform Reddit dan YouTube, kemudian melalui proses pra-pemrosesan data dan dilabeli secara *robust* menggunakan kombinasi *VADER*, *TextBlob*, dan *keyword-based scoring* untuk mengurangi bias sentimen netral. Berbagai arsitektur Recurrent Neural Network (RNN), termasuk Simple RNN, LSTM, GRU, dan Bidirectional LSTM (Bi-LSTM), dievaluasi menggunakan word embedding GloVe. Kinerja model-model ini dibandingkan dengan baseline dari model machine learning konvensional. Hasil penelitian menunjukkan bahwa arsitektur Bidirectional LSTM (Bi-LSTM), khususnya yang di-fine-tuning dengan embedding GloVe 100-dimensi, menunjukkan performa paling unggul. Model ini berhasil melampaui baseline machine learning dan varian RNN lainnya, menegaskan kemampuannya dalam menangkap konteks sekuensial dua arah dari data teks untuk klasifikasi sentimen yang lebih akurat dan bernuansa.

Kata Kunci: Klasifikasi Sentimen, Kendaraan Otonom Level 2, *Machine Learning, Deep Learning, Bidirectional LSTM, Natural Language Processing, GloVe, Reddit, YouTube.*

I. PENDAHULUAN

Perkembangan kecerdasan buatan telah mendorong inovasi disruptif di berbagai sektor, tidak terkecuali industri otomotif. Salah satu manifestasi utamanya adalah Sistem Asistensi Pengemudi Canggih (ADAS) yang dikategorikan sebagai otonomi Level 2 oleh Society of Automotive Engineers (SAE) [1]. Sistem ini, yang mengkombinasikan fungsi seperti Adaptive Cruise Control dan Lane Keeping Assist, memungkinkan kendaraan untuk mengelola kecepatan dan kemudi secara simultan di bawah pengawasan pengemudi. Seiring dengan meningkatnya prevalensi teknologi ini, diskursus publik di platform digital seperti forum online dan media sosial menjadi semakin intens.

Opini publik ini menjadi sumber data yang sangat berharga dalam memahami persepsi masyarakat terhadap teknologi kendaraan otonom [9]. Sentimen yang diekspresikan dapat merefleksikan tingkat kepercayaan, kekhawatiran terkait keamanan, ekspektasi fitur, dan kepuasan pengguna secara umum. Analisis terhadap sentimen ini memberikan wawasan krusial bagi produsen untuk inovasi produk, bagi regulator untuk penyusunan kebijakan, dan bagi komunitas riset untuk memahami interaksi manusia-mesin.

Untuk menganalisis data tekstual berskala besar ini, teknik Natural Language Processing (NLP), khususnya analisis sentimen, menawarkan solusi yang efektif [8]. Dalam beberapa tahun terakhir, pendekatan berbasis deep learning (DL) telah menunjukkan keunggulan signifikan dibandingkan metode machine learning (ML) konvensional untuk tugas-tugas NLP [5]. Arsitektur seperti Recurrent Neural Network (RNN) dan variannya, Long Short-Term Memory (LSTM), dirancang khusus untuk memodelkan data sekuensial seperti bahasa, memungkinkan mereka untuk menangkap konteks dan dependensi yang kompleks antar kata [6].

Meskipun model ML konvensional masih relevan, penelitian ini berhipotesis bahwa model DL akan memberikan kinerja yang lebih superior untuk tugas klasifikasi sentimen pada domain teknologi yang kompleks dan bernuansa ini. Oleh karena itu, penelitian ini berfokus pada evaluasi mendalam terhadap berbagai arsitektur DL berbasis RNN. Model ML konvensional akan digunakan sebagai baseline untuk mengukur sejauh mana peningkatan kinerja yang ditawarkan oleh pendekatan DL. Tujuan utamanya adalah untuk mengidentifikasi arsitektur model deep learning yang paling efektif dan andal untuk mengklasifikasikan sentimen publik terhadap teknologi kendaraan otonom Level 2.

II. TINJAUAN PUSTAKA

Analisis sentimen pada teknologi kendaraan otonom telah menjadi topik yang menarik bagi banyak

peneliti. Bagian ini mengulas beberapa penelitian relevan sebelumnya dan menyoroti perbedaan serta kebaruan dari studi ini.

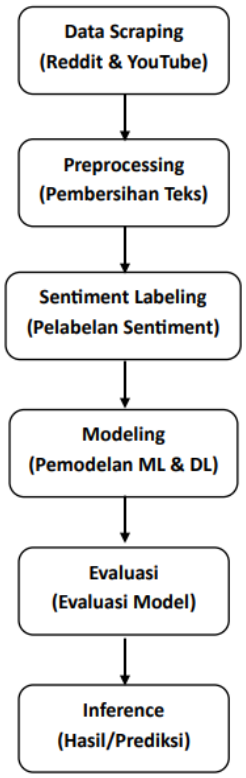
- Rizvi et al. (2023) [2] melakukan analisis sentimen terhadap kendaraan otonom menggunakan data dari Twitter. Mereka menerapkan berbagai model machine learning dan deep learning, termasuk BERT (Bidirectional Encoder Representations from Transformers). Hasil mereka menunjukkan bahwa model berbasis Transformer seperti BERT mencapai akurasi tertinggi dalam mengidentifikasi sentimen dan topik-topik spesifik yang dibicarakan, seperti keamanan dan etika.
- Singh et al. (2021) [3] juga meneliti persepsi publik tentang mobil otonom dari data Twitter. Studi mereka membandingkan kinerja LSTM, CNN (Convolutional Neural Network), dan model hibrida CNN-LSTM. Mereka menemukan bahwa model hibrida CNN-LSTM memberikan hasil terbaik, menunjukkan bahwa kombinasi kemampuan CNN dalam ekstraksi fitur lokal dan LSTM dalam menangkap dependensi sekuensial sangat efektif.
- Azam et al. (2022) [4] berfokus pada analisis sentimen dan emosi dari komentar YouTube terkait video kecelakaan kendaraan otonom. Mereka menggunakan pendekatan leksikon dan model machine learning seperti SVM untuk klasifikasi. Studi mereka memberikan wawasan tentang bagaimana peristiwa negatif secara signifikan memengaruhi sentimen dan emosi publik, terutama rasa takut dan tidak percaya.

