

**PERBANDINGAN MODEL KLASIFIKASI UJARAN
KEBENCIAN DI MEDIA SOSIAL BERBASIS BERT,
XGBOOST, DAN HYBRID BERT-XGBOOST**



SKRIPSI

**ARRAFI AJI PAMUNGKAS
00000064717**

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN INFORMATIKA
UNIVERSITAS MULTIMEDIA NUSANTARA
TANGERANG
2025**

**PERBANDINGAN MODEL KLASIFIKASI UJARAN
KEBENCIAN DI MEDIA SOSIAL BERBASIS BERT,
XGBOOST, DAN HYBRID BERT-XGBOOST**



SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh
Gelar Sarjana Komputer (S.Kom.)

**ARRAFI AJI PAMUNGKAS
00000064717**

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN INFORMATIKA
UNIVERSITAS MULTIMEDIA NUSANTARA
TANGERANG
2025**

HALAMAN PERNYATAAN TIDAK PLAGIAT

Dengan ini saya,

Nama : Arrafi Aji Pamungkas
Nomor Induk Mahasiswa : 00000064717
Program Studi : Informatika

Skripsi dengan judul:

Perbandingan Model Klasifikasi Ujaran Kebencian di Media Sosial Berbasis BERT, XGBoost, dan Hybrid BERT-XGBoost serta Interpretasi Naratif Hasil Menggunakan Generative AI

merupakan hasil karya saya sendiri bukan plagiat dari laporan karya tulis ilmiah yang ditulis oleh orang lain, dan semua sumber, baik yang dikutip maupun dirujuk, telah saya nyatakan dengan benar serta dicantumkan di Daftar Pustaka.

Jika di kemudian hari terbukti ditemukan kecurangan/penyimpangan, baik dalam pelaksanaan maupun dalam penulisan laporan karya tulis ilmiah, saya bersedia menerima konsekuensi dinyatakan TIDAK LULUS untuk mata kuliah yang telah saya tempuh.

Tangerang, 16 Juni 2025



(Arrafi Aji Pamungkas)

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

HALAMAN PENGESAHAN

Skripsi dengan judul

PERBANDINGAN MODEL KLASIFIKASI UJARAN KEBENCIAN DI MEDIA SOSIAL BERBASIS BERT, XGBOOST, DAN HYBRID BERT-XGBOOST

oleh

Nama : Arrafi Aji Pamungkas
NIM : 00000064717
Program Studi : Informatika
Fakultas : Fakultas Teknik dan Informatika

Telah diujikan pada hari Senin, 14 Juli 2025

Pukul 13.00 s/d 15.00 dan dinyatakan

LULUS

Dengan susunan penguji sebagai berikut

Ketua Sidang

Penguji

(Arya Wicaksana, S.Kom., M.Eng.Sc.,
OCA)

NIDN: 0315109103

(Marlinda Vasty Overbeek, S.Kom.,
M.Kom.)

NIDN: 0818038501

Pembimbing

(Aditiyawan, S.Kom., M.Si.)

NIDK: 8994550022

Ketua Program Studi Informatika,

(Arya Wicaksana, S.Kom., M.Eng.Sc., OCA)

NIDN: 0315109103

HALAMAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini:

Nama : Arrafi Aji Pamungkas
NIM : 00000064717
Program Studi : Informatika
Jenjang : S1
Judul Karya Ilmiah : Perbandingan Model Klasifikasi
Ujaran Kebencian di Media Sosial
Berdasarkan BERT, XGBoost, dan Hybrid
BERT-XGBoost

Menyatakan dengan sesungguhnya bahwa saya bersedia (**pilih salah satu**):

- ☒ Saya bersedia memberikan izin sepenuhnya kepada Universitas Multimedia Nusantara untuk mempublikasikan hasil karya ilmiah saya ke dalam repositori Knowledge Center sehingga dapat diakses oleh Sivitas Akademika UMN/Publik. Saya menyatakan bahwa karya ilmiah yang saya buat tidak mengandung data yang bersifat konfidensial.
- ☐ Saya tidak bersedia mempublikasikan hasil karya ilmiah ini ke dalam repositori Knowledge Center, dikarenakan: dalam proses pengajuan publikasi ke jurnal/konferensi nasional/internasional (dibuktikan dengan *letter of acceptance*) **.
- ☐ Lainnya, pilih salah satu:
- Hanya dapat diakses secara internal Universitas Multimedia Nusantara
 - Embargo publikasi karya ilmiah dalam kurun waktu tiga tahun.

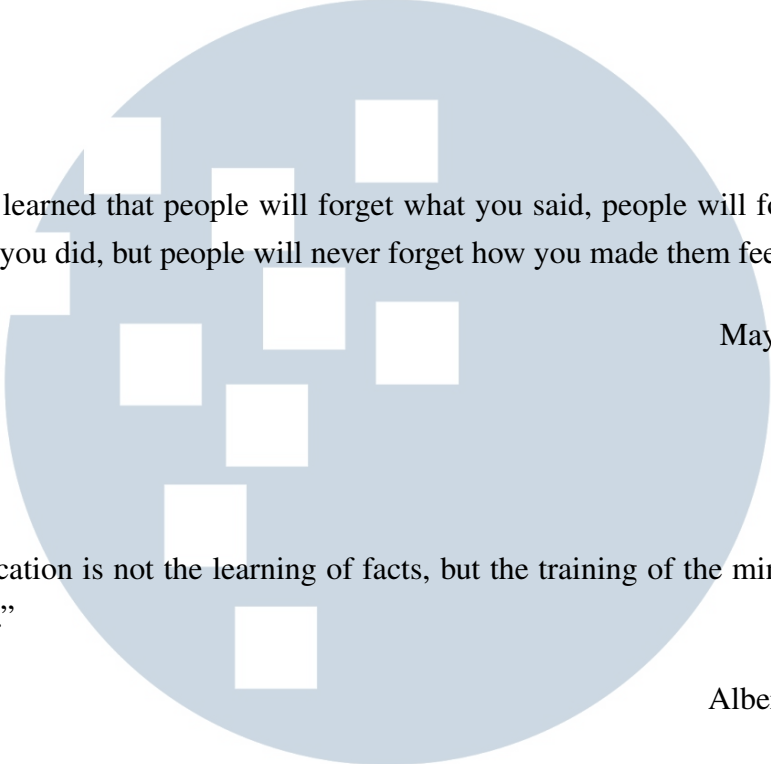
Tangerang, 16 Juni 2025

Yang menyatakan



Arrafi Aji Pamungkas

HALAMAN PERSEMBAHAN / MOTTO



"I've learned that people will forget what you said, people will forget what you did, but people will never forget how you made them feel."

Maya Angelou

"Education is not the learning of facts, but the training of the mind to think."

Albert Einstein

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala limpahan rahmat, karunia, petunjuk, dan kekuatan yang senantiasa diberikan sehingga penulis dapat menyelesaikan skripsi yang berjudul “*Perbandingan Model Klasifikasi Ujaran Kebencian di Media Sosial Berbasis BERT, XGBoost, dan Hybrid BERT-XGBoost*” dengan baik dan lancar. Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Informatika di Universitas Multimedia Nusantara. Penulis menyadari bahwa proses penyusunan skripsi ini bukanlah hal yang mudah, karena di dalamnya terdapat berbagai tantangan yang menguji kemampuan dalam mengatur waktu, mengendalikan emosi, menjaga fokus, dan mempertahankan motivasi ditengah berbagai keterbatasan. Oleh sebab itu, penulis ingin menyampaikan rasa terima kasih yang sedalam-dalamnya kepada semua pihak yang telah membantu, membimbing, dan memberikan dorongan moral maupun materiil selama proses penyusunan skripsi ini kepada:

1. Bapak Dr. Ir. Andrey Andoko, M.Sc., selaku Rektor Universitas Multimedia Nusantara.
2. Bapak Dr. Eng. Niki Prastomo, S.T., M.Sc., selaku Dekan Fakultas Teknik dan Informatika Universitas Multimedia Nusantara.
3. Bapak Arya Wicaksana, S.Kom., M.Eng.Sc., OCA, selaku Ketua Program Studi Informatika Universitas Multimedia Nusantara.
4. Bapak Aditriyawan, S.Kom., M.Si., sebagai Pembimbing pertama yang telah meluangkan waktu serta memberikan bimbingan, arahan, dan motivasi atas terselesainya tugas akhir ini.
5. Dosen dan staf pengajar di Teknik Informatika, yang telah memberikan ilmu dan wawasan selama masa perkuliahan.
6. Keluarga saya yang telah memberikan dukungan penuh serta selalu mendukung apapun keputusan yang penulis ambil hingga dapat menyelesaikan penulisan skripsi.

7. Teman-teman selama perkuliahan penulis yang selalu menemani, *shout out* buat kalian **Fadhil Rahman, Cornellius Baros, Ville Jason, William Reyhan** secara tidak langsung dalam penulisan skripsi ini.
8. Seluruh pihak yang telah menemani selama proses penulisan skripsi ini dan memberikan kontribusi, dukungan, serta doa kepada penulis

Semoga karya ilmiah ini dapat bermanfaat bagi pembaca dan menjadi referensi berguna untuk mengembangkan keilmuan dalam bidang teknologi.

Tangerang, 16 Juni 2025



Arrafi Aji Pamungkas



**PERBANDINGAN MODEL KLASIFIKASI UJARAN KEBENCIAN DI
MEDIA SOSIAL BERBASIS BERT, XGBOOST, DAN HYBRID
BERT-XGBOOST**

Arrafi Aji Pamungkas

ABSTRAK

Perkembangan media sosial telah memicu peningkatan signifikan dalam penyebaran ujaran kebencian daring, khususnya pada platform X (Twitter). Penelitian ini mengusulkan pendekatan *hybrid* berbasis *Bidirectional Encoder Representations from Transformers* (BERT) dan algoritma *Extreme Gradient Boosting* (XGBoost) untuk mendeteksi konten bermuatan ujaran kebencian secara lebih akurat. Dataset dikumpulkan melalui scraping menggunakan Twitter Search API dan anotasi berbasis *crowdsourcing*, mencakup kategori, target, dan intensitas ujaran kebencian, dengan total sebanyak 13.014 data. Tiga model dikembangkan dan dibandingkan: BERT + XGBoost, XGBoost berbasis TF-IDF, dan BERT *finetuned*. Hasil evaluasi menunjukkan bahwa model *hybrid* mencapai akurasi 81%, namun tidak melampaui performa model BERT *finetuned*, yang memperoleh akurasi 88,99% dan F1-score 0,8893. Temuan ini mengindikasikan bahwa pendekatan *hybrid* tidak selalu menjamin peningkatan kinerja dalam konteks klasifikasi teks berbahasa Indonesia.

Kata kunci: klasifikasi teks, ujaran kebencian, BERT, XGBoost, *hybrid model*.



**COMPARATIVE ANALYSIS OF HATE SPEECH CLASSIFICATION
MODELS ON SOCIAL MEDIA: BERT, XGBOOST, AND HYBRID
BERT-XGBOOST**

Arrafi Aji Pamungkas

ABSTRACT

The rise of social media has significantly accelerated the spread of online hate speech, particularly on Platform X (Twitter). This study proposes a hybrid approach that combines Bidirectional Encoder Representations from Transformers (BERT) and Extreme Gradient Boosting (XGBoost) to improve the accuracy of hate speech detection. The dataset was collected via scraping using the Twitter Search API and annotated through crowdsourcing, covering the categories, targets, and intensity levels of hate speech, resulting in a total of 13,014 entries. Three models were developed and compared: a hybrid BERT + XGBoost model, an XGBoost model with TF-IDF features, and a fine-tuned BERT model. Evaluation results show that the hybrid model achieved an accuracy of 81%, but did not outperform the fine-tuned BERT model, which attained an accuracy of 88.99% and an F1-score of 0.8893. These findings indicate that hybrid approaches do not always guarantee performance improvements, especially in the context of Indonesian-language text classification.

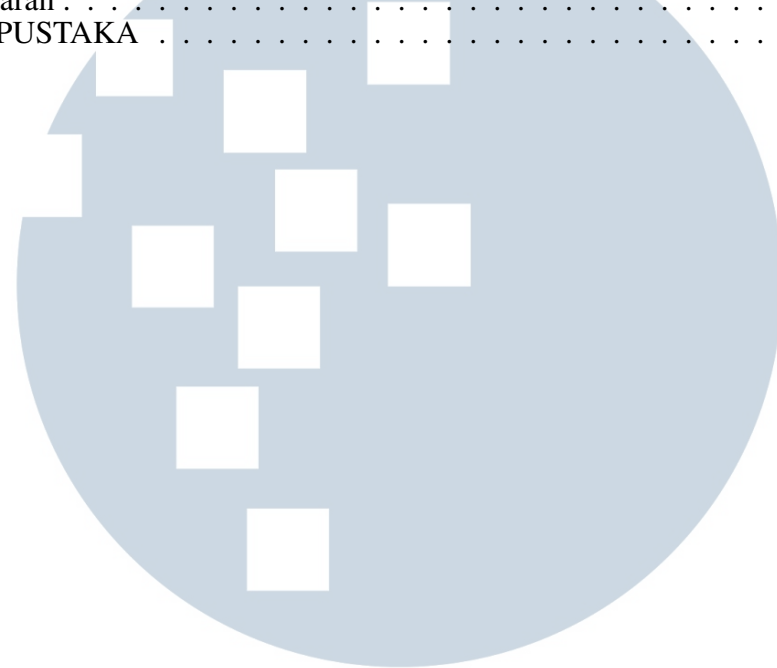
Keywords: text classification, hate speech, BERT, XGBoost, *hybrid model*.



DAFTAR ISI

HALAMAN JUDUL	i
PERNYATAAN TIDAK MELAKUKAN PLAGIAT	ii
HALAMAN PENGESAHAN	iii
HALAMAN PERSETUJUAN PUBLIKASI KARYA ILMIAH	iv
HALAMAN PERSEMBAHAN/MOTO	v
KATA PENGANTAR	vi
ABSTRAK	viii
ABSTRACT	ix
DAFTAR ISI	x
DAFTAR TABEL	xii
DAFTAR GAMBAR	xiii
DAFTAR KODE	xiv
DAFTAR RUMUS	xv
DAFTAR LAMPIRAN	xvi
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah	2
1.3 Batasan Permasalahan	3
1.4 Tujuan Penelitian	3
1.5 Manfaat Penelitian	3
1.6 Sistematika Penulisan	4
BAB 2 LANDASAN TEORI	5
2.1 Ujaran Kebencian	5
2.2 Media Sosial X	5
2.3 TF-IDF	6
2.4 Transformer	7
2.4.1 Bidirectional Encoder Representations from Transformers (BERT)	8
2.4.2 indoBERT	10
2.5 Ensemble Learning	10
2.5.1 Gradient Boosting	11
2.5.2 XGBoost	12
2.6 Confussion Matrix	15
BAB 3 METODOLOGI PENELITIAN	17
3.1 Studi Literatur	17
3.2 Pengumpulan Data	18
3.3 Preprocessing Data	18
3.4 Perancangan Model	20
3.4.1 Split Dataset	21
3.4.2 Feature Extraction	22
3.4.3 Hyperparameter Tuning	22
3.5 Pengujian dan Evaluasi	23
BAB 4 HASIL DAN DISKUSI	25
4.1 Spesifikasi Sistem	25
4.2 Deskripsi Dataset	26
4.3 Implementasi Sistem	27
4.3.1 Import Libraries	27
4.3.2 Preprocessing Data	28

4.3.3	Perancangan Model	36
4.4	Uji Coba dan Evaluasi Model	47
4.4.1	Rangkuman dan Evaluasi Akhir Antar Model	54
BAB 5	SIMPULAN DAN SARAN	56
5.1	Simpulan	56
5.2	Saran	57
DAFTAR PUSTAKA	58



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

DAFTAR TABEL

Tabel 4.1	Contoh lima data pertama	26
Tabel 4.2	Hasil <i>remove</i> kolom	29
Tabel 4.3	Data sebelum dan sesudah <i>cleaning</i>	33
Tabel 4.4	Ringkasan jumlah data dan hasil audit label	34
Tabel 4.5	Distribusi label <i>hate speech</i> sebelum dan sesudah <i>relabeling</i>	35
Tabel 4.6	Distribusi dataset	37
Tabel 4.7	Hasil evaluasi model pada data uji	48
Tabel 4.8	Hasil evaluasi model pada data uji	50
Tabel 4.9	Hasil evaluasi model pada data uji	53
Tabel 4.10	Evaluasi model untuk klasifikasi <i>hate-speech</i> dan <i>non-hate</i>	54



DAFTAR GAMBAR

Gambar 2.1	Arsitektur <i>transformer</i>	8
Gambar 2.2	Arsitektur proses <i>pretraining</i> dan <i>finetuning</i> pada <i>bidirectional encoder representations from transformers</i> (bert)	9
Gambar 2.3	Arsitektur Gradient Boosting	12
Gambar 2.4	Arsitektur XGBoost	15
Gambar 3.1	Alur metodologi penelitian	17
Gambar 3.2	Alur Preprocessing Dataset	19
Gambar 3.3	Alur Perancangan Model	20
Gambar 4.1	Grafik <i>loss</i> dan akurasi <i>training</i> dan <i>validation</i> model bert+ <i>xgboost</i>	47
Gambar 4.2	<i>Confusion matrix</i> model bert+ <i>xgboost</i>	48
Gambar 4.3	Grafik <i>loss</i> dan akurasi <i>training</i> dan <i>validation</i> model <i>xgboost</i> tf-idf	50
Gambar 4.4	<i>Confusion matrix</i> model <i>xgboost</i> tf-idf	51
Gambar 4.5	Grafik <i>loss</i> dan akurasi <i>training</i> dan <i>validation</i> model bert finetuned	52
Gambar 4.6	<i>Confusion matrix</i> model bert finetuned	53



DAFTAR KODE

Kode 4.1	Import libraries	27
Kode 4.2	Pemeriksaan dataset	28
Kode 4.3	Potongan kode <i>remove</i> kolom	29
Kode 4.4	Cek <i>null</i> dan hapus duplikasi	30
Kode 4.5	Fungsi-fungsi <i>cleaning text</i>	30
Kode 4.6	Fungsi <i>preprocessing</i>	32
Kode 4.7	Deteksi <i>mislabeled</i>	34
Kode 4.8	<i>Update label</i>	35
Kode 4.9	Split dataset	36
Kode 4.10	Menampilkan distribusi dataset setelah split	37
Kode 4.11	Fungsi ekstraksi <i>embedding</i> dengan <i>mean pooling</i>	38
Kode 4.12	Proses <i>embedding</i> untuk seluruh dataset	40
Kode 4.13	<i>Hyperparameter</i> dengan grid search	41
Kode 4.14	Inisialisasi <i>dmatrix</i> dan <i>best parameters</i>	42
Kode 4.15	<i>Callback logging metrics</i> evaluasi	43
Kode 4.16	Pelatihan model <i>xgboost</i>	45
Kode 4.17	Evaluasi <i>metrics</i> pada data uji	46



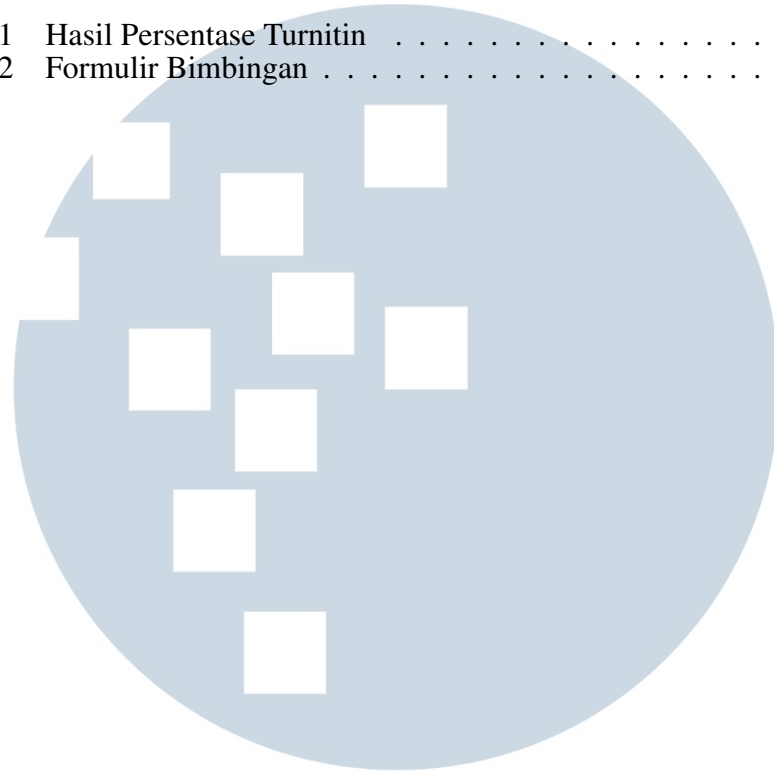
DAFTAR RUMUS

Rumus 2.1	<i>Term Frequency (TF) Function</i>	6
Rumus 2.2	<i>Inverse Document Frequency (IDF) Function</i>	7
Rumus 2.3	<i>TF-IDF Function</i>	7
Rumus 2.4	<i>Gradient Boosting Prediciton Function</i>	11
Rumus 2.5	<i>XGBoost Prediction Function</i>	13
Rumus 2.6	<i>XGBoost Objective Function</i>	13
Rumus 2.7	<i>XGBoost Regularization Term</i>	13
Rumus 2.8	<i>Second-Order Approximation of XGBoost Loss</i>	13
Rumus 2.9	<i>XGBoost Gain Function</i>	14
Rumus 2.10	<i>Accuracy</i>	16
Rumus 2.11	<i>Precision</i>	16
Rumus 2.12	<i>Recall (Sensitivity)</i>	16
Rumus 2.13	<i>F1-Score</i>	16



DAFTAR LAMPIRAN

Lampiran 1	Hasil Persentase Turnitin	62
Lampiran 2	Formulir Bimbingan	72



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

BAB 1 PENDAHULUAN

1.1 Latar Belakang Masalah

Perkembangan teknologi digital telah mengubah berbagai aspek kehidupan, meliputi ekonomi, sosial, dan pendidikan. Kemajuan ini membawa tantangan baru, termasuk meningkatnya ancaman konten ujaran kebencian (*hate speech*) pada platform digital. Ujaran kebencian merujuk pada ekspresi berbasis teks, gambar, atau simbol yang menyebarkan kebencian, diskriminasi, atau permusuhan terhadap individu maupun kelompok berdasarkan atribut seperti ras, agama, atau etnis, sering kali memanfaatkan anonimitas di dunia maya [1, 2]. Dampaknya mencakup gangguan emosional, polarisasi sosial, dan dalam kasus tertentu, eskalasi konflik di dunia nyata.

Pendekatan *machine learning* telah banyak diterapkan dalam mendeteksi ujaran kebencian pada platform digital. Sejak awal kemunculan *Natural Language Processing* (NLP), metode tradisional seperti Bag-of-Words dan TF-IDF—serta penggunaan kamus khusus—telah digunakan untuk menganalisis teks [3, 4]. Pendekatan ini kemudian berkembang dengan penerapan algoritma klasifikasi tradisional seperti Support Vector Machine (SVM) yang menggunakan representasi fitur sederhana [5], dan untuk mengurangi kompleksitas vektor fitur yang tinggi, teknik reduksi dimensi seperti Principal Component Analysis (PCA) sering diterapkan untuk menghasilkan representasi yang lebih ringkas dan efisien, tetapi kurang mampu menangani analisis kompleks [6].

Platform media sosial, khususnya Platform X (Twitter), menjadi saluran utama penyebaran ujaran kebencian di Indonesia [7]. Dengan basis pengguna yang besar dan interaksi yang intens, konten agresif dapat menyebar dengan cepat melalui fitur retweet dan hashtag [8]. Penelitian terkini menunjukkan adanya peningkatan signifikan dalam paparan ujaran kebencian daring di Indonesia, dengan rasio yang meningkat sepuluh kali lipat dalam dua tahun terakhir, terutama menargetkan kelompok minoritas agama dan etnis [9]. Fenomena ini menimbulkan kekhawatiran besar terkait keamanan, privasi, serta kualitas interaksi di ruang digital, khususnya bagi generasi muda yang merupakan pengguna aktif media sosial.

Maraknya ujaran kebencian menuntut pendekatan yang lebih kompleks

untuk mendeteksi dan mencegah perilaku tersebut secara efektif. Salah satu teknik yang semakin mendapat perhatian adalah penggunaan model *Bidirectional Encoder Representations from Transformers* (BERT). Model ini mampu memahami konteks bahasa secara lebih mendalam dibandingkan metode tradisional, sehingga lebih akurat dalam mengidentifikasi kalimat ujaran kebencian yang rumit [10, 11]. Studi menunjukkan efektivitas BERT dalam mendeteksi ujaran kebencian, mencapai *F1-measure* 92% untuk bahasa Inggris pada dataset Davidson dan *macro-F1* 0,78 untuk bahasa Indonesia pada IndoToxic2024 [12, 9]. Selain itu, penelitian lainnya telah menunjukkan bahwa pendekatan berbasis BERT mampu meningkatkan performa deteksi *hate-speech* dengan tingkat akurasi yang lebih tinggi, menjadikannya solusi yang menjanjikan dalam analisis teks [13, 14, 15, 16].

Sebagai pengembangan dari pendekatan tersebut, penelitian ini mengusulkan model *hybrid* yang menggabungkan BERT dengan algoritma *Extreme Gradient Boosting* (XGBoost). Integrasi ini bertujuan untuk memanfaatkan kekuatan representasi semantik yang dihasilkan oleh BERT dan menggabungkannya dengan kemampuan klasifikasi XGBoost yang efisien dan cakap dalam menangani data berdimensi tinggi dan ketidakseimbangan kelas [17]. Dalam pendekatan ini, representasi vektor teks yang diekstraksi oleh BERT digunakan sebagai *input* bagi XGBoost dalam proses klasifikasi [18]. Dengan demikian, strategi *hybrid* menggabungkan pemahaman bahasa alami yang mendalam dengan performa klasifikasi yang tinggi, dan diharapkan mampu meningkatkan akurasi dalam mendeteksi ujaran kebencian, terutama dalam konteks sosial media berbahasa Indonesia seperti Platform X [17, 18, 19].

1.2 Rumusan Masalah

Berdasarkan pemaparan latar belakang masalah diatas, rumusan masalah yang dibahas dalam penulisan adalah sebagai berikut :

1. Bagaimana cara mendeteksi komentar tweet yang mengandung unsur ujaran kebencian.
2. Bagaimana performa model dalam membedakan komentar tweet berdasarkan *accuracy*, *precision*, *recall*, *F1-score*, dan *confusion matrix* yang mengandung unsur ujaran kebencian.

1.3 Batasan Permasalahan

Pada bagian ini dijabarkan batasan yang diterapkan dalam penelitian ini agar pelaksanaan penelitian menjadi lebih terfokus kepada aspek-aspek berikut :

1. Penelitian hanya berfokus pada deteksi tweet ujaran kebencian.
2. Penelitian berfokus pada hasil model mengenai klasifikasi tweet ujaran kebencian.
3. Klasifikasi akan dibagi menjadi sentimen positif dan negatif.
4. Dataset yang digunakan terbatas pada hasil tweets yang diambil dari github.

1.4 Tujuan Penelitian

Berdasarkan permasalahan yang telah dirumuskan, berikut adalah tujuan yang telah dijabarkan :

1. Membuat model yang dapat mendeteksi komentar tweet yang mengandung unsur ujaran kebencian menggunakan pendekatan berbasis model *machine learning hybrid*.
2. Mengevaluasi performa model dalam membedakan komentar tweet yang mengandung ujaran kebencian berdasarkan *accuracy*, *precision*, *recall*, *F1-score*, dan *confusion matrix*.

1.5 Manfaat Penelitian

Berdasarkan penelitian dalam implementasi BERT dan XGBoost untuk deteksi konten hate speech pada media sosial X, berikut adalah manfaat penelitian ini:

1. **Bagi Peneliti**
 - (a) Menerapkan dan memperdalam pemahaman teori *machine learning* dan *Natural Language Processing* (NLP) melalui pengembangan serta evaluasi kombinasi algoritma BERT dan XGBoost untuk deteksi ujaran kebencian.

- (b) Memberikan pengalaman praktis dalam mengintegrasikan model *hybrid* BERT dan XGBoost untuk menghasilkan analisis klasifikasi teks ujaran kebencian di media sosial.

2. Bagi Pihak Lain

- (a) Mengidentifikasi konten ujaran kebencian secara lebih akurat dan efisien melalui pendekatan model *hybrid* BERT dan XGBoost.
- (b) Menyediakan wawasan akademik mengenai penerapan model kombinasi BERT, XGBoost, dalam mendeteksi ujaran kebencian yang dapat menjadi dasar bagi penelitian lanjutan untuk mengoptimalkan akurasi dan performa model.

1.6 Sistematika Penulisan

Sistematika penulisan berupa struktur penulisan yang terdiri dari lima bab yang membahas aspek penting penelitian, yaitu :

Sistematika penulisan laporan adalah sebagai berikut:

- Bab 1 PENDAHULUAN
Bab ini berisi latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.
- Bab 2 LANDASAN TEORI
Bab ini membahas teori-teori dan konsep-konsep yang relevan dengan topik penelitian, termasuk penelitian terdahulu yang mendukung dasar teori penelitian ini.
- Bab 3 METODOLOGI PENELITIAN
Bab ini menjelaskan metode yang digunakan dalam penelitian, seperti pengumpulan data, perancangan sistem, teknik implementasi, serta tahapan-tahapan yang dilakukan dalam penelitian.
- Bab 4 HASIL DAN DISKUSI
Bab ini menyajikan hasil penelitian, analisis data, serta penjelasan mengenai perbandingan hasil yang diberikan.
- Bab 5 SIMPULAN DAN SARAN
Bab ini memuat kesimpulan dari hasil penelitian dan saran-saran yang dapat diberikan untuk pengembangan lebih lanjut atau penelitian selanjutnya.

BAB 2

LANDASAN TEORI

Landasan teori ini berfungsi sebagai landasan dalam melakukan analisis permasalahan dan merancang solusi dalam penelitian ini. Secara umum, landasan teori ini mencakup ujaran kebencian (*hate speech*), media sosial X, *Natural Language Processing*, *transformer*, BERT, *indoBERT*, Ensemble Learning, XGBoost, serta penelitian terdahulu mengenai teori tersebut.