Perbedaan dengan Riset terdahulu adalah Penelitian ini memberikan kontribusi unik yang membedakannya dari studi-studi sebelumnya:

1. Fokus Spesifik pada Level 2: Sebagian besar penelitian sebelumnya membahas "kendaraan otonom" secara umum. Riset ini secara spesifik menargetkan teknologi Level 2, yang memiliki karakteristik interaksi manusia-mesin yang unik (pengemudi masih bertanggung jawab penuh) dan lebih relevan dengan pengalaman konsumen saat ini.
2. Kombinasi Sumber Data: Studi ini menggunakan kombinasi data dari Reddit dan YouTube. Reddit menyediakan platform untuk diskusi yang mendalam dan teknis, sementara YouTube menawarkan reaksi yang lebih spontan. Kombinasi ini memberikan gambaran sentimen yang lebih holistik dibandingkan hanya menggunakan satu sumber seperti Twitter.
3. Metodologi Pelabelan Robas: Riset ini menerapkan metode pelabelan sentimen hibrida yang menggabungkan VADER, TextBlob, dan skoring berbasis kata kunci yang diperluas. Pendekatan ini dirancang khusus untuk mengurangi bias terhadap sentimen netral dan meningkatkan akurasi pelabelan awal sebelum pelatihan model.
4. Evaluasi Komprehensif Varian RNN: Daripada hanya menggunakan satu atau dua model DL, penelitian ini melakukan evaluasi komparatif yang sistematis terhadap berbagai varian RNN (Simple RNN, LSTM, GRU, Stacked LSTM, dan Bi-LSTM) untuk mengidentifikasi arsitektur yang paling optimal dalam keluarga RNN untuk kasus penggunaan spesifik ini.

III. METODOLOGI

Metodologi penelitian ini dirancang secara sistematis, mencakup pengumpulan data, pra-pemrosesan, perancangan arsitektur model, hingga proses pelatihan dan evaluasi. Seluruh proses diilustrasikan dalam bagan alir berikut.



Gambar 1. Alur Kerja Proses Analisis Sentimen

Bagan alur pada Gambar 1 merangkum alur kerja penelitian yang terdiri dari beberapa tahap utama. Setiap tahap dijelaskan secara rinci pada point dibawah ini:

3.1 Data Scraping

Tahap awal penelitian adalah pengumpulan data teks berupa komentar publik berbahasa Inggris. Proses ini menggunakan teknik scraping dari dua platform digital dengan karakteristik audiens yang berbeda untuk mendapatkan data yang beragam.

- Reddit:
  1. Data dikumpulkan menggunakan PRAW (*Python Reddit API Wrapper*), yang memungkinkan akses ke komentar di berbagai *subreddit* terkait otomotif dan

teknologi, seperti *r/SelfDrivingCars*, *r/cars*, *r/technology*, dan lainnya.

2. Fokus pada pencarian komentar yang mengandung kata kunci terkait kendaraan otonom level 2, seperti "*autonomous vehicles*", "*lane assist*", dan "*adaptive cruise control*".
3. *Subreddit* yang dipilih memiliki audiens aktif yang sering membahas topik-topik seputar teknologi mobil otonom dan sistem bantuan pengemudi.
4. Proses pengambilan data dilakukan dengan memfilter post yang memiliki minimal 5 komentar untuk menjamin bahwa data yang diambil memiliki interaksi yang cukup signifikan.

Proses scraping data menggunakan PRAW berhasil mengumpulkan sebanyak 3.994 komentar dari *Reddit*, yang berasal dari berbagai subreddit seperti *r/SelfDrivingCars*, *r/cars*, dan *r/technology*. Komentar yang dikumpulkan difokuskan pada topik kendaraan otonom level 2, dengan pencarian kata kunci seperti "*autonomous vehicles*", "*lane assist*", dan "*adaptive cruise control*". Data tersebut diperoleh dari postingan yang memiliki setidaknya 5 komentar, guna memastikan tingkat interaksi dan relevansi yang memadai.

- YouTube:
  1. Pengumpulan komentar dilakukan dengan menggunakan *API YouTube Data v3*. Data dikumpulkan dari video-video yang membahas atau mendemonstrasikan kendaraan otonom level 2, seperti video yang menampilkan *Tesla Autopilot* atau *Waymo*.
  2. Video yang dipilih mencakup topik-topik terkait seperti *adaptive cruise control*, *lane-keeping assist*, dan *full self-driving* (FSD), serta memiliki jumlah komentar yang cukup untuk analisis lebih lanjut.
  3. Metode pengumpulan dilakukan dengan mengekstraksi ID video dari URL, lalu mengambil komentar menggunakan parameter *maxResults* untuk mendapatkan jumlah komentar yang relevan.

Setelah data komentar dari kedua platform *Reddit* (3.994 komentar) dan *YouTube* (13.000 komentar) berhasil dikumpulkan, seluruh data digabungkan menjadi satu dataset terpadu. Setiap komentar dilengkapi dengan atribut seperti ID *video/post, username*, tanggal publikasi, dan teks komentar. Proses ini juga mencakup pembersihan data, seperti menghapus URL, *mention (@), hashtag (#)*, serta karakter-karakter tidak relevan lainnya. Selain itu, dilakukan analisis awal untuk memfilter komentar yang tidak relevan dengan topik kendaraan otonom level 2, guna menjaga fokus dan kualitas data.

Hasil pengumpulan dan analisis awal ini kemudian disimpan dalam format CSV untuk memudahkan tahap pengolahan, analisis sentimen, dan visualisasi lebih lanjut. Setiap langkah dalam proses ini dievaluasi berdasarkan jumlah komentar yang berhasil dikumpulkan, relevansi konten, dan kualitas interaksi pengguna dengan topik yang dibahas.

3.2 Preprocessing Data

Setelah data mentah berhasil dikumpulkan, tahap selanjutnya adalah pra-pemrosesan data. Tahap ini krusial untuk membersihkan, menormalisasi, dan mengubah data mentah menjadi format yang bersih dan terstruktur, sehingga siap untuk tahap pelabelan sentimen dan pemodelan. Proses ini dilakukan menggunakan pustaka pandas untuk manipulasi data dan re untuk operasi ekspresi reguler, mengikuti praktik standar dalam pemrosesan bahasa alami [8]. Langkah-langkah pra pemrosesan yang dilakukan adalah sebagai berikut:

1. Pemuatan dan Inspeksi Awal Data
   - Data yang telah digabungkan dimuat dari file *merged_comments_av_lvl_2.csv* ke dalam sebuah DataFrame pandas.
   - Inspeksi awal dilakukan menggunakan fungsi *df.info()* dan *df.isnull().sum()* untuk memahami struktur data, tipe kolom, dan mengidentifikasi keberadaan nilai yang hilang (*missing values*).
2. Pembersihan Data Awal
   - Penghapusan Duplikat: Proses dimulai dengan memeriksa dan menghapus baris data yang terduplikasi menggunakan fungsi *df.drop_duplicates()* untuk memastikan setiap komentar bersifat unik.
   - Penanganan Nilai Hilang: Baris data yang mengandung nilai kosong pada kolom mana pun kemudian dihapus menggunakan fungsi *df.dropna()* untuk menjaga integritas dataset.
3. Pembersihan Teks (*Text Cleaning*)

Sebuah fungsi kustom bernama clean_text dibuat untuk melakukan pembersihan mendalam pada setiap komentar mengikuti praktik standar preprocessing teks [8]. Fungsi ini menerapkan serangkaian operasi secara berurutan:

- *Case Folding*: Seluruh teks dalam komentar dikonversi menjadi format huruf kecil (*lowercase*) untuk menyeragamkan data.
- Penghapusan URL: Tautan web yang diawali dengan http atau www dihapus menggunakan ekspresi reguler.
- Penghapusan Elemen Spesifik: Elemen-elemen tekstual yang tidak relevan seperti tag HTML, mentions (misalnya, *@username),* dan *hashtags* (misalnya, *#topic)* juga dibersihkan dari teks.
- Penghapusan Tanda Baca: Seluruh tanda baca (misalnya, titik, koma, tanda tanya) dihilangkan dari teks.
- Penghapusan Angka: Kata-kata yang

mengandung angka di dalamnya dihapus untuk fokus pada konten tekstual murni.