2.1 Ujaran Kebencian

Ujaran Kebencian atau *hate speech* merujuk pada ekspresi berbasis teks, gambar, atau simbol yang bertujuan menyebarkan kebencian, diskriminasi, atau permusuhan terhadap individu maupun kelompok berdasarkan atribut seperti ras, agama, etnis, jenis kelamin, atau orientasi seksual [1]. Ekspresi ini kerap memanfaatkan anonimitas platform digital untuk memperluas jangkauan serta intensitas dampaknya [2]. Karakteristik utama mencakup bahasa ofensif, stereotip negatif, dan niat untuk merendahkan atau memicu konflik terhadap targetnya.

Paparan ujaran kebencian menyebabkan gangguan emosional seperti kecemasan dan penurunan harga diri, terutama pada remaja dan kelompok rentan. Dampak lebih luas meliputi polarisasi sosial, perpecahan antar kelompok, hingga potensi eskalasi kekerasan fisik [2]. Oleh karena itu, deteksi ujaran kebencian menjadi krusial untuk mitigasi risiko tersebut, terutama melalui pendekatan berbasis *machine learning* yang memanfaatkan analisis teks otomatis. Teknologi ini mendukung identifikasi konten berbahaya secara efisien di lingkungan digital.

2.2 Media Sosial X

Media Sosial X atau biasa dikenal sebagai Twitter, merupakan platform daring yang memungkinkan pengguna berbagi pesan singkat berjumlah maksimum 280 karakter per postingan [20]. Fitur utama, seperti *retweet*, *hashtag*, dan *reply*, mendukung interaksi cepat antar pengguna, menjadikan platform ini pusat komunikasi global pengguna aktif secara *real-time* [21]. Struktur terbuka memfasilitasi penyebaran informasi secara instan, termasuk konten berupa ujaran kebencian, akibat kemampuan amplifikasi melalui mekanisme jaringan sosial [8].

Media sosial X telah menjadi wadah bagi masyarakat untuk menyuarakan berbagai opini, menjadikannya sebagai ruang digital yang terbuka untuk diskusi publik. Namun, di sisi lain, platform ini juga dimanfaatkan sebagai sarana penyebaran ujaran kebencian, khususnya melalui penggunaan *hashtag* yang memungkinkan konten bermuatan ofensif tersebar luas dalam waktu singkat [22]. Penelitian mencatat rasio ujaran kebencian secara daring di Indonesia meningkat sepuluh kali lipat dalam dua tahun terakhir, dengan kelompok minoritas agama dan etnis menjadi sasaran utama [9].

Tingginya volume interaksi dan akses secara simultan oleh pengguna mempercepat laju penyebaran konten negatif, menjadikannya sulit untuk dikendalikan secara *real time*. Kondisi ini semakin diperparah oleh tingkat anonimitas yang tinggi di platform, yang memungkinkan individu mengekspresikan kebencian tanpa konsekuensi langsung atau hambatan yang berarti [23].

Paparan ujaran kebencian di Media Sosial X memengaruhi individu dan kelompok secara luas. Konten tersebut kerap menggunakan bahasa ofensif serta stereotip negatif untuk menargetkan korban berdasarkan ras, agama, atau etnis [24]. Dampaknya mencakup gangguan emosional seperti kecemasan pada pengguna rentan, polarisasi sosial antar komunitas, hingga potensi eskalasi konflik di dunia nyata [23].

2.3 TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) adalah metode statistik yang digunakan untuk mengevaluasi seberapa penting suatu kata dalam sebuah dokumen relatif terhadap kumpulan dokumen lainnya (corpus). Metode ini banyak digunakan dalam pencarian informasi dan penambangan teks untuk merepresentasikan teks ke dalam bentuk numerik yang dapat diproses oleh algoritma pembelajaran mesin [25].

Term Frequency (TF) mengukur seberapa sering sebuah kata muncul dalam sebuah dokumen. Semakin sering kata tersebut muncul, semakin besar nilainya. Dinyatakan sebagai:

$$TF(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \quad (2.1)$$

di mana $f_{t,d}$ adalah jumlah kemunculan kata t dalam dokumen d .

Inverse Document Frequency (IDF) mengukur seberapa unik atau jarang kata tersebut di seluruh dokumen dalam korpus. Dinyatakan sebagai:

$$IDF(t) = \log \left(\frac{N}{df_t} \right) \quad (2.2)$$

dengan N adalah jumlah total dokumen, dan df_t adalah jumlah dokumen yang mengandung kata t .

Nilai akhir dari TF-IDF diperoleh dengan mengalikan nilai TF dan IDF:

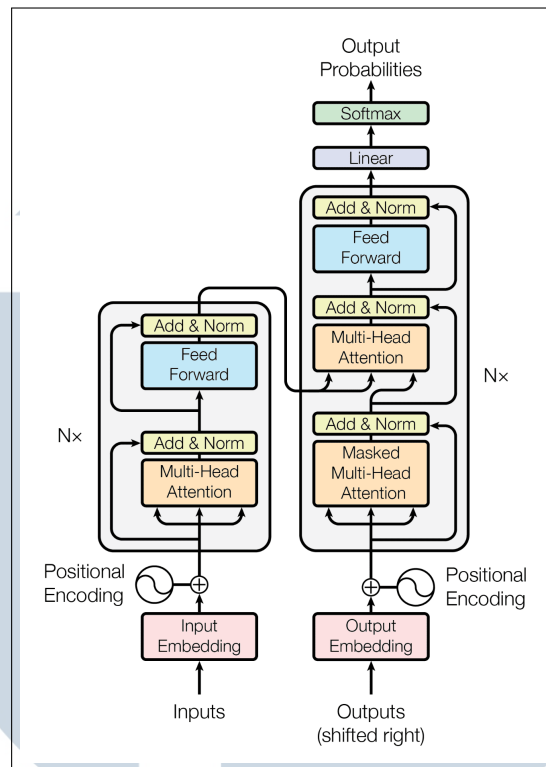
$$TFIDF(t, d) = TF(t, d) \times IDF(t) \quad (2.3)$$

Dengan pendekatan ini, kata-kata umum yang sering muncul di banyak dokumen akan memiliki bobot rendah, sementara kata-kata yang jarang namun relevan dalam konteks dokumen tertentu akan memiliki bobot lebih tinggi.

2.4 Transformer

Transformer merupakan model *neural network* berbasis *sequence-to-sequence* yang sepenuhnya mengandalkan mekanisme *self attention* untuk memproses data urutan, tanpa menggunakan jaringan berulang (*recurrent neural networks*) atau konvolusi (*convolutional neural networks*). Berbeda dengan pendekatan sebelumnya yang memproses teks secara berurutan, *self-attention* memungkinkan *Transformer* menangkap hubungan antar elemen dalam data secara simultan. Model ini menjadi dasar bagi arsitektur modern seperti BERT dan *IndoBERT*, atau model BERT lainnya yang relevan untuk tugas klasifikasi teks seperti deteksi tweet ujaran kebencian pada Platform X.

Arsitektur *Transformer* terdiri dari encoder-decoder dan setiap layer memiliki mekanisme *self-attention* bertumpuk yang dilengkapi dengan *multi head attention* untuk menangkap berbagai aspek hubungan dalam teks. Encoder mengolah data masukan menjadi representasi kontekstual, sedangkan decoder menghasilkan keluaran berdasarkan representasi tersebut, seperti terlihat pada Gambar 2.1.



Gambar 2.1. Arsitektur *transformer*

Sumber: [26]

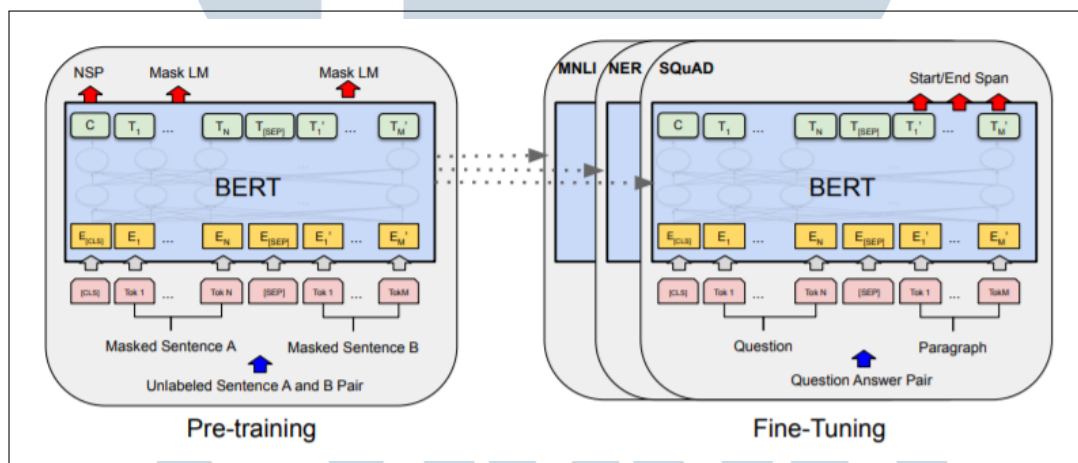
2.4.1 Bidirectional Encoder Representations from Transformers (BERT)

Bidirectional Encoder Representations from Transformers (BERT) adalah model *natural language processing* (NLP) yang dikembangkan untuk memahami konteks kata dalam kalimat secara mendalam. Model ini pertama kali diperkenalkan oleh Devlin et al. (2019) pada Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. BERT unik karena menggunakan pendekatan bidirectional, yang memungkinkan model mempertimbangkan konteks dari arah kiri dan kanan kata dalam kalimat secara bersamaan [11]. Pendekatan ini berbeda dengan model sebelumnya seperti Word2Vec yang bersifat unidirectional [27].

BERT dibangun berdasarkan arsitektur *Transformer*, yang pertama kali diperkenalkan oleh Vaswani et al. (2017) dalam *Advances in Neural Information Processing Systems*. *Transformer* terdiri dari encoder dan decoder, tetapi BERT hanya menggunakan bagian encoder yang terdiri dari lapisan *self attention* dan *feed-forward neural network* [28]. Mekanisme *self attention* memungkinkan BERT untuk menimbang pentingnya setiap kata dalam kalimat terhadap kata lainnya,

menghasilkan representasi vektor yang kontekstual.

Proses kerja BERT melibatkan dua tahap utama: *pretraining* dan *finetuning*, sebagaimana divisualisasikan pada Gambar 2.1. Pada tahap *pre-training*, BERT dilatih dengan dua tugas utama: *Masked Language Model* (MLM) dan *Next Sentence Prediction* (NSP). Dalam MLM, sekitar 15% kata dalam kalimat disembunyikan secara acak, dan model dilatih untuk memprediksi kata-kata tersebut berdasarkan konteks. Sementara itu, NSP melatih model untuk memprediksi apakah dua kalimat berurutan memiliki hubungan logis, seperti ditunjukkan pada bagian kiri Gambar 2.2 dengan "Masked Sentence A" dan "Masked Sentence B". Tahap *finetuning* memungkinkan BERT disesuaikan untuk tugas spesifik, seperti klasifikasi ujaran kebencian dalam penelitian ini, dengan menambahkan lapisan output pada vektor [CLS], sebagaimana digambarkan pada bagian kanan Gambar 2.2.



Gambar 2.2. Arsitektur proses *pretraining* dan *finetuning* pada *bidirectional encoder representations from transformers* (bert)

Sumber: [29]

Keunggulan BERT terletak pada kemampuan *finetuning* untuk berbagai tugas NLP, seperti klasifikasi teks dan deteksi ujaran kebencian. Menurut Mozafari et al. (2020) dalam jurnal PLOS ONE, BERT efektif menangkap nuansa bahasa yang kompleks seperti sarkasme dalam deteksi ujaran kebencian [12]. Representasi *bidirectional* yang dihasilkan oleh BERT, sebagaimana divisualisasikan pada Gambar 2.2, memungkinkan model untuk memahami konteks yang lebih kaya dibandingkan pendekatan tradisional, menjadikannya dasar yang kuat untuk pengembangan varian seperti BERT lainnya.

2.4.2 indoBERT

IndoBERT merupakan model bahasa berbasis arsitektur BERT yang dikembangkan secara khusus untuk pemrosesan bahasa alami (NLP) dalam bahasa Indonesia [30]. Model ini dilatih menggunakan lebih dari 220 juta kata yang dikumpulkan dari berbagai sumber teks berbahasa Indonesia, termasuk artikel Wikipedia, berita daring, dan media sosial. IndoBERT dirancang untuk memberikan representasi linguistik yang lebih akurat dan kontekstual dalam bahasa Indonesia. Kemampuannya yang unggul dalam berbagai tugas NLP, seperti klasifikasi teks, analisis sentimen, dan peringkasan, menjadikan IndoBERT sebagai *base line* baru dalam pengembangan aplikasi NLP berbahasa Indonesia. Secara arsitektural, IndoBERT mengadopsi struktur *BERT-base* dengan 12 lapisan *transformer*, dimensi *hidden state* sebesar 768, serta 12 *attention heads* [30]. Dengan jumlah parameter mencapai sekitar 110 juta, IndoBERT menawarkan keseimbangan antara performa dan efisiensi komputasi.

Sejumlah penelitian telah menunjukkan efektivitas IndoBERT dalam berbagai aplikasi NLP berbahasa Indonesia. Koto et al. [30] menguji performa IndoBERT pada benchmark dataset IndoLEM, yang mencakup berbagai tugas seperti analisis sentimen dan peringkasan teks. Hasilnya menunjukkan bahwa IndoBERT mampu mengungguli model BERT multilingual, dengan peningkatan akurasi hingga 5% pada beberapa tugas. Selanjutnya, Cahyawijaya et al. [31] menunjukkan bahwa IndoBERT dapat diadaptasi untuk tugas-tugas generatif seperti teks otomatisasi, dengan hasil yang kompetitif dibandingkan model sejenis. Selain itu, studi oleh [32] mengeksplorasi penerapan IndoBERT dalam analisis sentimen di media sosial, dan mencatatkan skor F1 diatas 0,85, yang menunjukkan kemampuan model ini dalam menangani teks informal berbahasa Indonesia secara efektif.

2.5 Ensemble Learning

Ensemble learning adalah pendekatan dalam pembelajaran mesin yang menggabungkan beberapa model pembelajaran (*base learners*) untuk menghasilkan prediksi yang lebih akurat, stabil, dan robust dibandingkan model tunggal. Teknik ini bertujuan meminimalkan bias, variance, serta kesalahan prediksi dengan memanfaatkan keunggulan masing-masing model secara bersamaan[33]. Secara umum, terdapat tiga strategi utama dalam ensemble learning, yaitu bagging, boosting, dan stacking. Bagging (bootstrap aggregating) bekerja dengan melatih

beberapa model secara paralel pada subset data acak untuk mengurangi variance, seperti yang diterapkan pada algoritma Random Forest [34]. Sebaliknya, boosting melatih model secara berurutan dengan menitikberatkan pada perbaikan kesalahan prediksi model sebelumnya sehingga dapat mengurangi bias dan meningkatkan akurasi, contohnya pada AdaBoost dan XGBoost [35]. Sementara itu, stacking menggabungkan output dari beberapa model dasar menggunakan meta-learner untuk menghasilkan prediksi akhir yang optimal dengan memanfaatkan pola kesalahan antar model [36].