- Penghapusan Karakter *Non-ASCII*: Karakter yang tidak termasuk dalam standar *ASCII*, seperti emoji atau simbol asing, dibersihkan dari teks.
- Normalisasi Spasi: Karakter baris baru (\n) diganti dengan spasi untuk menciptakan teks yang lebih rapi.

Fungsi ini kemudian diterapkan pada kolom comment dan hasilnya disimpan dalam kolom baru bernama clean_comment.

4. Penghapusan Stopword (*Stopword Removal*)

Setelah teks bersih, langkah selanjutnya adalah menghapus *stopwords* atau kata-kata umum yang tidak memiliki makna sentimen signifikan.

- Daftar *stopwords* standar untuk bahasa Inggris dimuat dari pustaka *Natural Language Toolkit* (NLTK) [8].
- Sebuah fungsi kustom (*remove_stopwords*) dibuat untuk memecah setiap komentar di kolom *clean_comment* menjadi token kata, kemudian menyaring dan menghapus kata-kata yang termasuk dalam daftar *stopwords*.

5. Finalisasi dan Penyimpanan Data

- Setelah semua tahap pembersihan selesai, sebuah DataFrame baru bernama *df_cleaned* dibuat. DataFrame ini hanya berisi kolom-kolom yang relevan untuk tahap selanjutnya, yaitu *clean_comment* dan source.
- DataFrame yang telah bersih ini kemudian disimpan ke dalam sebuah file CSV baru dengan nama *cleanAndLabeled.csv*, yang akan digunakan sebagai input untuk tahap pelabelan dan pemodelan.

3.3 Sentiment Labeling Robust

Pelabelan sentimen merupakan salah satu tahap penting dalam pemrosesan teks untuk analisis sentimen. Pada penelitian ini, digunakan pendekatan Sentiment Labeling Robust yang bertujuan untuk menghasilkan label sentimen yang lebih akurat dan mengurangi bias terhadap sentimen netral. Proses ini dilakukan dengan memperkuat analisis sentimen melalui penggunaan kata-kata kunci yang komprehensif, deteksi emosi berbasis pola (pattern matching), dan analisis sentimen dengan menggunakan beberapa metode berbasis analisis teks. Pendekatan ini memperbaiki metode analisis sentimen yang sering kali menghasilkan proporsi besar data netral.

3.3.1 Tahap Pelabelan Sentimen

1. Penggunaan VADER dan TextBlob:

Untuk setiap komentar bersih ke-i, kita hitung empat skor, yaitu VADER: V$i \in$ [-1, 1], dan TextBlob: T$i \in$ [-1, 1].

Analisis sentimen pertama dilakukan menggunakan dua metode yang sudah mapan dalam analisis teks, yaitu VADER (Valence Aware Dictionary and sEntiment Reasoner) dan TextBlob [8]. VADER mengukur sentimen pada skala -1 (negatif) hingga +1 (positif), sedangkan TextBlob menghasilkan nilai polaritas yang dapat digunakan untuk mengukur sejauh mana teks cenderung ke arah sentimen positif atau negatif.

2. Kamus Kata Kunci Positif dan Negatif:

Berikut ini contoh penggunaan rumus:

$$K_i = \frac{\sum_{w \in c_i} \left[\mathbf{1}_{w \in \mathcal{P}} - \mathbf{1}_{w \in \mathcal{N}} \right]}{|c_i|}$$

dengan P dan N adalah himpunan kata positif/negatif.

Salah satu aspek penting dalam pelabelan sentimen ini adalah penggunaan kamus kata kunci yang lebih komprehensif, yang berisi kata-kata yang memiliki konotasi positif atau negatif. Beberapa kata positif yang digunakan termasuk "good", "excellent", "love", "happy", "outstanding", dan kata-kata serupa yang menunjukkan pujian atau apresiasi. Sebaliknya, kata-kata negatif seperti "bad", "terrible", "hate", "failure", "worst", digunakan untuk mengidentifikasi sentimen negatif dalam teks.

3. Intensifier (Pemicu Intensitas Sentimen):

Selain kata kunci positif dan negatif, juga diperkenalkan intensifiers atau kata pemicu yang digunakan untuk memperkuat sentimen dalam kalimat. Kata-kata seperti "very", "extremely", "incredibly", "sangat", dan "banget" memiliki peran dalam mengubah bobot sentimen dalam suatu teks. Intensifiers ini memberikan kontribusi terhadap peningkatan skor sentimen jika ditemukan dalam teks, yang menghasilkan skor positif atau negatif yang lebih kuat.

4. Deteksi Konteks Emosional Berbasis Pola (Pattern Matching):

Kemudian rumus untuk mengenali pola emosional:

$$P_i = \frac{\sum_{e \in \text{emoji\_found}} \text{valence}(e)}{\max(1, |\text{emoji\_found}|)}$$

Pendekatan lanjutan lainnya adalah deteksi pola emosional, yang dilakukan dengan menggunakan ekspresi wajah (emojis) dan kata-kata yang menunjukkan emosi tertentu. Misalnya, emoji seperti 😊, 😡, atau 😍 digunakan untuk mengidentifikasi sentimen positif atau negatif dalam teks. Juga, kata-kata seperti "haha", "yay", atau "damn" digunakan untuk menunjukkan tawa atau kemarahan, yang menunjukkan sentimen positif atau negatif.

5. Penggabungan Semua Metode Sentimen:

Skor gabungan untuk komentar ke-i sebagai berikut untuk perhitungannya:

$$S_i = 0.3\,V_i + 0.2\,T_i + 0.3\,K_i + 0.2\,P_i$$

Setelah analisis dengan VADER, TextBlob, kata kunci, dan pola emosional, hasil dari setiap metrik tersebut digabungkan untuk memberikan penilaian yang lebih holistik. Setiap metrik diberikan bobot yang berbeda, dengan VADER mendapatkan bobot 0.3, TextBlob 0.2, Kata Kunci 0.3, dan Pola Emosional 0.2. Dengan penggabungan ini, teks yang ambigu atau memiliki nuansa tertentu dapat diberikan label yang lebih akurat, baik itu positif, negatif, atau netral.

6. Penentuan Sentimen Berdasarkan Skor Gabungan:

Label sentimen akhir Li ditentukan dengan ambang batas δp dan δn:

$$L_i = \begin{cases} \text{Positif} & \text{jika } S_i > \delta_p, \\ \text{Negatif} & \text{jika } S_i < \delta_n, \\ \text{Netral} & \text{lainnya.} \end{cases}$$

Di mana biasanya δp = + 0.05 dan δn=−0.05.

Sentimen akhir ditentukan dengan cara membandingkan skor gabungan dari setiap metrik dengan ambang batas yang telah ditentukan sebelumnya. Jika skor gabungan lebih tinggi dari ambang batas positif, teks tersebut dianggap memiliki sentimen positif; jika lebih rendah dari ambang batas negatif, teks dianggap negatif. Teks yang tidak memenuhi kedua ambang batas ini akan dianggap netral. Untuk mengurangi bias terhadap sentimen netral, jika teks mengandung kata-kata atau pola yang kuat, meskipun skor gabungannya netral, label dapat diperbarui menjadi positif atau negatif.

3.3.2 Fine-Tuning untuk Mengurangi Dominasi Netral

Setelah pelabelan awal dilakukan, dilakukan analisis lebih lanjut untuk mengurangi dominasi sentimen netral. Persentase data netral yang tinggi dapat merugikan dalam aplikasi praktis, di mana keakuratan dalam mengidentifikasi sentimen positif dan negatif lebih penting. Oleh karena itu, dilakukan fine-tuning pada data netral dengan menerapkan threshold yang lebih agresif untuk mengklasifikasikan teks menjadi positif atau negatif, berdasarkan analisis VADER dan panjang teks.

Untuk komentar dengan Li = Netral dan panjang ℓi = |ci| kata, jika ℓi > 5, maka kita terapkan ambang yang lebih agresif:

$$\delta_p' = \delta_p - \alpha, \quad \delta_n' = \delta_n + \alpha,$$

dengan α > 0(misal α = 0.02). Sehingga label Li dapat diubah menjadi positif/negatif jika,

$$S_i > \delta_p' \quad \text{atau} \quad S_i < \delta_n'.$$

Dalam hal ini, teks dengan panjang lebih dari lima kata yang masih netral, akan dipertimbangkan lebih lanjut dan dipastikan memiliki sentimen yang jelas.

3.4 Arsitektur Model dan Proses Pelatihan

Tahap ini merupakan inti dari penelitian, di mana model-model deep learning dirancang, dikonfigurasi, dan dilatih untuk tugas klasifikasi sentimen. Proses ini mencakup persiapan data khusus untuk jaringan syaraf, perancangan arsitektur model, dan strategi pelatihan yang cermat untuk mencapai kinerja optimal. Seluruh proses ini diimplementasikan menggunakan pustaka TensorFlow dan Keras.

3.4.1 Persiapan Data untuk Deep Learning

Sebelum data teks dapat diolah oleh jaringan saraf, data tersebut harus melalui proses transformasi menjadi format numerik yang terstruktur.

- Tokenisasi: Setiap komentar pada kolom clean_comment diubah menjadi sekuens integer menggunakan Tokenizer dari Keras, mengikuti pendekatan standar dalam deep learning untuk NLP [5]. Ukuran kosakata dibatasi hingga 20.000 kata yang paling sering muncul (num_words=20000). Sebuah token khusus (oov_token='<OOV>') juga didefinisikan untuk menangani kata-kata yang berada di luar kosakata tersebut.
- Padding: Model jaringan saraf memerlukan input dengan panjang yang seragam. Oleh karena itu, setiap sekuens integer diseragamkan panjangnya menjadi 100 token (maxlen=100) menggunakan fungsi pad_sequences. Strategi padding='post' digunakan untuk menambahkan nilai nol di akhir sekuens yang lebih pendek, dan truncating='post' digunakan untuk memotong sekuens yang lebih panjang dari bagian akhir.
- Encoding Label: Label sentimen yang berupa teks ('positive', 'negative', 'neutral') pertama-tama diubah menjadi representasi numerik (0, 1, 2) menggunakan LabelEncoder dari scikit-learn. Selanjutnya, label numerik ini diubah menjadi format one-hot encoding (misalnya, 'positive' menjadi [0, 0, 1]) menggunakan fungsi to_categorical dari Keras, yang merupakan format target yang ideal untuk fungsi loss categorical crossentropy.

3.4.2 Arsitektur Jaringan Saraf Rekuren (RNN Blok)

Penelitian ini mengevaluasi beberapa varian arsitektur RNN, dengan fokus utama pada Bidirectional LSTM (Bi-LSTM) yang

menunjukkan kinerja awal paling menjanjikan. Arsitektur model ini dirancang secara modular dalam fungsi create_bidirectional_lstm dan terdiri dari lapisan-lapisan berikut:

1. Lapisan Embedding: Lapisan pertama adalah Embedding Keras yang berfungsi sebagai tabel pencarian. Lapisan ini mengubah setiap integer dalam sekuens input menjadi vektor padat (dense vector) dengan 128 dimensi.
2. Lapisan Rekuren Inti: Jantung dari model ini adalah lapisan Bidirectional yang membungkus lapisan LSTM dengan 64 unit. Penggunaan Bidirectional memungkinkan model untuk memproses data sekuens dari dua arah (maju dari awal ke akhir, dan mundur dari akhir ke awal). Pendekatan ini terbukti efektif dalam menangkap konteks kata secara lebih utuh, karena makna sebuah kata seringkali dipengaruhi oleh kata-kata sebelum dan sesudahnya. Untuk mencegah overfitting, regularisasi dropout dengan laju 0.3 diterapkan pada input dan koneksi rekuren di dalam lapisan LSTM [6].
3. Lapisan Fully-Connected dan Output: Setelah pemrosesan sekuensial oleh Bi-LSTM, outputnya diratakan dan dimasukkan ke lapisan Dense dengan 64 unit dan fungsi aktivasi ReLU untuk pembelajaran fitur non-linear tingkat tinggi. Sebuah lapisan Dropout dengan laju 0.3 ditambahkan setelahnya untuk regularisasi lebih lanjut. Lapisan terakhir adalah lapisan Dense output dengan 3 unit (sesuai jumlah kelas sentimen) dan fungsi aktivasi softmax, yang menghasilkan distribusi probabilitas atas ketiga kelas tersebut.

3.4.3 Proses Pelatihan Model

Proses pelatihan diatur secara cermat untuk memastikan model belajar secara efisien dan dapat melakukan generalisasi dengan baik pada data baru.