2.5.1 Gradient Boosting

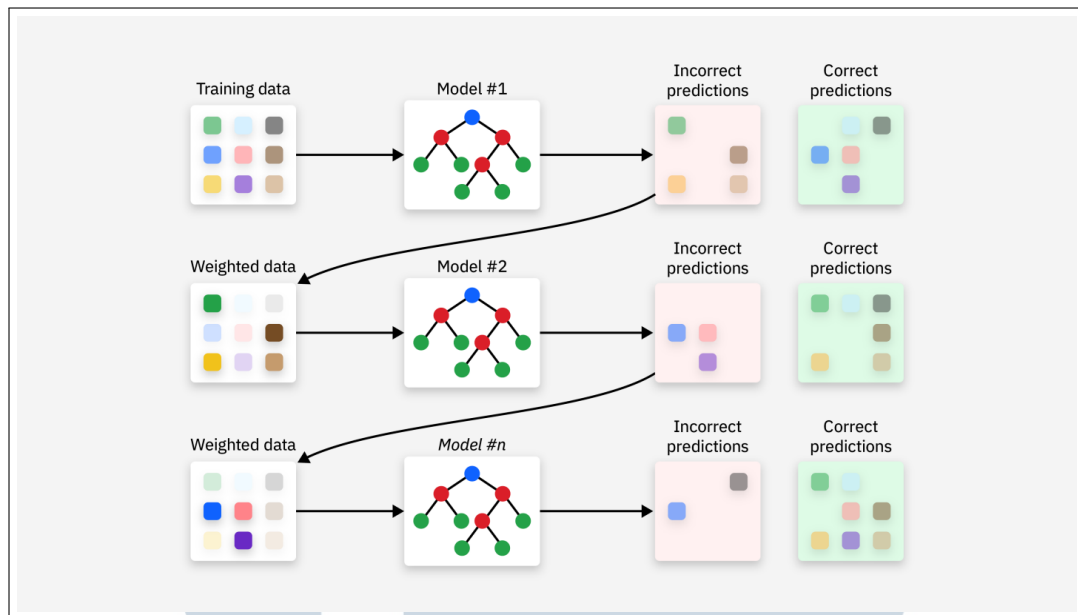
Gradient Boosting adalah metode *ensemble learning* yang digunakan untuk meningkatkan akurasi prediksi dengan menggabungkan sejumlah model sederhana (biasanya berupa pohon keputusan) secara bertahap. Konsep utamanya adalah setiap model baru dibangun untuk memperbaiki kesalahan prediksi dari model sebelumnya dengan menggunakan pendekatan *gradient descent* terhadap fungsi kerugian [37].

Secara sederhana, proses kerja Gradient Boosting dapat diringkas dalam formula berikut:

$$\hat{y}_i = \sum_{m=1}^M h_m(x_i) \quad (2.4)$$

dengan \hat{y}_i adalah hasil prediksi akhir untuk data ke- i , dan $h_m(x)$ adalah pohon keputusan ke- m yang dilatih untuk meminimalkan kesalahan dari prediksi sebelumnya. Model ini berfokus pada data yang sulit diprediksi dengan memberikan bobot lebih besar terhadap kesalahan sebelumnya, sehingga proses pembelajaran menjadi lebih adaptif.

Visualisasi alur kerja Gradient Boosting dapat dilihat pada Gambar 2.3, yang menggambarkan proses pembentukan pohon keputusan secara bertahap, di mana setiap pohon bertugas mengoreksi prediksi dari pohon sebelumnya.



Gambar 2.3. Arsitektur Gradient Boosting

Sumber: [38]

Visualisasi pada Gambar 2.3 menggambarkan alur kerja Gradient Boosting secara bertahap. Proses dimulai dari pelatihan model pertama (*Model 1*) menggunakan data awal. Setelah prediksi dilakukan, data yang salah diklasifikasikan akan dikenali dan diberi bobot lebih besar. Data berbobot ini kemudian digunakan untuk melatih model kedua (*Model 2*), yang fokus memperbaiki kesalahan dari model sebelumnya. Proses ini berlanjut hingga model ke-*n*, di mana setiap model baru dilatih untuk mengurangi error (*residual*) dari hasil prediksi kumulatif sebelumnya. Akhirnya, semua model digabungkan untuk menghasilkan prediksi akhir yang lebih akurat.

2.5.2 XGBoost

XGBoost (*Extreme Gradient Boosting*) adalah algoritma *machine learning* berbasis *gradient boosting* yang dikembangkan oleh Chen dan Guestrin (2016). Algoritma ini merupakan penyempurnaan dari metode *boosting* tradisional dengan menggabungkan serangkaian pohon keputusan (*decision trees*) secara berurutan, di mana setiap pohon baru dibangun untuk mengoreksi kesalahan prediksi pohon sebelumnya [35]. XGBoost menggunakan pendekatan optimasi *gradient* untuk meminimalkan fungsi kerugian, yang memungkinkannya mencapai performa tinggi pada tugas klasifikasi dan regresi, termasuk dalam konteks deteksi hate speech pada penelitian ini [39].

XGBoost membentuk model prediksi secara aditif, di mana hasil akhir merupakan penjumlahan dari kontribusi seluruh pohon keputusan yang dibangun secara berurutan. Fungsi prediksinya dituliskan sebagai berikut:

$$\hat{y}_i = \sum_{k=1}^K f_k(x_i) \quad (2.5)$$

dengan K adalah jumlah total pohon, $f_k(x_i)$ merupakan prediksi dari pohon ke- k terhadap input x_i , dan \hat{y}_i adalah hasil akhir prediksi untuk data ke- i . Setiap f_k merupakan fungsi dari ruang pohon regresi \mathcal{F} .

Model ini kemudian dioptimalkan dengan meminimalkan fungsi objektif, yang mencakup dua komponen utama, yaitu fungsi kerugian dan fungsi regularisasi:

$$obj(\theta) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k) \quad (2.6)$$

Komponen $l(y_i, \hat{y}_i)$ berfungsi mengukur selisih antara nilai sebenarnya dan prediksi model, sementara $\Omega(f_k)$ adalah fungsi regulasi yang mengontrol kompleksitas pohon, untuk mencegah overfitting.

Fungsi regulasi tersebut didefinisikan sebagai berikut:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (2.7)$$

dengan T sebagai jumlah daun (terminal nodes) dalam pohon keputusan, w_j sebagai nilai output dari daun ke- j , γ adalah parameter penalti terhadap jumlah daun, dan λ adalah koefisien regularisasi L2 terhadap besar bobot w_j .

Agar proses optimasi lebih efisien, XGBoost mendekati fungsi kerugian menggunakan ekspansi Taylor orde kedua. Dengan pendekatan ini, fungsi objektif pada iterasi ke- t dapat dihamperi sebagai:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^n \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t) \quad (2.8)$$

di mana g_i dan h_i masing-masing adalah turunan pertama (gradien) dan kedua (Hessian) dari fungsi kerugian terhadap prediksi sebelumnya.

Lebih lanjut, dalam proses pembangunan pohon, XGBoost menentukan

pemisahan (split) terbaik berdasarkan nilai *gain*, yaitu seberapa besar peningkatan akurasi yang diperoleh dari membagi satu simpul menjadi dua. Rumus *gain* dirumuskan sebagai:

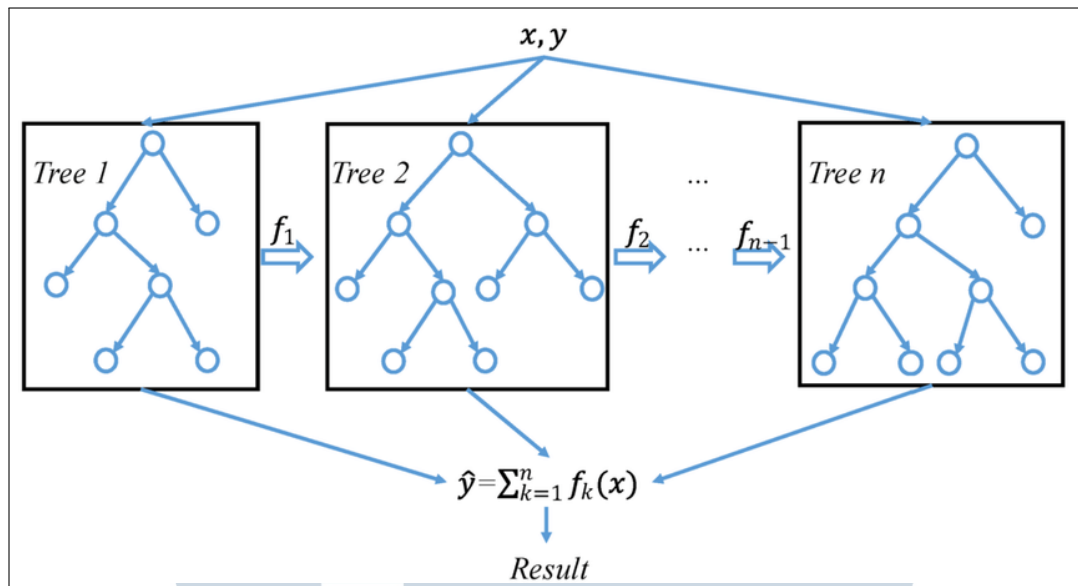
$$Gain = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma \quad (2.9)$$

dengan G_L dan G_R adalah jumlah gradien pada simpul anak kiri dan kanan, H_L dan H_R adalah jumlah Hessian-nya, serta λ dan γ sebagai parameter regularisasi. Nilai *gain* yang tidak melebihi γ akan menyebabkan split dibatalkan, guna menghindari pembentukan model yang terlalu kompleks.

Arsitektur XGBoost berbasis pada konsep *ensemble learning*, di mana model akhir merupakan agregasi dari banyak pohon keputusan lemah. Setiap pohon dilatih dengan mempertimbangkan gradien error dari pohon sebelumnya, yang ditingkatkan dengan fitur seperti regularisasi L1 dan L2 untuk mencegah overfitting, serta dukungan untuk data yang hilang [35]. Algoritma ini juga mendukung pelatihan paralel dan skalabilitas pada dataset besar, menjadikannya dapat mengolah fitur teks yang dihasilkan oleh model bahasa lainnya [40].

Proses kerja XGBoost divisualisasikan pada Gambar 2.4, yang menggambarkan alur pembangunan pohon keputusan secara berurutan. Gambar ini menunjukkan bagaimana input data (x, y) diproses melalui beberapa pohon (*Tree 1*, *Tree 2*, ..., *Tree n*), dengan setiap pohon menghasilkan fungsi prediksi (f_1, f_2, \dots, f_n) yang dikoreksi secara bertahap. Prediksi akhir dihitung sebagai penjumlahan skor dari semua pohon ($\hat{y} = \sum f_i(x)$), yang digunakan untuk klasifikasi teks hate speech berdasarkan fitur dari BERT.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 2.4. Arsitektur XGBoost

Sumber: [41]

Keunggulan XGBoost terletak pada kemampuannya untuk menangani dataset kompleks dan memberikan interpretasi model melalui analisis pentingnya fitur. Selain itu, dalam *Journal of Machine Learning Research* disampaikan bahwa XGBoost unggul dalam kecepatan dan skalabilitas dibandingkan algoritma boosting lain, menjadikannya pilihan yang ideal untuk pengolahan data besar seperti teks media sosial [42].

2.6 Confussion Matrix

Confusion matrix merupakan metode evaluasi performa model klasifikasi yang paling umum digunakan. Confussion matrix menyajikan perbandingan antara hasil prediksi model dan label aktual dalam bentuk tabel dua dimensi. Setiap elemen pada matriks menunjukkan jumlah kasus yang diklasifikasikan ke dalam kombinasi tertentu antara prediksi dan kebenaran aktual, baik untuk kelas positif maupun negatif. Empat komponen utama dalam confusion matrix dapat dijelaskan sebagai berikut:

- **True Positive (TP):** jumlah data positif yang diprediksi benar sebagai positif oleh model.
- **True Negative (TN):** jumlah data negatif yang diprediksi benar sebagai negatif oleh model.

- **False Positive (FP)**: jumlah data negatif yang secara keliru diprediksi sebagai positif.
- **False Negative (FN)**: jumlah data positif yang secara keliru diprediksi sebagai negatif.

Dari keempat nilai ini, sejumlah metrik evaluasi kinerja model dapat dihitung untuk memberikan gambaran lebih komprehensif:

- **Accuracy**, yaitu proporsi prediksi yang benar dari seluruh jumlah prediksi:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (2.10)$$

- **Precision**, yang mengukur seberapa banyak dari prediksi positif yang benar:

$$Precision = \frac{TP}{TP + FP} \quad (2.11)$$

- **Recall** (atau *Sensitivity*), yang menunjukkan seberapa banyak data positif yang berhasil dikenali oleh model:

$$Recall = \frac{TP}{TP + FN} \quad (2.12)$$

- **F1-Score**, yaitu harmonic mean dari precision dan recall, memberikan keseimbangan antara keduanya:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (2.13)$$

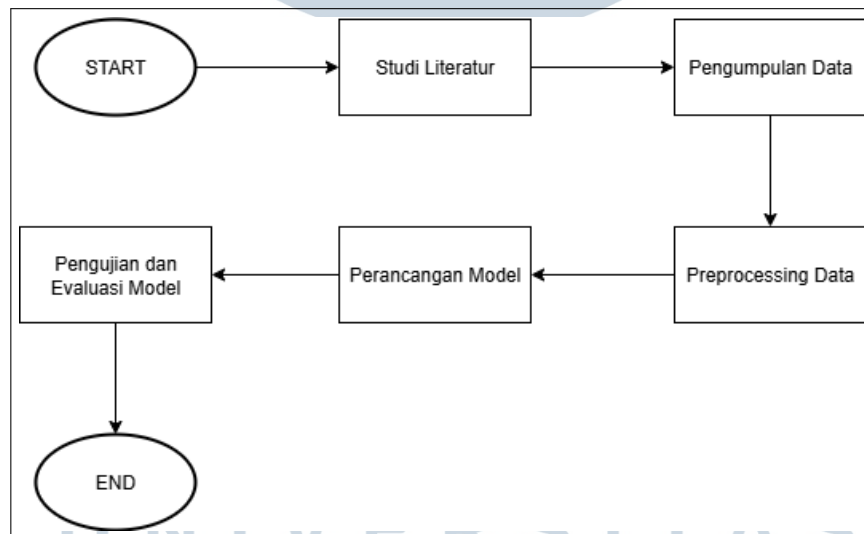
Evaluasi berbasis confusion matrix sangat krusial terutama dalam kasus klasifikasi teks yang tidak seimbang, seperti pada deteksi ujaran kebencian. Dengan memahami distribusi prediksi model terhadap masing-masing kelas, confusion matrix dapat menjadi dasar analisis yang kuat untuk perbaikan model lebih lanjut [43].

BAB 3

METODOLOGI PENELITIAN

Bab ini membahas metodologi penelitian yang digunakan untuk menganalisis ujaran kebencian (*hate speech*) di media sosial X. Penelitian ini menggabungkan model bahasa BERT sebagai pembentuk representasi teks dengan XGBoost sebagai algoritma klasifikasi. Proses penelitian dilakukan secara sistematis, dimulai dari studi literatur, pengumpulan dan preprocessing data, perancangan sistem, implementasi model, pengujian, evaluasi, hingga pelaporan hasil.

Alur metodologi ditunjukkan pada Gambar 3.1. Data yang digunakan berasal dari dataset Okky Ibrohim [44], yang berisi tweet berbahasa Indonesia dan telah dilabeli oleh 30 anotator. Setelah melalui tahap preprocessing, data digunakan dalam perancangan dan implementasi sistem klasifikasi. Evaluasi dilakukan untuk menilai performa model, dan seluruh tahapan didokumentasikan dalam bentuk laporan penelitian



Gambar 3.1. Alur metodologi penelitian

3.1 Studi Literatur

Studi literatur dilakukan sebagai landasan untuk memahami secara komprehensif topik penelitian yang berkaitan dengan ujaran kebencian (*hate speech*) dan penerapan algoritma *machine learning* dalam mendeteksinya.

Penelitian ini meninjau berbagai jurnal ilmiah dan artikel, baik dari sumber nasional maupun internasional, dengan rentang waktu publikasi antara tahun 2016 hingga 2024. Fokus utama kajian tertuju pada fenomena *hate speech* yang terjadi di platform media sosial X, serta pendekatan-pendekatan teoritis dan teknis yang telah dikembangkan untuk deteksi ujaran kebencian. Kajian ini mencakup teori dasar mengenai deteksi *hate speech*, tahapan *text processing*, serta penerapan model *machine learning hybrid* BERT dan XGBoost, serta metode evaluasi kinerja model menggunakan Confusion Matrix.

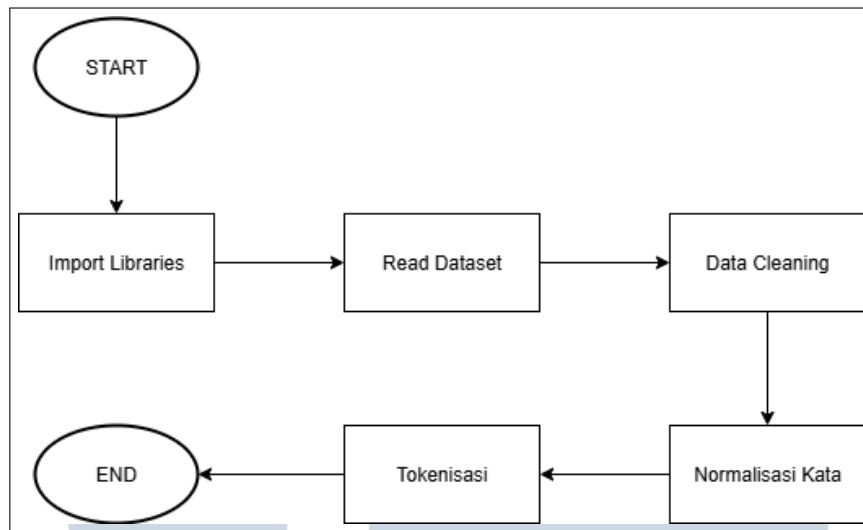
3.2 Pengumpulan Data

Pengumpulan data dalam penelitian ini dilakukan dengan menggabungkan dua sumber utama, yaitu dataset yang telah tersedia dari penelitian sebelumnya oleh Ibrohim dan Budi [44], serta data hasil *scraping* menggunakan *Twitter Search API*. Proses *scraping* dilakukan dengan menggunakan kata kunci dan frasa yang umum digunakan dalam ujaran kebencian dan bahasa kasar, yang telah diidentifikasi berdasarkan studi literatur terdahulu.

Data hasil *crawling* kemudian dikurasi dan dianotasi dalam dua tahap, yakni klasifikasi terhadap *hate speech* dan *abusive language*, serta anotasi lanjutan yang mencakup identifikasi target, kategori, dan tingkat intensitas ujaran kebencian. Untuk menjaga kualitas dan objektivitas data, proses anotasi dilakukan menggunakan pendekatan *crowdsourcing* melalui sistem berbasis web yang dirancang khusus. Sebanyak 30 anotator dengan latar belakang yang beragam dilibatkan dalam proses ini guna meningkatkan validitas dan keandalan hasil anotasi.

3.3 Preprocessing Data

Dataset yang digunakan telah melalui proses pelabelan oleh penyedia data berdasarkan anotasi manual menggunakan 30 anotator. Setelah pelabelan, data memasuki tahap preprocessing untuk memastikan kualitas dan kebersihan data sebelum digunakan sebagai input model XGBoost. Proses preprocessing ini mencakup pembersihan teks dari elemen tidak relevan, normalisasi kata, serta validasi ulang terhadap label yang diberikan. Seluruh tahapan preprocessing ditampilkan pada Gambar 3.2 dan dijelaskan secara detail berikut ini:



Gambar 3.2. Alur Preprocessing Dataset

Tahapan preprocessing pada Gambar 3.2 dapat dijelaskan lebih mendetail sebagai berikut:

Tahapan-tahapan *pre-processing* yang ditunjukkan pada Gambar 3.2 dapat dijelaskan sebagai berikut:

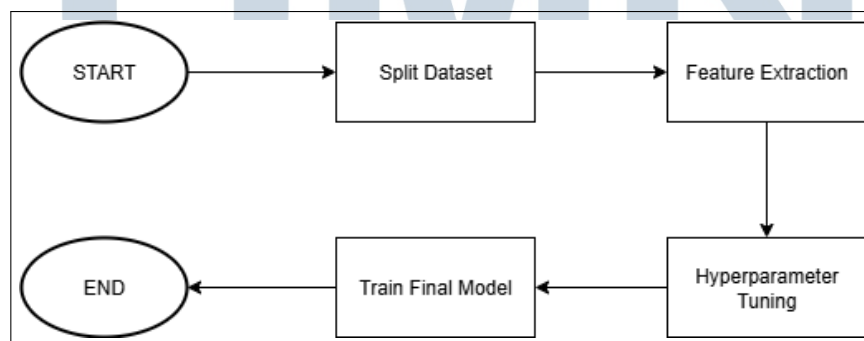
- **Import Libraries:** Tahapan awal berupa pemanggilan seluruh pustaka Python yang dibutuhkan dalam proses analisis data, mencakup pustaka untuk manipulasi data (seperti *pandas*), pemrosesan teks (seperti *re*), serta algoritma pemodelan (seperti *xgboost*) dan libraries lainnya.
- **Read Dataset:** Membaca data tweet yang telah diberi label oleh penyedia dataset. Dataset dimuat ke dalam struktur data yang sesuai, seperti *DataFrame*, agar dapat diproses lebih lanjut.
- **Data Cleaning:** Melakukan pembersihan terhadap konten teks tweet dengan menghapus URL, mention (@), hashtag (#), angka, emoji, simbol, karakter non-ASCII, serta elemen-elemen lain yang tidak relevan terhadap analisis ujaran kebencian.
- **Normalisasi Kata:** Mengonversi kata-kata tidak baku, seperti slang, singkatan, atau kesalahan penulisan, ke bentuk baku menggunakan kamus normalisasi. Proses ini juga mencakup *case folding* (mengubah semua huruf menjadi huruf kecil).

- **Tokenisasi:** Memecah teks tweet menjadi unit-unit kata atau *token*, yang merupakan bentuk representasi awal dari teks sebelum dilakukan ekstraksi fitur. Proses tokenisasi dilakukan menggunakan *BERT tokenizer* untuk menyesuaikan format input dengan kebutuhan model BERT, termasuk penambahan token khusus seperti [CLS] dan [SEP].

Tahapan *preprocessing* dilakukan secara sistematis untuk memastikan data bersih, dan konsisten. Penggunaan *BERT tokenizer* memungkinkan tokenisasi kontekstual yang mempertahankan struktur dan konten semantik teks, sehingga mendukung pembentukan representasi vektor yang optimal bagi model klasifikasi XGBoost.

3.4 Perancangan Model

Perancangan sistem dilakukan untuk membangun model klasifikasi yang mampu mendeteksi ujaran kebencian pada platform X secara otomatis. Penelitian ini menggunakan algoritma XGBoost (Extreme Gradient Boosting), yaitu metode ensemble learning berbasis pohon keputusan yang efektif untuk klasifikasi. Input model berupa representasi vektor hasil ekstraksi embedding dari BERT, yang diperoleh melalui proses tokenisasi menggunakan BERT tokenizer dan agregasi dengan mean pooling. Seluruh tahapan mulai dari preprocessing data, ekstraksi fitur, hingga pelatihan model dirancang secara sistematis untuk menghasilkan klasifikasi yang akurat. Alur lengkap perancangan sistem divisualisasikan dalam Gambar 3.3.



Gambar 3.3. Alur Perancangan Model

Langkah-langkah perancangan model yang ditunjukkan pada Gambar 3.3 dapat dijelaskan sebagai berikut:

- **Split Dataset:** Dataset dibagi menjadi data latih, data validasi dan data uji. Pembagian ini dilakukan untuk memisahkan data yang digunakan dalam pelatihan model dan data yang digunakan untuk evaluasi agar proses pengujian lebih objektif.
- **Feature Extraction:** Representasi fitur dari data dihasilkan pada tahap ini. Dalam konteks model BERT, setiap teks akan diubah menjadi representasi vektor berbasis embedding yang mencerminkan konteks semantik dari kalimat. Dalam memperoleh representasi satu vektor per kalimat, digunakan teknik *mean pooling* terhadap semua token, dengan mengecualikan token khusus seperti [CLS] dan [SEP]. Hasilnya adalah vektor berdimensi tinggi yang merepresentasikan konteks keseluruhan kalimat, dan digunakan sebagai fitur input pada tahap klasifikasi selanjutnya.
- **Hyperparameter Tuning:** Proses ini bertujuan untuk mencari kombinasi parameter terbaik dari algoritma XGBoost guna meningkatkan performa model. Parameter seperti *learning rate*, *max depth*, *n estimators*, *subsample*, dan *colsample bytree* disesuaikan secara manual berdasarkan hasil evaluasi terhadap data latih.
- **Train Final Model:** Model dilatih ulang menggunakan data latih dengan konfigurasi hyperparameter yang telah disesuaikan secara manual berdasarkan eksplorasi dari inisialisasi awal. Model ini merupakan versi final yang digunakan dalam tahap evaluasi maupun prediksi.

3.4.1 Split Dataset

Pembagian dataset dilakukan untuk memisahkan data ke dalam tiga subset, yaitu data latih (*training set*), data validasi (*validation set*), dan data uji (*testing set*) dengan rasio 70:15:15. Pembagian ini bertujuan untuk memastikan proses pelatihan model berlangsung optimal serta evaluasi performa model dilakukan secara objektif.

Data latih digunakan untuk melatih model, data validasi digunakan untuk menyetel parameter dan menghindari *overfitting*, sedangkan data uji digunakan untuk mengukur performa akhir model terhadap data yang belum pernah dilihat sebelumnya.

Pembagian dataset dilakukan secara acak dengan mempertimbangkan distribusi kelas melalui metode *stratified split*, sehingga proporsi kelas tetap

seimbang pada ketiga subset data. Implementasi teknis pembagian ini dilakukan menggunakan fungsi `train_test_split` dari pustaka `scikit-learn` secara bertahap, dengan parameter `stratify=y` untuk menjaga keseimbangan label.

3.4.2 Feature Extraction

Proses ekstraksi fitur dalam penelitian ini dilakukan dengan memanfaatkan model BERT (Bidirectional Encoder Representations from Transformers) sebagai *feature extractor*. BERT memiliki kemampuan memahami konteks kata dalam kalimat secara mendalam melalui mekanisme *self-attention*, sehingga dapat menghasilkan representasi teks yang lebih kaya secara semantik dibandingkan metode konvensional seperti *TF-IDF*.

Setiap komentar yang telah melalui split diubah menjadi vektor embedding melalui model *pre-trained* BERT. Dalam memperoleh representasi vektor dari suatu komentar secara keseluruhan, digunakan teknik *mean pooling*, yaitu dengan menghitung rata-rata dari semua token embeddings yang dihasilkan BERT. Vektor hasil *mean pooling* ini kemudian digunakan sebagai input fitur untuk model klasifikasi XGBoost.

Pendekatan ini memungkinkan sistem untuk memanfaatkan pemahaman konteks yang dihasilkan oleh BERT tanpa perlu melatih ulang seluruh model, sehingga efisien dalam hal komputasi namun tetap memberikan performa yang baik.

3.4.3 Hyperparameter Tuning

Hyperparameter merupakan parameter yang tidak dipelajari secara langsung oleh model dari data, melainkan harus ditentukan sebelum proses pelatihan. Pada model XGBoost, beberapa *hyperparameter* penting yang memengaruhi kompleksitas model dan performa akhir antara lain kedalaman maksimum pohon (`max_depth`), laju pembelajaran (`learning_rate`), serta jumlah pohon estimasi (`n_estimators`) [35]. Pemilihan nilai parameter yang tepat dapat meningkatkan kemampuan generalisasi model serta menghindari masalah *overfitting* maupun *underfitting*.

Penyesuaian *hyperparameter* secara sistematis dikenal sebagai *hyperparameter tuning*, yaitu proses pencarian kombinasi nilai optimal dari parameter-parameter tertentu untuk memaksimalkan kinerja model. Dibandingkan penyetelan manual yang bersifat subjektif, tuning otomatis menawarkan pendekatan

yang lebih terstruktur dan reproducible, sehingga menghasilkan performa model yang lebih stabil dan konsisten [45].

Metode tuning yang digunakan dalam penelitian ini adalah *Grid Search*, yaitu pendekatan yang mengevaluasi seluruh kombinasi parameter dalam ruang pencarian yang telah ditentukan secara eksplisit. Proses ini diimplementasikan menggunakan fungsi `GridSearchCV` dari pustaka `scikit-learn`, yang secara otomatis melakukan pencarian parameter terbaik dengan pendekatan *cross-validation*. Evaluasi setiap kombinasi parameter dilakukan berdasarkan metrik F1-score sebagai skor utama [46].

Meskipun *Grid Search* cenderung lebih lambat dibandingkan pendekatan seperti *Random Search*, metode ini menjamin bahwa seluruh kombinasi dalam ruang pencarian diuji, sehingga kemungkinan menemukan konfigurasi terbaik menjadi lebih tinggi. Dalam penelitian ini, ruang pencarian parameter dibatasi pada nilai-nilai yang umum digunakan untuk XGBoost, seperti $\text{max_depth} = \{4, 6, 8\}$, $\text{learning_rate} = \{0.01, 0.1, 0.2\}$, dan $\text{n_estimators} = \{100, 200\}$, agar waktu komputasi tetap efisien namun hasil tuning tetap optimal.

3.5 Pengujian dan Evaluasi

Tahap pengujian dan evaluasi pada model penelitian digunakan untuk menilai tingkat kinerja model dalam klasifikasi tweet ke dalam kategori yang sesuai khususnya dalam konteks ujaran kebencian. Evaluasi performa model dilakukan menggunakan metode *confusion matrix*. Confusion matrix memperlihatkan jumlah prediksi yang benar maupun salah dalam tiap kategori, terdiri atas empat komponen utama: *True Positive (TP)*, *True Negative (TN)*, *False Positive (FP)*, dan *False Negative (FN)*. Berdasarkan nilai-nilai ini, sejumlah metrik performa utama dapat dihitung sebagai berikut:

- **Accuracy** mengukur proporsi prediksi yang benar dari seluruh prediksi.
- **Precision** menilai seberapa banyak dari prediksi positif yang benar-benar positif.
- **Recall** atau sensitivitas menunjukkan sejauh mana model berhasil mengenali data positif.
- **F1-Score** merupakan rata-rata harmonik dari precision dan recall, digunakan untuk memberikan keseimbangan antara keduanya terutama pada kasus data

yang tidak seimbang.