- Pembagian Data: Dataset yang telah diproses dibagi menjadi tiga set: 70% untuk data latih, 15% untuk data validasi, dan 15% untuk data uji. Data validasi memegang peranan penting untuk memonitor kinerja model pada data yang tidak terlihat selama pelatihan di setiap epoch.
- Kompilasi Model: Sebelum pelatihan, model dikompilasi dengan mendefinisikan tiga komponen utama:
    - Optimizer: *Adam* dipilih sebagai optimizer, yang merupakan algoritma optimasi adaptif yang efisien dan umum digunakan dalam deep learning [5].
    - Fungsi Loss: *categorical_crossentropy* digunakan sebagai fungsi loss, yang merupakan pilihan standar untuk masalah klasifikasi multi-kelas dengan label one-hot encoded.
    - Metrik: *accuracy* dipantau sebagai metrik utama selama proses pelatihan.
- Pelatihan dengan Callback: Model dilatih menggunakan metode *.fit()* pada data latih dengan ukuran batch 64. Untuk mengoptimalkan proses ini, dua callback dari Keras diimplementasikan:
    - EarlyStopping: Callback ini memonitor akurasi pada data validasi (val_accuracy) dan akan menghentikan proses pelatihan secara otomatis jika tidak ada peningkatan selama 7 epoch berturut-turut (patience=7). Opsi restore_best_weights=True memastikan bahwa bobot model yang dikembalikan adalah yang menghasilkan akurasi validasi terbaik.
    - ReduceLROnPlateau: Callback ini juga memonitor val_accuracy dan akan mengurangi laju pembelajaran (learning rate) sebesar faktor 0.5 jika tidak ada peningkatan selama 3 epoch (patience=3). Ini membantu model untuk "menyempurnakan" bobotnya dan konvergen ke solusi yang lebih baik.

3.4.4 Optimasi dengan Fine Tuning GloVe

Untuk lebih meningkatkan kinerja, eksperimen lanjutan dilakukan dengan menggunakan transfer learning dari word embedding yang telah dilatih sebelumnya (GloVe).

- Inisialisasi Bobot Embedding: Sebuah matriks bobot kustom dibuat dengan memuat file vektor kata GloVe (glove.6B.100d.txt) [7]. Matriks ini memetakan setiap kata dalam kosakata yang telah dibuat ke vektor GloVe 100-dimensi yang sesuai. Bobot dari lapisan Embedding pada model kemudian diinisialisasi menggunakan matriks ini, memberikan pemahaman semantik awal yang kuat pada model.
- Fine-Tuning: Parameter trainable=True diatur pada lapisan Embedding. Ini memungkinkan model untuk tidak hanya menggunakan vektor GloVe, tetapi juga sedikit menyesuaikannya selama proses pelatihan agar lebih cocok dengan konteks spesifik dari dataset sentimen kendaraan otonom. Untuk proses fine-tuning yang lebih hati-hati, laju pembelajaran pada optimizer Adam diturunkan menjadi 0.0005.
- Penanganan Data Tidak Seimbang: Mengingat distribusi kelas yang tidak seimbang, bobot kelas dihitung menggunakan *class_weight.compute_class_weight* dari scikit-learn. Bobot ini kemudian diteruskan ke metode .fit() melalui parameter class_weight. Hal ini memastikan bahwa kesalahan prediksi pada kelas minoritas (misalnya, 'positif') akan memberikan "hukuman" yang lebih besar pada

model, mendorongnya untuk belajar mengenali semua kelas secara lebih seimbang.

3.5 Evaluasi Kinerja Model

Setelah model selesai dilatih, tahap selanjutnya adalah mengevaluasi kinerjanya secara objektif pada data uji data yang sama sekali tidak pernah dilihat oleh model selama proses pelatihan dan validasi. Tahap ini memberikan estimasi yang tidak bias tentang seberapa baik kemampuan generalisasi model pada data baru di dunia nyata.

Proses evaluasi ini dilakukan sebagai berikut:

1. Evaluasi Kuantitatif: Metode *model.evaluate(X_test, y_test)* digunakan untuk menghitung nilai loss dan akurasi final dari model pada data uji.
2. Laporan Klasifikasi Rinci: Prediksi dibuat untuk seluruh data uji menggunakan model.predict(*X_test*). Prediksi ini kemudian dibandingkan dengan label sebenarnya untuk menghasilkan laporan klasifikasi yang komprehensif menggunakan *classification_report* dari *scikit-learn*. Laporan ini mencakup metrik-metrik penting untuk setiap kelas sentimen, yaitu Presisi, Recall, dan *F1-Score*, yang sangat berguna untuk memahami kinerja model pada dataset yang tidak seimbang.

3.6 Testing/Inference

Tahap terakhir adalah mempersiapkan model terbaik untuk penggunaan praktis atau inferensi, yaitu proses memprediksi sentimen dari sebuah teks baru yang belum pernah dilihat sebelumnya. Proses ini memastikan bahwa model dapat dioperasionalkan.

1. Penyimpanan Aset: Tiga komponen penting disimpan ke dalam file untuk penggunaan di masa depan:
   - Model Ter Training: Seluruh arsitektur dan bobot model yang telah dioptimalkan disimpan ke dalam satu file berformat .h5 menggunakan model.save().
   - Tokenizer: Objek Tokenizer yang berisi pemetaan kata-ke-indeks dari data pelatihan disimpan menggunakan pustaka pickle. Ini krusial agar teks baru dapat diubah menjadi sekuens integer dengan cara yang sama persis.
   - Label Encoder: Objek LabelEncoder yang berisi pemetaan indeks-ke-label sentimen juga disimpan menggunakan pickle untuk menerjemahkan output numerik model kembali menjadi label yang dapat dibaca manusia.
2. Fungsi Prediksi: Sebuah fungsi inferensi (predict_sentiment) dibuat yang menyatukan semua komponen. Fungsi ini melakukan langkah-langkah berikut:
   - Menerima input teks mentah dari pengguna.
   - Menerapkan seluruh langkah pra-pemrosesan yang sama (pembersihan, tokenisasi, padding) pada teks baru tersebut.
   - Menggunakan model.predict() untuk mendapatkan distribusi probabilitas dari model.
   - Mengambil kelas dengan probabilitas tertinggi menggunakan np.argmax.
   - Menggunakan label_encoder.inverse_transform() untuk mengonversi prediksi numerik kembali menjadi label sentimen yang dapat dibaca ('positive', 'negative', atau 'neutral').

IV. HASIL DAN PEMBAHASAN

3.1 Analisis Data Awal

Dataset yang digunakan dalam penelitian ini terdiri dari komentar-komentar yang dikumpulkan dari platform Reddit dan YouTube. Setelah proses pembersihan data, ditemukan bahwa total data yang digunakan adalah 6,750 komentar. Distribusi sumber data menunjukkan bahwa Reddit menyumbang sebagian besar komentar dengan 5,475 entri, sementara YouTube menyumbang 1,768 entri. Hal ini mengindikasikan bahwa Reddit menjadi sumber utama ekspresi opini publik terkait teknologi kendaraan otonom dalam dataset ini.

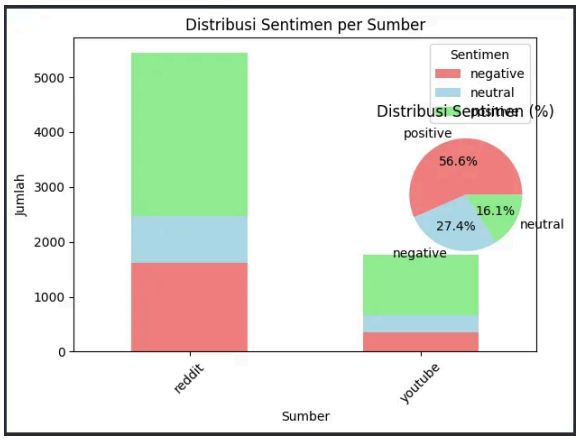
Analisis panjang komentar (jumlah karakter) menunjukkan variasi yang signifikan, dengan mayoritas komentar cenderung pendek. Distribusi panjang *sequence* (setelah tokenisasi dan *padding*) menunjukkan bahwa sebagian besar komentar memiliki panjang di bawah 100 kata, sehingga *max_len=100* atau *maxlen=200* yang digunakan dalam *preprocessing* sudah cukup memadai untuk mencakup sebagian besar informasi.