Evaluasi ini bersifat kuantitatif, bertujuan untuk mengukur sejauh mana model mampu melakukan klasifikasi dengan benar. Seluruh perhitungan metrik dilakukan secara otomatis menggunakan fungsi evaluasi bawaan dari pustaka `scikit-learn`, yang telah teruji dalam banyak penelitian *Natural Language Processing* dan klasifikasi teks. Selain itu, digunakan juga visualisasi confusion matrix untuk memberikan gambaran distribusi prediksi model secara intuitif.



BAB 4

HASIL DAN DISKUSI

Bab Hasil dan Diskusi menyajikan implementasi algoritma *hybrid* BERT dan XGBoost dalam mengklasifikasikan tweet yang mengandung ujaran kebencian pada platform X. Hasil penelitian disampaikan secara sistematis, dimulai dari penggunaan dataset, proses *preprocessing* data, pembagian data menjadi data latih dan data uji, pelatihan model *hybrid*, dan evaluasi performa model.

4.1 Spesifikasi Sistem

Penelitian ini dilaksanakan dengan menggunakan perangkat keras dan perangkat lunak yang mendukung proses eksperimen dan pengolahan data. Spesifikasi sistem digunakan untuk memastikan bahwa proses komputasi, khususnya pada tahap pelatihan model BERT dan XGBoost yang membutuhkan daya pemrosesan tinggi, dapat berjalan secara optimal. Adapun spesifikasi sistem yang digunakan adalah sebagai berikut:

1. Hardware

- Processor: AMD Ryzen 5 5600X (6 Core, 12 Thread)
- RAM: 16 GB DDR4 3200 MHz
- GPU: NVIDIA GeForce RTX 3060Ti (8 GB GDDR6)
- SSD 1: Samsung SSD 980 500 GB M.2 NVMe
- SSD 2: Pioneer 120 GB Sata 3
- Hard Disk 1: Western Digital Caviar Blue 1 TB SATA 3

2. Software

- Sistem Operasi: Windows 11 Pro
- Anaconda 3
- Jupyter Notebook
- Opera GX

Pemilihan spesifikasi tersebut disesuaikan dengan kebutuhan pemrosesan teks dan pelatihan model *transformers* yang relatif kompleks. Penggunaan GPU

bertujuan untuk mempercepat proses pelatihan model BERT, yang secara signifikan lebih lambat jika hanya menggunakan CPU.

4.2 Deskripsi Dataset

Penelitian ini menggunakan dataset berupa kumpulan tweet berbahasa Indonesia yang telah dilabeli sebagai *hate speech* dan *non hate speech*. Setiap class memiliki value antara 0 dan 1, value 1 mengindikasikan bahwa kolom tersebut positif terindikasi dan value 0 mengindikasikan bahwa kolom negatif terindikasi. Dataset diperoleh dari hasil kurasi dan *preprocessing* data publik yang dikumpulkan dari media sosial Twitter menggunakan Twitter API [44]. Dataset ini berisi total 13.169 tweet yang dianotasi dengan pendekatan *multilabel*, memungkinkan identifikasi lebih dari satu kategori dalam satu tweet. Setiap tweet diberi label berdasarkan dua kategori utama: "HS" (*hate speech*) dan "Abusive" (bahasa kasar). Selain itu, terdapat sub-kategori spesifik untuk *hate speech*, yaitu:

Tabel 4.1. Contoh lima data pertama

No	Tweet	HS	Abusive	HSI	HSG	HSRI	HSRc	HSP	HSGn	HSO	HSW	HSM	HSS
0	- disaat semua cowok berusaha melacak perhatian...	1	1	1	0	0	0	0	0	1	1	0	0
1	RT USER: USER siapa yang telat ngasih tau elu?...	0	1	0	0	0	0	0	0	0	0	0	0
2	41. Kadang aku berfikir, kenapa aku tetap percaya...	0	0	0	0	0	0	0	0	0	0	0	0
3	USER USER AKU ITU AKU KU TAU MATAMU SIPIT...	0	0	0	0	0	0	0	0	0	0	0	0
4	USER USER Kaum cebong kapir udah keliatan dong...	1	1	0	1	1	0	0	0	0	0	1	0

- HS_Individual (ditujukan kepada individu)
- HS_Group (ditujukan kepada kelompok)
- HS_Religion (berkaitan dengan agama/kepercayaan)
- HS_Race (berkaitan dengan ras/etnis)
- HS_Physical (berkaitan dengan fisik/disabilitas)
- HS_Gender (berkaitan dengan gender/orientasi seksual)
- HS_Other (berkaitan dengan fitnah/penistaan lainnya)
- HS_Weak (*hate speech* lemah)

- HS_Moderate (*hate speech* sedang)
- HS_Strong (*hate speech* kuat)

Dataset ini memiliki representativitas yang kuat dalam mendukung pembangunan dan evaluasi sistem deteksi ujaran kebencian pada media sosial berbahasa Indonesia.

4.3 Implementasi Sistem

Bagian ini menjelaskan proses implementasi sistem klasifikasi *hate speech* yang telah dirancang, mulai dari import libraries, *preprocessing* data, hingga validasi dan pengujian model menggunakan *metrics* evaluasi. Implementasi dilakukan menggunakan Python dengan bantuan model algoritma BERT dan XGBoost. Tahapan ini bertujuan untuk merealisasikan rancangan sistem secara teknis dan menguji performanya terhadap data komentar yang telah dikumpulkan.

4.3.1 Import Libraries

Memulai proses implementasi, langkah pertama yang dilakukan adalah mengimpor berbagai pustaka (*library*) yang dibutuhkan. Pustaka-pustaka ini mencakup modul untuk pemrosesan data, pembelajaran mesin, serta visualisasi terlampir pada Kode 4.1.

```

1 import pandas as pd
2 import torch
3 import re
4 import emoji
5 import string
6 import numpy as np
7 import torch
8 import time
9 import optuna
10 import matplotlib.pyplot as plt
11 import torch.nn as nn
12 import torch.nn.functional as F
13 import xgboost as xgb
14 import json
15 import joblib
16 import seaborn as sns
17 from xgboost import XGBClassifier

```

```

18 from xgboost.callback import TrainingCallback
19 from torch.utils.data import Dataset, DataLoader
20 from transformers import AutoTokenizer, AutoConfig,
    AutoModelForSequenceClassification
21 from transformers import BertTokenizer,
    BertForSequenceClassification, AdamW, get_scheduler, BertModel
22 from sklearn.preprocessing import StandardScaler
23 from sklearn.metrics import classification_report, accuracy_score,
    f1_score, precision_score, recall_score, roc_auc_score,
    confusion_matrix, ConfusionMatrixDisplay, log_loss
24 from sklearn.model_selection import train_test_split, GridSearchCV
25 from collections import Counter
26 from tqdm.auto import tqdm

```

Kode 4.1: Import libraries

Pustaka-pustaka tersebut menyediakan fungsi-fungsi penting yang akan digunakan dalam seluruh tahapan penelitian, mulai dari pembersihan data, pembentukan dataset, pelatihan model, evaluasi performa, hingga visualisasi hasil. Dengan melakukan impor diawal, seluruh modul yang diperlukan telah tersedia dan dapat dipanggil sesuai kebutuhan pada bagian-bagian implementasi berikutnya.

4.3.2 Preprocessing Data

Proses *preprocessing* merupakan tahap krusial dalam pemodelan teks karena secara langsung memengaruhi kualitas data yang digunakan untuk pelatihan model. Pada tahap ini, data mentah dibersihkan dari elemen-elemen yang tidak relevan. Selain itu, dilakukan juga standarisasi teks untuk memastikan konsistensi format, yang penting dalam pemrosesan bahasa alami.

A Menampilkan Dataset

Dalam proses analisis data, langkah awal yang dilakukan adalah pemeriksaan awal (*initial inspection*) untuk memahami struktur serta karakteristik data secara menyeluruh.

```

1 # Show top of dataset
2 df = pd.read_csv('./multilabel-dataset/re_dataset.csv', encoding='
    latin1')
3 df.head()

```

Kode 4.2: Pemeriksaan dataset

Tahapan ini bertujuan untuk memuat dataset awal dan menampilkan struktur umum dari dataset, meliputi jumlah entri, jumlah kolom, serta distribusi awal label yang menjadi fokus analisis. Selanjutnya, dilakukan proses penghapusan kolom-kolom yang tidak relevan terhadap tugas klasifikasi *hate speech*, guna menyederhanakan struktur data dan meningkatkan efisiensi analisis. Potongan Kode 4.3 merupakan fungsi untuk melakukan menghilangkan kolom yang tidak digunakan.

```

1 # Remove columns yang tidak diperlukan
2 df.drop(columns = ['Abusive',
3                    'HS_Individual',
4                    'HS_Group',
5                    'HS_Religion',
6                    'HS_Race',
7                    'HS_Physical',
8                    'HS_Gender',
9                    'HS_Other',
10                   'HS_Weak',
11                   'HS_Moderate',
12                   'HS_Strong'], inplace=True)

```

Kode 4.3: Potongan kode *remove* kolom

Tabel 4.2. Hasil *remove* kolom

	Tweet	HS
0	- disaat semua cowok berusaha melacak perhatian...	1
1	RT USER: USER siapa yang telat ngasih tau elu?...	0
2	41. Kadang aku berfikir, kenapa aku tetap percaya...	0
3	USER USER AKU ITU AKU KU TAU MATAMU SIPIT...	0
4	USER USER Kaum cebong kapir udah keliatan dong...	1

Pada Tabel 4.2 disajikan perubahan dataset setelah kolom yang tidak dibutuhkan dihapus, tahap berikutnya adalah melakukan pemeriksaan terhadap nilai-nilai kosong (*null values*). Langkah ini penting karena keberadaan nilai kosong dapat menyebabkan *error* pada saat pelatihan model. Selain itu, duplikasi data juga dihapus untuk memastikan bahwa setiap entri bersifat unik dan tidak memberikan bias terhadap hasil model. Potongan Kode 4.4 merupakan kode untuk *check null* dan menghilangkan duplikasi pada kolom Tweet.

```

1 # Check is null value
2 df.isnull().sum()
3
4 # Remove duplicates
5 df = df.drop_duplicates(subset=['Tweet'])

```

Kode 4.4: Cek *null* dan hapus duplikasi

B Cleaning Text

Tahap selanjutnya, dilakukan proses pembersihan teks (*text cleaning*) pada kolom Tweet. Proses ini mencakup serangkaian transformasi untuk meningkatkan kualitas data teks sebelum digunakan dalam pelatihan model. Beberapa langkah yang diterapkan meliputi: konversi seluruh karakter ke huruf kecil, penghapusan tautan (URL), tagar (*hashtag*), karakter *unicode*, simbol, kata spesifik "USER", karakter non-ASCII serta spasi berlebih (*whitespace*) yang dihilangkan untuk mengurangi *noise* pada teks.

Tahap pembersihan ini juga mencakup normalisasi kata slang atau alay menggunakan kamus eksternal, sehingga kata-kata tidak baku dapat dikonversi ke bentuk formal yang lebih representatif. Potongan kode yang menunjukkan implementasi berbagai fungsi dalam proses *preprocess_text* ditampilkan pada Kode 4.5.

```

1 # to lower
2 def to_lower(text):
3     return text.lower()
4
5 # remove URLs
6 def remove_URL(text):
7     text = re.sub(r'https?:\/\/\S+|www\.\S+', '', text)
8     return text
9
10 # remove mention and hashtag
11 def remove_hashtag(text):
12     text = re.sub(r'@\w+', '', text)
13     text = re.sub(r'#\w+', '', text)
14     return text.strip()
15
16 # remove words like "\x9f\x80\xA0"
17 def normalize_unicode(text):
18     return re.sub(r'(\x[da-fA-F]{2})+', '', text)

```

```

19
20 # remove symbols
21 def remove_punctuation(text):
22     text = re.sub(f"[{re.escape(string.punctuation)}]", " ", text)
23     # Hilangkan tanda baca standar
24     text = re.sub(r"[^\w\s]", " ", text) # Hilangkan simbol non
    -alfanumerik selain spasi
25     text = re.sub(r"\s+", " ", text).strip() # Hapus spasi
    berlebih
26     return text
27
28 # remove repetitive character
29 def remove_repetitive_char(text):
30     return re.sub(r'(\1{2,})', r'\1', text)
31
32 # remove repetitive words
33 def remove_repetitive_words(text):
34     text = re.sub(r'\b(\w+)(\1)+\b', r'\1', text)
35     return text
36
37 # remove whitespace (spasi)
38 def remove_whitespace(text):
39     # Perbaiki escape sequence literal
40     text = re.sub(r"/t", "\t", text) # Ganti "/t" dengan tab asli
41
42     # Gabungkan whitespace yang berlebihan secara selektif
43     text = re.sub(r" +", " ", text) # Hapus spasi berlebih
44     text = re.sub(r"\n+", "\n", text) # Hapus newline berlebih
45     text = re.sub(r"\r+", "\r", text) # Hapus carriage return
    berlebih
46     text = re.sub(r"\t+", "\t", text) # Hapus tab berlebih
47     return text.strip(" ")
48
49 # normalize alay words using dictionary
50 def normalize_alay(text, alay_dict=alay_dict):
51     return ' '.join([alay_dict.get(word, word) for word in text.
    split()])
52
53 # remove user Words
54 def remove_user_words(text):
55     return re.sub(r'\buser\b', '', text)
56
57 # remove retweet

```

```

57 def remove_retweet(text):
58     return re.sub(r'^\s*RT\s*[:\-\ ]?\s*', '', text)
59
60 # remove non-ascii
61 def remove_non_ascii(text):
62     return re.sub(r'^\x00-\x7f]', r'', text)
63
64 # remove emojis dan emojis classic
65 def clean_emojis_and_noise(text):
66     text = ''.join(char for char in text if char not in emoji.
        EMOJI_DATA) # Hapus emoji dengan libraries
67     text = re.sub(r'[:;=8][\-\o*\']?[\]\]\(dDpPxX/\|]\]', '', text)
        # Hapus emotikon klasik seperti :) :( :-D
68     text = re.sub(r'[ ]', '', text) # Hapus tanda
        kutip atau kurung aneh
69     return text

```

Kode 4.5: Fungsi-fungsi *cleaning text*

Setelah mendefinisikan berbagai fungsi untuk pembersihan teks, seluruh fungsi tersebut dikemas ke dalam satu fungsi utama bernama *preprocess_text* berguna mempermudah proses penerapan secara menyeluruh. Fungsi *preprocess_text* terlampir pada Kode 4.6, kode ini kemudian digunakan untuk membersihkan data tweet, dan hasil pembersihannya disimpan dalam kolom baru bernama *clean_text*. Selanjutnya, data yang telah melalui tahap praproses ini disimpan ke dalam file CSV baru untuk digunakan pada tahap analisis berikutnya.

```

1 # Preprocess function
2 def preprocess_text(text, alay_dict):
3     text = to_lower(text)
4     text = remove_URL(text)
5     text = remove_retweet(text)
6     text = remove_hastag(text)
7     text = normalize_unicode(text)
8     text = remove_non_ascii(text)
9     text = remove_repetitive_char(text)
10    text = remove_whitespace(text)
11    text = remove_punctuation(text)
12    text = remove_user_words(text)
13    text = normalize_alay(text, alay_dict)
14    return text
15
16 # Cleaning Tweet dan save in clean_text column

```

```

17 df['clean_text'] = df['Tweet'].apply(lambda x: preprocess_text(x,
    alay_dict))
18
19 # Save to csv
20 df.to_csv("cleaned_dataset.csv", index=False)
21 print("Cleaning selesai! Dataset disimpan sebagai cleaned_dataset.
    csv")

```

Kode 4.6: Fungsi *preprocessing*

Tabel 4.3. Data sebelum dan sesudah *cleaning*

No	Tweet	HS	clean_text
0	- disaat semua cowok berusaha melacak perhatian...	1	di saat semua cowok berusaha melacak perhatian...
1	RT USER: USER siapa yang telat ngasih tau elu?...	0	rt siapa yang telat memberi tau kamu edan sara...
2	41. Kadang aku berfikir, kenapa aku tetap percaya...	0	41 kadang aku berpikir kenapa aku tetap percaya...
3	USER USER AKU ITU AKU KU TAU MATAMU SIPIT...	0	aku itu aku dan ku tau matamu sipit tapi dilihat...
4	USER USER Kaum cebong kapir udah keliatan dong...	1	kaum cebong kafir sudah kelihatan dongoknya dan...

Pada Tabel 4.3 menggambarkan hasil transformasi teks sebelum dan sesudah dilakukan proses pembersihan. Perbandingan ini menunjukkan bahwa fungsi praproses yang diterapkan berhasil mengurangi berbagai elemen dan *noise* tidak relevan yang dapat mengganggu kinerja model. Pembersihan ini bertujuan untuk menyederhanakan struktur teks sekaligus mempertahankan konteks semantik utama, sehingga model pembelajaran mesin dapat lebih efektif dalam mengenali pola-pola ujaran kebencian.

C Deteksi dan Koreksi Mislabeled

Setelah proses pembersihan teks dari berbagai *noise* yang dapat mengganggu kinerja model deteksi, tahap selanjutnya adalah melakukan deteksi ulang dan koreksi terhadap label dengan memanfaatkan list kata kasar (*abusive*) yang terbesar dalam menentukan kategori *hate speech*. Proses pembaruan label ini

dilakukan dengan memuat kamus eksternal serta file CSV yang telah melalui tahap *preprocessing*. Selanjutnya, dilakukan proses filterisasi untuk mengidentifikasi apakah dalam teks terdapat indikasi ujaran kebencian sesuai dengan entri dalam kamus tersebut. Potongan Kode 4.7 menunjukkan implementasi proses filterisasi untuk mengidentifikasi label yang berpotensi termasuk dalam kategori *hate speech*.

```

1 # Load clean dataset & show
2 df_2 = pd.read_csv('cleaned_dataset.csv')
3 df_2 = df_2.drop(columns=['Tweet'])
4 df_2.tail()
5
6 # Check is null value
7 df_2 = df_2.dropna(subset=['clean_text'])
8 df_2.isnull().sum()
9
10 # Tambahkan Abusive -> Hate-Speech
11 # Load kamus kata kasar dari eksternal dictionary
12 with open('./kata_kasar.txt', 'r', encoding='utf-8') as f:
13     daftar_kasar = [baris.strip().lower() for baris in f if baris.
14                     strip()]
15
16 # Cek distribusi label 0 yang mengandung kata kasar
17 def mengandung_kata_kasar(teks, daftar_kasar):
18     return any(kata in teks.lower().split() for kata in
19               daftar_kasar)
20
21 # Filter data
22 potensi_mislabeled = df_2[(df_2['HS'] == 0) & (df_2['clean_text'].
23         apply(lambda x: mengandung_kata_kasar(str(x), daftar_kasar)))]
24
25 # Simpan untuk audit manual
26 potensi_mislabeled.to_csv('potensi_mislabeled.csv', index=False)
27 print(f"Jumlah potensi mislabeling ditemukan: {len(
28       potensi_mislabeled)}")

```

Kode 4.7: Deteksi *mislabeled*

Tabel 4.4. Ringkasan jumlah data dan hasil audit label

Tahap	Jumlah Data
Total data awal	13.014
Potensi mislabeling ditemukan	1.030

Setelah mengidentifikasi sejumlah data yang berpotensi mengalami *mislabeling*, terdapat 1030 data yang mengandung ujaran kebencian. Sejumlah data yang terindikasi sebagai *hate speech* disimpan ke dalam file CSV dengan 50 sample untuk dilakukan proses audit lebih lanjut. Setelah hasil audit manual menunjukkan bahwa alasan pelabelan ulang valid dikarenakan terindikasi ujaran kata kasar di dalamnya dan hal tersebut sesuai dengan kriteria *hate speech*, maka semua data tersebut diperbarui dan dimasukkan ke dalam kategori *hate speech* pada kolom label yang bersangkutan. Potongan Kode 4.8 menyajikan penyimpanan potensial hasil deteksi ke dalam CSV sebanyak 50 sample, lalu mengupdate kolom dari kategori 'HS' 0 menjadi 1.

```

1 # Sample untuk audit data potensial mislabel
2 sample = potensi_mislabel.sample(50, random_state=42)
3 sample.to_csv('audit_sample.csv', index=False)
4
5 # Update label HS 0 menjadi 1 untuk data yang mengandung kata
  kasar
6 df_2.loc[potensi_mislabel.index, 'HS'] = 1
7
8 # Simpan DataFrame yang sudah diperbarui jika perlu
9 df_2.to_csv('df_2_terupdate.csv', index=False)
10 print("Label telah diperbarui untuk data yang mengandung kata
    kasar.")

```

Kode 4.8: *Update label*

Tabel 4.5. Distribusi label *hate speech* sebelum dan sesudah *relabeling*

Label HS	Sebelum Relabeling	Sesudah Relabeling
0 (Bukan Hate Speech)	7.516	6.486
1 (Hate Speech)	5.498	6.528

Berdasarkan hasil *relabeling* yang ditampilkan pada Tabel 4.6, dapat diamati adanya pergeseran distribusi label yang signifikan, khususnya pada peningkatan jumlah data yang dikategorikan sebagai *Hate Speech*. Proses ini bertujuan untuk mengkategorikan tweet *abusive* atau mengandung kata kekerasan menjadi kategori ujaran kebencian (*hate speech*).

4.3.3 Perancangan Model

Pada tahap ini, dilakukan proses integrasi antara representasi fitur dari model BERT dengan algoritma klasifikasi XGBoost. Model BERT digunakan untuk menghasilkan representasi vektor dari setiap teks melalui proses vektor *embedding*, yang kemudian diekstraksi dari lapisan akhir BERT. Vektor-vektor ini selanjutnya digunakan sebagai input bagi model XGBoost untuk melakukan klasifikasi terhadap label *hate speech*.

A Split Dataset

Dalam memastikan bahwa model memperoleh distribusi data yang seimbang selama pelatihan dan evaluasi, dilakukan proses pembagian dataset menggunakan teknik *stratified split*. Teknik ini menjaga proporsi label dalam subset data tetap konsisten dengan distribusi asli dataset. Dataset dibagi menjadi tiga bagian, yaitu data pelatihan (70%), data validasi (15%), dan data uji (15%). Pembagian ini dilakukan dalam dua tahap, di mana tahap pertama memisahkan data pelatihan dari data sementara (30%), dan tahap kedua membagi data sementara secara merata menjadi data validasi dan data pengujian. Potongan Kode 4.9 menunjukkan implementasi proses pembagian data tersebut.

```
1 # Stratified split (tetap menjaga distribusi label dengan seimbang
  )
2 # Split 70% train dan 30% temp validation
3 train_idx, temp_idx = train_test_split(
4     list(range(len(df_2))), test_size=0.3, random_state=42,
5     stratify=df_2['HS']
6 )
7 # Ambil label dari temp untuk stratified split kedua
8 temp_labels = [df_2['HS'].iloc[i] for i in temp_idx]
9
10 # Split temp menjadi validation dan test (masing-masing 15%)
11 val_idx, test_idx = train_test_split(
12     temp_idx, test_size=0.5, random_state=42, stratify=temp_labels
13 ) # 30% . 0.5 = 15%
14
15 # Check jumlah data
16 print("Jumlah data:")
17 print("Train      :", len(train_idx))
18 print("Validation:", len(val_idx))
```

```
19 print("Test      :", len(test_idx))
```

Kode 4.9: Split dataset

Tabel 4.6. Distribusi dataset

Data	Jumlah Data
Train	9.116
Validation	1.953
Test	1.954

Pada Kode 4.10 ditampilkan proses menampilkan distribusi dataset hasil split. Setelah dilakukan proses pembagian data menggunakan metode *stratified split*, diperoleh distribusi data sebesar 9.116 data untuk pelatihan (*train*), 1.953 data untuk validasi, dan 1.954 data untuk pengujian (*test*). Metode ini digunakan untuk memastikan proporsi label yang seimbang pada setiap subset data, sehingga mengurangi potensi bias selama proses pelatihan dan evaluasi model. Langkah selanjutnya adalah melakukan verifikasi terhadap hasil pembagian tersebut. Verifikasi ini mencakup pemeriksaan distribusi label pada masing-masing subset, yaitu data pelatihan, validasi, dan pengujian, guna memastikan bahwa distribusi kelas tetap seimbang. Selain itu, dilakukan analisis terhadap rata-rata panjang teks pada ketiga subset untuk mengetahui sejauh mana kompleksitas input teks tersebar secara merata di seluruh bagian data.

```
1 # Count distribusi label train / val / test
2 print("Train:", Counter(train_df['HS']))
3 print("Validation:", Counter(val_df['HS']))
4 print("Test:", Counter(test_df['HS']))
5
6 # check avg panjang text train / val / test
7 train_df['length'] = train_df['clean_text'].apply(lambda x: len(x.
    split()))
8 val_df['length'] = val_df['clean_text'].apply(lambda x: len(x.
    split()))
9 test_df['length'] = test_df['clean_text'].apply(lambda x: len(x.
    split()))
10
11 print("Train avg len:", train_df['length'].mean())
12 print("Val avg len:", val_df['length'].mean())
13 print("Test avg len:", test_df['length'].mean())
```

Kode 4.10: Menampilkan distribusi dataset setelah split

Hasil dari proses pembagian data ini menunjukkan bahwa distribusi label pada masing-masing subset tetap terjaga secara proporsional. Pemeriksaan terhadap jumlah data dan distribusi kelas pada data pelatihan, validasi, dan pengujian memastikan bahwa tidak terdapat ketimpangan yang signifikan yang dapat memengaruhi performa model. Selain itu, rata-rata panjang teks yang relatif serupa pada ketiga subset menunjukkan bahwa kompleksitas input teks tersebar secara merata.

B Feature Extraction

Pada tahap ini, proses representasi teks dilakukan dengan memanfaatkan model bahasa berbasis *transformer*, yaitu IndoBERT-Base, yang diimplementasikan melalui pustaka *Hugging Face Transformers*. Tujuan utama dari tahap ini adalah mengubah data teks yang telah dibersihkan menjadi representasi numerik berdimensi tetap, yang dapat digunakan sebagai fitur input untuk algoritma klasifikasi pada tahap selanjutnya. Model IndoBERT-Base dipilih karena dilatih secara khusus pada korpus berbahasa Indonesia dan memiliki arsitektur BERT-Base dengan dimensi vektor sebesar 768, sehingga mampu menangkap informasi semantik secara lebih mendalam dibandingkan dengan model yang berukuran lebih kecil.

Langkah pertama dalam proses ini adalah memuat *tokenizer* dan model IndoBERT-Base dari repositori `indobenchmark/indobert-base-pl`. Selanjutnya, setiap teks yang terdapat dalam dataset dikonversi menjadi token menggunakan tokenizer tersebut dan dipetakan ke dalam format tensor yang kompatibel dengan model BERT. Proses tokenisasi ini dilakukan dengan menetapkan panjang maksimum 128 token dan menerapkan *padding* serta *truncation* secara otomatis untuk menjaga konsistensi input.

Setelah teks dikonversi, representasi vektor diperoleh dengan mengambil keluaran dari lapisan terakhir model BERT, yaitu `last_hidden_state`, yang memiliki bentuk tensor berdimensi `(sequence_length, hidden_size)`. Untuk mereduksi dimensi menjadi satu vektor tetap per teks, digunakan teknik *mean pooling* hanya terhadap token yang valid (tidak dipadding), dengan memperhatikan `attention_mask`. Hasil akhir dari proses ini adalah sebuah vektor berdimensi 768 yang merepresentasikan informasi semantik dari masing-masing teks secara keseluruhan.

```
1 # Load IndoBERT
```

```

2 tokenizer = BertTokenizer.from_pretrained('indobenchmark/indobert-
  base-pl')
3 bert_model = BertModel.from_pretrained('indobenchmark/indobert-
  base-pl')
4 bert_model.eval()
5 bert_model.to(device)
6 hidden_size = bert_model.config.hidden_size
7
8 # Fungsi untuk ekstrak embedding dengan mean pooling
9 def get_bert_embedding(text):
10     inputs = tokenizer(
11         text,
12         return_tensors="pt",
13         truncation=True,
14         padding='max_length',
15         max_length=128
16     )
17     inputs = {k: v.to(device) for k, v in inputs.items()}
18
19     with torch.no_grad():
20         outputs = bert_model(**inputs)
21
22     # Ambil seluruh token embeddings dan attention mask
23     last_hidden_state = outputs.last_hidden_state.squeeze(0)
24     # shape: (seq_len, hidden_size)
25     attention_mask = inputs['attention_mask'].squeeze(0)
26     # shape: (seq_len)
27
28     # Masked mean pooling (hanya token yang bukan padding)
29     masked_embeddings = last_hidden_state * attention_mask.
30     unsqueeze(-1) # (seq_len, hidden_size)
31     sum_embeddings = masked_embeddings.sum(dim=0)
32     count_tokens = attention_mask.sum()
33     mean_embedding = sum_embeddings / count_tokens
34
35     return mean_embedding.cpu().numpy()

```

Kode 4.11: Fungsi ekstraksi *embedding* dengan *mean pooling*

Proses ekstraksi vektor *embeddings* ini kemudian diterapkan terhadap seluruh data pada set pelatihan, validasi, dan pengujian. Setiap baris teks akan dikonversi menjadi satu vektor representasi menggunakan fungsi `get_bert_embedding`. Untuk menjamin stabilitas proses dan menghindari

kegagalan saat pemrosesan teks yang tidak valid atau rusak, fungsi ini dilengkapi dengan penanganan `exception` yang akan mengembalikan vektor nol apabila proses ekstraksi gagal. Embedding yang telah diperoleh dikembalikan dalam bentuk matriks berdimensi (`jumlah_data, 768`).

```
1 # Generate embeddings dari DataFrame
2 def generate_embeddings(dataframe):
3     embeddings = []
4     for text in tqdm(dataframe['clean_text'], desc="Generating
5         embeddings"):
6         try:
7             emb = get_bert_embedding(text)
8         except Exception as e:
9             print(f"Error: {e}")
10            emb = np.zeros(hidden_size) # fallback sesuai model
11            embeddings.append(emb)
12    return np.vstack(embeddings)
13
14 # Hasilkan embedding untuk train, val, test
15 X_train = generate_embeddings(train_df)
16 X_val = generate_embeddings(val_df)
17 X_test = generate_embeddings(test_df)
18
19 # Label target
20 y_train = train_df['HS'].astype(int).values
21 y_val = val_df['HS'].astype(int).values
22 y_test = test_df['HS'].astype(int).values
```

Kode 4.12: Proses *embedding* untuk seluruh dataset

Hasil akhir dari tahap ini adalah matriks *embedding* yang berisi representasi semantik dari setiap teks dalam dataset. Matriks ini digunakan sebagai input fitur untuk model klasifikasi XGBoost pada tahap berikutnya dalam *pipeline* penelitian. Sementara itu, label target (y) telah dipersiapkan dalam bentuk vektor numerik untuk setiap subset data (*train*, *validation*, *test*). Diharapkan bahwa penggunaan representasi vektor dari IndoBERT-base ini mampu menangkap konteks dan makna teks secara lebih akurat, sehingga dapat meningkatkan performa model dalam mengklasifikasikan apakah suatu komentar mengandung ujaran kebencian (*hate speech*) atau tidak.