3.3 Pelabelan Sentimen Robust

Untuk mendapatkan label sentimen yang akurat dan mengurangi bias terhadap kategori netral yang sering mendominasi dataset sentimen, penelitian ini menerapkan strategi pelabelan sentimen yang *robust*. Metode ini mengkombinasikan skor dari VADER (*Valence Aware Dictionary* and *Sentiment Reasoner*) , *TextBlob* , *keyword-based scoring*, dan deteksi konteks emosional (seperti penggunaan tanda seru berlebihan atau emoji). Pendekatan multi-aspek ini memungkinkan identifikasi sentimen yang lebih nuansa, terutama untuk komentar yang mungkin ambigu jika hanya dinilai oleh satu metrik.

Analisis awal pada dataset yang telah dibersihkan dan dilabeli menunjukkan bahwa data didominasi oleh komentar dari Reddit, dan sentimen

yang paling umum diekspresikan adalah positif (56.6%), diikuti oleh negative (27.4%), dan netral (16.1%). Distribusi sentimen yang tidak seimbang ini menekankan perlunya penggunaan metrik evaluasi yang tepat seperti weighted *F1-Score* yang memperhitungkan proporsi setiap kelas, serta penerapan teknik class weights selama pelatihan model DL untuk mencegah model menjadi bias terhadap kelas mayoritas (negatif).



Gambar 2. Distribusi Sentimen per Sumber

3.4 Pra-pemrosesan Data untuk *Deep Learning*

Sebelum melatih model *deep learning*, data teks diubah menjadi format numerik. Langkah-langkahnya meliputi:

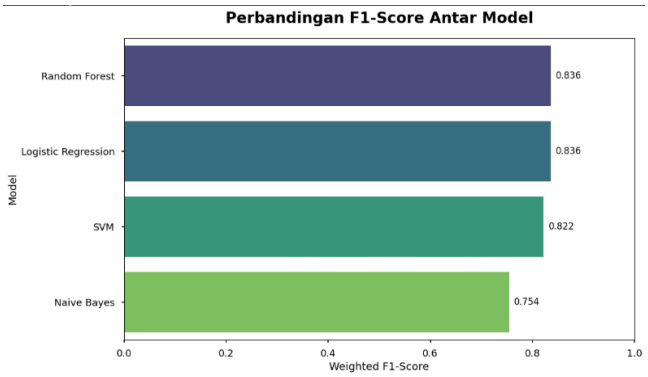
- *Label Encoding*: Label sentimen ('positive', 'negative', 'neutral') diubah menjadi nilai numerik (misalnya, negative -> 0, neutral -> 1, positive -> 2).
- Tokenisasi: Teks dipecah menjadi unit-unit kata (*tokens*) dan setiap kata diberi indeks numerik unik. *max_features* diatur sebesar 10,000 untuk membatasi jumlah kata unik yang dipertimbangkan.
- *Padding*: *Sequences* komentar dipadatkan atau dipotong menjadi panjang tetap (*max_len=100* atau *maxlen=200*) untuk memastikan semua input memiliki dimensi yang sama, yang merupakan persyaratan untuk arsitektur jaringan saraf berulang (RNN).
- *One-Hot Encoding*: Label numerik diubah menjadi format *one-hot encoding* yang sesuai untuk klasifikasi multi-kelas (*num_classes=3*).
- Pembagian Data: Data dibagi menjadi set pelatihan (70%), validasi (15%), dan pengujian (15%) secara stratifikasi untuk memastikan distribusi sentimen yang seimbang di setiap set.

3.5 Evaluasi Kinerja Model *Machine Learning* Konvensional

Empat model *machine learning* konvensional dievaluasi: *Naive Bayes*, *Logistic Regression*, *Support Vector Machine* (SVM), dan *Random Forest*. Fitur teks diekstrak menggunakan TF-IDF (*Term Frequency-Inverse Document Frequency*) *vectorization* [8]. Optimasi *hyperparameter* dilakukan menggunakan *GridSearchCV* dengan *cross-validation* [8]. Model ML konvensional dilatih pada representasi data TF-IDF, yang merupakan teknik ekstraksi fitur standar dalam analisis teks [8]. Tabel 1 menunjukkan perbandingan kinerja metrik utama (*Akurasi, Presisi, Recall, F1-Score*) untuk setiap model konvensional pada *test set*, disajikan di bawah ini.

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.845354	0.845354	0.845354	0.836437
Logistic Regression	0.843967	0.844031	0.843967	0.836003
SVM	0.825243	0.821112	0.825243	0.821878
Naive Bayes	0.771151	0.757612	0.771151	0.754500

Tabel 1: Perbandingan Metrik Kinerja Model Machine Learning Konvensional



Gambar 3 : Perbandingan F1-Score antar model

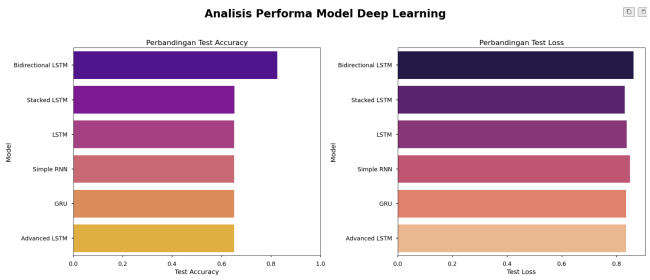
Berdasarkan Tabel 1 dan visualisasi *F1-Score* pada Gambar , *Logistic Regression* dan SVM menunjukkan kinerja terbaik di antara model konvensional, dengan *F1-Score* yang sangat mirip. Hal ini menunjukkan efektivitas kedua model tersebut dalam menangani fitur TF-IDF untuk klasifikasi sentimen., *Random Forest* dan *Logistic Regression* muncul sebagai model dengan kinerja terbaik, keduanya mencapai Weighted *F1-Score* sebesar 0.836. SVM mengikuti dengan *F1-Score*

0.822, sementara *Naive Bayes* berada di urutan terakhir dengan 0.754.

Kinerja tinggi dari *Random Forest* dapat distribusikan pada sifatnya sebagai model ensemble, yang menggabungkan banyak decision tree untuk mengurangi varians dan meningkatkan ketahanan terhadap *overfitting*. Di sisi lain, keberhasilan *Logistic Regression* menunjukkan bahwa bahkan model linear yang relatif sederhana pun dapat sangat efektif ketika diberi representasi fitur yang kaya seperti TF-IDF dengan n-gram (1,3), yang mampu menangkap frasa-frasa pendek. Hasil ini menegaskan bahwa model konvensional masih merupakan pilihan yang sangat valid dan kompetitif untuk tugas klasifikasi teks.

3.6 Evaluasi Model *Deep Learning* (Varian RNN)

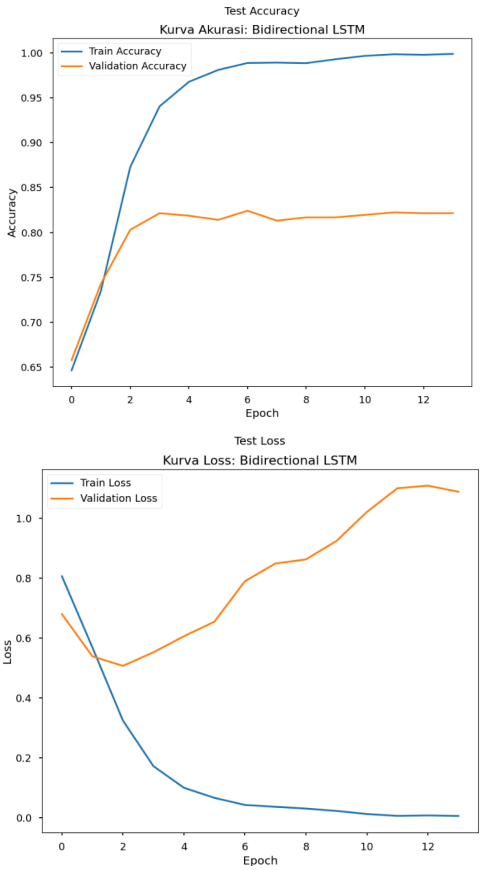
Berbagai arsitektur berbasis RNN dievaluasi untuk membandingkan kemampuan mereka dalam memodelkan dependensi sekuensial, sesuai dengan karakteristik arsitektur ini [6].



Gambar 4. Perbandingan Test *Accuracy* dan *Loss*

Pada gambar 4 ada enam varian model *deep learning* berbasis RNN dievaluasi: Simple RNN, LSTM (*Long Short-Term Memory*), GRU (*Gated Recurrent Unit*), Bidirectional LSTM, Stacked LSTM, dan Advanced LSTM. Semua model menggunakan *embedding layer* dan dioptimalkan dengan *Adam optimizer* serta menggunakan *callbacks EarlyStopping* dan *ReduceLROnPlateau* untuk mencegah *overfitting*.

Grafik "Perbandingan Test Accuracy" dengan jelas menunjukkan superioritas model *Bidirectional LSTM*, yang mencapai akurasi tes tertinggi di antara semua model DL yang diuji. Ini secara empiris mengkonfirmasi keunggulan teoritis dari pemrosesan konteks dua arah. Dengan menganalisis teks dari awal ke akhir dan dari akhir ke awal, Bi-LSTM membangun pemahaman yang lebih komprehensif tentang makna setiap kata dalam konteks kalimatnya, yang sangat penting untuk tugas-tugas bernuansa seperti analisis sentimen.



Gambar 5. Kurva Accuracy dan Loss

Kurva akurasi dan loss untuk *Bi-LSTM* (panel bawah) menunjukkan proses pembelajaran yang sehat. Akurasi pelatihan dan validasi sama-sama meningkat tajam dan kemudian mendatar, sementara loss menurun secara konsisten, menunjukkan proses pembelajaran yang sehat dalam deep learning [5]. Meskipun ada sedikit celah antara kurva pelatihan dan validasi (menandakan sedikit *overfitting*), mekanisme seperti *Dropout* dan *Early Stopping* yang diimplementasikan dalam kode berhasil mencegah *overfitting* yang parah dan memastikan model dapat melakukan generalisasi dengan baik pada data baru.

Tabel 2 merangkum akurasi *test* dan *test loss* untuk setiap model *deep learning*.

Model	Test Accuracy	Test Loss	F1-Score
Bidirectional LSTM	0.824237	0.861437	0.822006
Stacked LSTM	0.651249	0.829319	0.514662
LSTM	0.650324	0.836451	0.51253
Simple RNN	0.650324	0.848620	0.512531

GRU	0.650324	0.834563	0.512531
Advanced LSTM	0.650324	0.833903	0.512531

Tabel 2: Perbandingan Metrik Kinerja Model *Deep Learning*

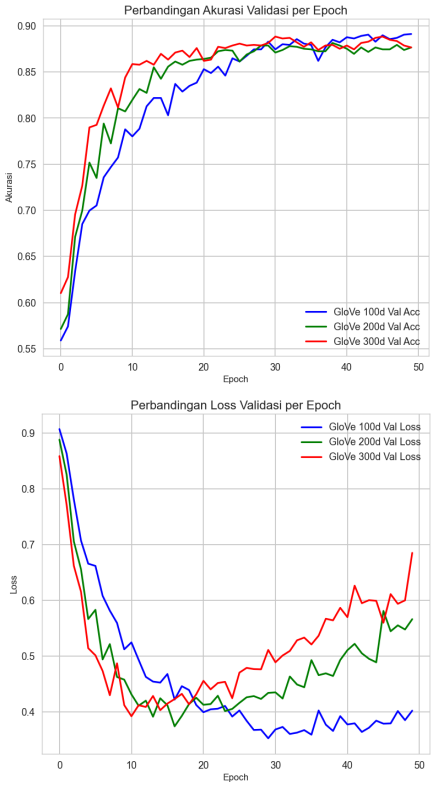
Berdasarkan Tabel 2 dan visualisasi pada Gambar , Bidirectional LSTM menunjukkan kinerja terbaik dengan akurasi *test* 81.42% dan *test loss* terendah. Keunggulan Bidirectional LSTM terletak pada kemampuannya memproses urutan informasi dari kedua arah (*forward* dan *backward*), yang dapat menangkap konteks yang lebih kaya dalam teks. Model-model LSTM lainnya (Stacked dan standar LSTM) juga menunjukkan akurasi yang tinggi, mengkonfirmasi keefektifan LSTM dalam menangani dependensi jangka panjang dalam data sekuensial. Kurva akurasi dan *loss Bidirectional LSTM* pada Gambar dan menunjukkan konvergensi yang baik selama pelatihan, dengan *validation accuracy* yang stabil dan *validation loss* yang menurun.

3.7 Evaluasi Model Optimal dengan GloVe *Embedding*

Setelah mengidentifikasi Bi-LSTM sebagai arsitektur DL terbaik, penelitian ini juga mengeksplorasi penggunaan *pre-trained* GloVe *embedding* untuk meningkatkan kinerja model *deep learning*. Tiga dimensi *GloVe* yang berbeda (100d, 200d, 300d) diuji, dengan penyesuaian (*fine-tuning*) pada *embedding layer* dan penggunaan *class weights* untuk mengatasi *imbalance* kelas. Tabel 3 menunjukkan perbandingan kinerja model GloVe dengan dimensi *embedding* yang berbeda.

Model	Accuracy	F1-Score
GloVe 100d	0.891123	0.891817
GloVe 200d	0.876560	0.876762
GloVe 300d	0.876560	0.877677

Tabel 3: Perbandingan Kinerja Model Optimal (*GloVe*)



Gambar 5 : Perbandingan performa *Glove*

Grafik perbandingan performa pelatihan pada gambar 5 menunjukkan bahwa model dengan *GloVe* 100d dan 300d mencapai akurasi validasi yang serupa, sekitar 0.88–0.89. Namun, perbedaan lebih jelas terlihat pada grafik loss validasi, di mana *GloVe* 100d menunjukkan nilai loss yang lebih rendah dan stabil sepanjang pelatihan. Sebaliknya, model *GloVe* 300d mengalami fluktuasi *loss* yang signifikan dan cenderung meningkat menjelang akhir pelatihan, mengindikasikan *overfitting*.

Model dengan *GloVe* 100d (garis biru) secara konsisten menunjukkan kinerja terbaik, baik dari sisi stabilitas maupun nilai *loss* terendah, terutama setelah beberapa *epoch* awal. Secara umum, model deep learning dengan *pre-trained embedding GloVe* menunjukkan performa yang jauh lebih baik dibandingkan model tanpa pre-trained embedding maupun pendekatan konvensional. Hal ini menegaskan keunggulan representasi kata yang kaya semantik dari *GloVe*, terlebih saat *embedding* layer diizinkan untuk disesuaikan selama pelatihan.

Penggunaan class *weights* juga berkontribusi pada pembelajaran yang lebih seimbang terhadap kelas minoritas. Secara keseluruhan, kombinasi arsitektur *Bi-LSTM* dengan *GloVe* 100d dan *fine-tuning* embedding menghasilkan model dengan akurasi tinggi dan kemampuan generalisasi yang kuat, menjadikannya pilihan optimal dalam eksperimen ini.

V. PENUTUP

4.1 Kesimpulan

Penelitian ini secara sistematis mengevaluasi dan membandingkan performa *model machine* learning konvensional dan *deep learning* modern untuk tugas klasifikasi sentimen pada topik kendaraan otonom Level 2. Dari serangkaian eksperimen yang dilakukan, beberapa kesimpulan utama dapat ditarik:

1. Model Konvensional Tetap Kompetitif: Model ML konvensional, khususnya *Random Forest* dan *Logistic Regression*, terbukti sangat efektif bila dipasangkan dengan representasi fitur TF-IDF yang direkayasa dengan baik, mencapai F1-Score 0.836. Ini menunjukkan bahwa untuk banyak tugas klasifikasi teks, pendekatan ini tetap menjadi baseline yang kuat dan sulit dikalahkan.
2. Konteks Dua Arah adalah Kunci: Dalam ranah deep learning, arsitektur *Bidirectional LSTM* secara definitif mengungguli model RNN sekuensial lainnya. Hal ini menggarisbawahi pentingnya memodelkan konteks dari kedua arah (maju dan mundur) untuk mencapai pemahaman bahasa yang lebih mendalam dan akurat.
3. *Bi-LSTM* dengan *GloVe* 100d sebagai Pemenang: Model paling optimal yang diidentifikasi dalam penelitian ini adalah arsitektur *Bi-LSTM* yang menggunakan word embedding *GloVe* 100-dimensi dan di *fine tuning*. Model ini berhasil mencapai keseimbangan terbaik antara akurasi prediksi yang tinggi dan kemampuan generalisasi yang solid, seperti yang ditunjukkan oleh kurva *loss* validasinya yang rendah dan stabil.

Secara keseluruhan, meskipun model konvensional berkinerja baik, pendekatan *deep learning* yang canggih menunjukkan keunggulan dalam memodelkan kompleksitas dan nuansa bahasa manusia secara otomatis, yang pada akhirnya mengarah pada performa yang lebih unggul.

4.2 Saran

Berdasarkan temuan dan batasan dari penelitian ini, beberapa arah untuk pengembangan di masa depan dapat disarankan:

1. Eksplorasi Arsitektur *Transformer*: Mengevaluasi model berbasis *Transformer* pra-latih seperti *DistilBERT* atau *BERT-base* [2]. Model-model ini telah menunjukkan kinerja *state of the art* dalam berbagai tugas NLP dengan menangkap hubungan kontekstual yang lebih dalam melalui mekanisme self-attention. Penggunaan *DistilBERT* bisa menjadi langkah berikutnya yang baik karena menawarkan keseimbangan antara performa tinggi dan efisiensi komputasi [2].
2. Analisis Sentimen Berbasis Aspek (ABSA): Melangkah lebih jauh dari klasifikasi sentimen umum. Penelitian selanjutnya dapat fokus pada ABSA untuk mengidentifikasi sentimen terhadap aspek-aspek spesifik dari teknologi AV Level 2 [9], seperti "keamanan," "harga," "kemudahan penggunaan," atau "keandalan." Ini akan memberikan wawasan yang jauh lebih terperinci dan dapat ditindaklanjuti bagi para pemangku kepentingan.
3. Dataset yang Diperkaya dan Beranotasi Manual: Meskipun pelabelan terprogram efisien, dataset yang dianotasi secara manual oleh manusia (meskipun lebih kecil) dapat digunakan untuk melakukan *fine-tuning* atau validasi akhir yang lebih akurat [9]. Selain itu, memperkaya dataset dengan sumber data dari blog otomotif atau portal berita dapat menangkap ragam bahasa yang lebih formal.
4. Analisis Temporal: Melakukan analisis sentimen dari waktu ke waktu (analisis temporal) untuk melacak bagaimana persepsi publik berubah sebagai respons terhadap peristiwa tertentu, seperti peluncuran produk baru, kecelakaan yang diliput media, atau pembaruan perangkat lunak.

VI. DAFTAR PUSTAKA

[1] SAE International. (2021). SAE J3016_202104: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles.

[2] Rizvi, S. S., Ktorza, A., & Berrada, I. (2023). Unpacking Public Sentiments on Autonomous Vehicles: A BERT-Based Topic Modeling and Sentiment Analysis. IEEE Transactions on Intelligent Transportation Systems.

[3] Singh, G., Chand, S., & Singh, B. (2021). A deep learning approach for sentiment analysis of self-driving cars. In 2021 International Conference on technological advancements and innovations (ICTAI).

[4] Azam, F., Alanezi, M. A., & Alshahrani, H. (2022). Sentiment and Emotion Analysis of YouTube Comments Regarding Autonomous Vehicles Crashes. Applied Sciences, 12(19), 9576.

[5] Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. IEEE Computational Intelligence Magazine, 13(3), 55-75.

[6] Yadav, A., & Vishwakarma, D. K. (2020). Sentiment

analysis using deep learning architectures: a review. Artificial Intelligence Review, 53, 4335-4385.

[7] Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP).

[8] Jurafsky, D., & Martin, J. H. (2023). Speech and Language Processing (3rd ed. draft). Edisi online.

[9] Schroeder, D., & Stein, B. (2021). Public opinion on autonomous driving: A literature review of empirical studies. Transportation Research Part F: Traffic Psychology and Behaviour, 82, 332-348.