C Hyperparameter Tuning

Pada tahap hyperparameter tuning, model XGBoost dilatih secara otomatis menggunakan kombinasi parameter yang telah ditentukan sebelumnya melalui pendekatan *Grid Search*. Proses ini bertujuan untuk mengeksplorasi ruang parameter secara sistematis guna memperoleh pijakan awal pemilihan konfigurasi model terbaik berdasarkan metrik evaluasi *F1-score*. Penggunaan metode Grid Search dipilih karena pendekatan ini menjamin pencarian yang menyeluruh terhadap semua kombinasi parameter dalam ruang pencarian yang telah ditentukan, sehingga memberikan kontrol penuh terhadap eksplorasi awal serta cocok digunakan saat ruang parameter relatif terbatas dan dapat ditentukan secara eksplisit. Selain itu, *hypertuning* membantu meminimalkan potensi bias yang muncul dari pemilihan awal parameter secara manual. Proses inisialisasi dan pelatihan model ditunjukkan pada Kode 4.13.

```
1 # Hitung perbandingan kelas untuk imbalanced data (tidak dipakai (
    data balanced))
2 # weight = len(y_train[y_train == 0]) / len(y_train[y_train == 1])
3
4 # Grid parameter
5 param_grid = {
6     'max_depth': [4, 6, 8],
7     'learning_rate': [0.01, 0.1, 0.2],
8     'n_estimators': [100, 200],
9     'subsample': [0.8, 1.0],
10    'colsample_bytree': [0.8, 1.0]
11 }
12
13 xgb = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss
    ')
14
15 grid_search = GridSearchCV(estimator=xgb, param_grid=param_grid,
16                             scoring='f1', cv=3, verbose=1, n_jobs
17                             =-1)
18
19 grid_search.fit(X_train, y_train)
20
21 print("Best Parameters:", grid_search.best_params_)
22 # Best Parameters: {'colsample_bytree': 1.0, 'learning_rate': 0.1,
23    'max_depth': 8, 'n_estimators': 200, 'subsample': 0.8}
```

Kode 4.13: *Hyperparameter* dengan grid search

Berdasarkan hasil pada potongan Kode 4.13 dengan menggunakan *GridSearchCV*, diperoleh kombinasi parameter terbaik sebagai berikut: *colsample_bytree*=1.0, *learning_rate*=0.1, *max_depth*=8, *n_estimators*=200, dan *subsample*=0.8. Kombinasi parameter ini kemudian digunakan untuk melatih ulang model XGBoost dengan harapan dapat meningkatkan kemampuan model dalam mendeteksi komentar bermuatan *hate speech* secara lebih akurat dan seimbang.

D Train Final Model

Setelah memperoleh representasi vektor dari setiap teks melalui *embedding* IndoBERT-base, tahap selanjutnya adalah pelatihan model klasifikasi menggunakan algoritma XGBoost. Potongan Kode 4.14 menjelaskan langkah awal pelatihan model XGBoost dimulai dengan mengonversi data fitur dan label ke dalam format *DMatrix*, yaitu format data internal milik pustaka XGBoost. Format ini dirancang khusus untuk mendukung komputasi paralel serta efisien dalam penggunaan memori. Kemudian, model dikonfigurasi dengan sejumlah parameter yang telah disesuaikan untuk mencapai keseimbangan antara kompleksitas model dan generalisasi terhadap data yang tidak terlihat. Selanjutnya, model dikonfigurasi dengan parameter-parameter penting seperti *max_depth*, *learning_rate*, *subsample*, dan *colsample_bytree* untuk mengendalikan kompleksitas dan mencegah *overfitting*, serta *reg_alpha* dan *reg_lambda* sebagai bentuk regularisasi tambahan guna menjaga kestabilan pelatihan dan kemampuan generalisasi model terhadap data baru.

```
1 # Load DMatrix
2 dtrain = xgb.DMatrix(X_train, label=y_train)
3 dval = xgb.DMatrix(X_val, label=y_val)
4 dtest = xgb.DMatrix(X_test, label=y_test)
5
6 # Parameter model
7 params = {
8     'objective': 'binary:logistic',
9     'max_depth': 4,
10    'learning_rate': 0.1,
11    'subsample': 0.8,
12    'colsample_bytree': 0.8,
13    'min_child_weight': 5,
14    'gamma': 0.5,
15    'reg_alpha': 0.7,
16    'reg_lambda': 2.0,
17    'eval_metric': 'logloss'
```

```
18 }
```

Kode 4.14: Inisialisasi *dmatrix* dan *best parameters*

Potongan Kode 4.15 dibuat guna memantau performa model secara lebih komprehensif selama proses pelatihan, digunakan fungsi callback khusus yang mencatat metrik evaluasi tambahan pada data validasi, yaitu akurasi, presisi, *recall*, dan *F1-score*. Metrik-metrik ini dipilih karena mampu memberikan gambaran yang lebih utuh mengenai efektivitas model, khususnya dalam konteks klasifikasi biner yang tidak selalu seimbang.

```
1 # Hasil evaluasi logloss dan metrik tambahan
2 evals_result = {}
3 history = {
4     'val_loss': [],
5     'val_acc': [],
6     'val_precision': [],
7     'val_recall': [],
8     'val_f1': [],
9
10    'train_loss': [],
11    'train_acc': [],
12    'train_precision': [],
13    'train_recall': [],
14    'train_f1': []
15 }
16
17 # Callback custom untuk logging metrik
18 class CustomMetricLogger(TrainingCallback):
19     def __init__(self, dtrain, y_train, dval, y_val, evals_result,
20                  history):
21         self.dtrain = dtrain
22         self.y_train = y_train
23         self.dval = dval
24         self.y_val = y_val
25         self.evals_result = evals_result
26         self.history = history
27
28     def after_iteration(self, model, epoch, evals_log):
29         # Prediksi untuk train
30         y_train_proba = model.predict(self.dtrain, iteration_range
31                                     =(0, epoch + 1))
32         y_train_pred = (y_train_proba > 0.5).astype(int)
```

```

32         train_acc = accuracy_score(self.y_train, y_train_pred)
33         train_prec = precision_score(self.y_train, y_train_pred,
zero_division=0)
34         train_rec = recall_score(self.y_train, y_train_pred,
zero_division=0)
35         train_f1 = f1_score(self.y_train, y_train_pred,
zero_division=0)
36
37         # Prediksi untuk val
38         y_val_proba = model.predict(self.dval, iteration_range=(0,
epoch + 1))
39         y_val_pred = (y_val_proba > 0.5).astype(int)
40
41         val_acc = accuracy_score(self.y_val, y_val_pred)
42         val_prec = precision_score(self.y_val, y_val_pred,
zero_division=0)
43         val_rec = recall_score(self.y_val, y_val_pred,
zero_division=0)
44         val_f1 = f1_score(self.y_val, y_val_pred, zero_division=0)
45
46         # Simpan log loss dari XGBoost
47         self.history['train_loss'].append(evals_log['train']['logloss'][epoch])
48         self.history['val_loss'].append(evals_log['val']['logloss'][epoch])
49
50         # Simpan metrik train
51         self.history.setdefault('train_acc', []).append(train_acc)
52         self.history.setdefault('train_precision', []).append(
train_prec)
53         self.history.setdefault('train_recall', []).append(
train_rec)
54         self.history.setdefault('train_f1', []).append(train_f1)
55
56         # Simpan metrik val
57         self.history['val_acc'].append(val_acc)
58         self.history['val_precision'].append(val_prec)
59         self.history['val_recall'].append(val_rec)
60         self.history['val_f1'].append(val_f1)
61
62         return False

```

Kode 4.15: *Callback logging metrics evaluasi*

Potongan Kode 4.16 merupakan proses pelatihan yang dilakukan dengan jumlah maksimum iterasi sebesar 2000 boosting rounds. Untuk mencegah pelatihan berlebih (*overfitting*), digunakan teknik *early stopping* dengan batas 10 iterasi. Artinya, pelatihan akan dihentikan secara otomatis jika tidak terdapat peningkatan performa pada data validasi selama 10 iterasi berturut-turut.

```
1 #Training
2 model = xgb.train(
3     params=params,
4     dtrain=dtrain,
5     num_boost_round=2000,
6     evals=[(dtrain, 'train'), (dval, 'val')],
7     early_stopping_rounds=10,
8     evals_result=evals_result,
9     callbacks=[CustomMetricLogger(dval, y_val, evals_result, history)
10         ],
11     verbose_eval=False
12 )
```

Kode 4.16: Pelatihan model xgboost

Tahap pelatihan model XGBoost merupakan komponen krusial dalam keseluruhan pipeline klasifikasi, representasi semantik hasil *embedding* dari IndoBERT-BASE dimanfaatkan secara maksimal untuk membedakan komentar yang mengandung ujaran kebencian dan non ujaran kebencian. Model ini dilatih dengan parameter-parameter yang telah disesuaikan secara optimal guna menyeimbangkan akurasi dan kemampuan generalisasi. Selama proses pelatihan iteratif, evaluasi dilakukan secara berkala dengan mencatat metrik performa seperti *log loss*, *accuracy*, *precision*, *recall*, dan *F1-score* ke dalam struktur *log evals_result* dan *history*, yang selanjutnya dapat dianalisis untuk memantau dinamika pembelajaran model.

E Metrics Pengujian Model

Setelah proses pelatihan selesai, tahap selanjutnya adalah mengevaluasi performa model terhadap data uji yang belum pernah dilihat oleh model sebelumnya. Evaluasi ini bertujuan untuk mengukur kemampuan generalisasi model dalam mengklasifikasikan komentar yang mengandung ujaran kebencian dan yang tidak. Selain menghitung metrik performa utama, dilakukan juga visualisasi menggunakan *confusion matrix* untuk memperoleh gambaran lebih jelas terkait

distribusi prediksi model.

```
1 # === TEST METRICS ===
2 y_test_proba = model.predict(dtest)
3 y_test_preds = (y_test_proba > 0.5).astype(int)
4
5 test_acc = accuracy_score(y_test, y_test_pred)
6 test_prec = precision_score(y_test, y_test_pred, zero_division=0)
7 test_rec = recall_score(y_test, y_test_pred, zero_division=0)
8 test_f1 = f1_score(y_test, y_test_pred, zero_division=0)
9
10 # Tambahkan ke history
11 history['test_acc'] = test_acc
12 history['test_precision'] = test_prec
13 history['test_recall'] = test_rec
14 history['test_f1'] = test_f1
15
16 # Print hasil
17 print("\n=== Evaluation on Test Set ===")
18 print(f"Accuracy : {test_acc:.4f}")
19 print(f"Precision: {test_prec:.4f}")
20 print(f"Recall    : {test_rec:.4f}")
21 print(f"F1 Score  : {test_f1:.4f}")
22
23 # Confusion Matrix - Test Set
24 cm = confusion_matrix(y_test, test_preds)
25 disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels
    =['Non-Hate', 'Hate'])
26
27 fig, ax = plt.subplots(figsize=(6, 6))
28 disp.plot(ax=ax, cmap='Blues', values_format='d')
29 plt.title("Confusion Matrix - Test Set")
30 plt.show()
```

Kode 4.17: Evaluasi *metrics* pada data uji

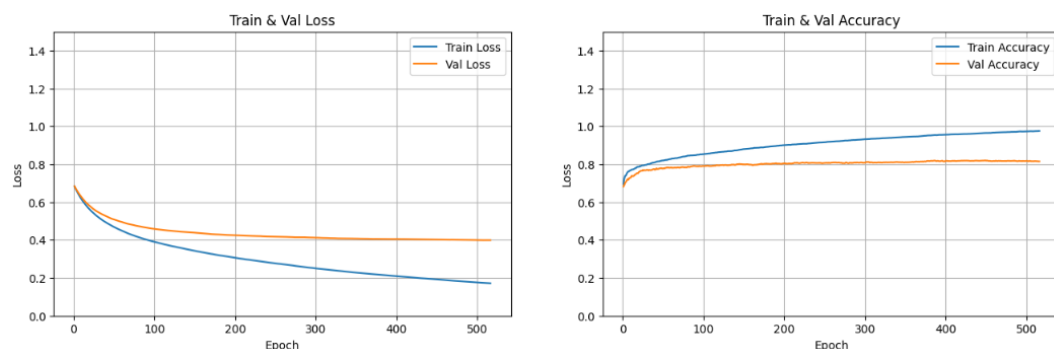
Kode 4.17 menunjukkan proses evaluasi model terhadap data uji, dengan mengukur empat metrik utama yaitu *accuracy*, *precision*, *recall*, dan *F1-score*. Prediksi dilakukan berdasarkan probabilitas yang dihasilkan oleh model XGBoost, dengan ambang batas sebesar 0.5. Seluruh nilai evaluasi disimpan ke dalam struktur *history* untuk kepentingan pelacakan dan analisis lebih lanjut.

4.4 Uji Coba dan Evaluasi Model

Pada bagian ini menjelaskan secara sistematis tahapan pengujian dan evaluasi terhadap model yang telah dikembangkan. Proses uji coba bertujuan untuk menilai kinerja model dalam mengklasifikasikan data sesuai dengan tujuan penelitian. Evaluasi dilakukan dengan menggunakan metrik yang relevan, seperti *accuracy*, *precision*, *recall*, dan *F1-score*, guna memberikan gambaran yang objektif mengenai efektivitas model.

A Hasil Evaluasi Model BERT + XGBoost

Berdasarkan Gambar 4.1 yang merupakan hasil evaluasi model klasifikasi yang diterapkan pada data validasi, diperoleh tingkat akurasi sebesar 81,32%, yang mengindikasikan bahwa proporsi prediksi yang benar terhadap seluruh data mencapai lebih dari 80%. Dalam hal kesalahan prediksi, nilai *log loss* pada data pelatihan tercatat sebesar 0,1609, sedangkan pada data validasi sebesar 0,3889. Pengujian dengan model *hybrid* ini dijadikan pembandingan dengan dua model lainnya sebagai *baseline* model. Nilai evaluasi *metrics* performa lainnya dari model BERT + XGBoost dihasilkan seperti pada Tabel 4.7. Nilai presisi sebesar 81,35% menunjukkan bahwa dari seluruh data yang diprediksi sebagai kelas “*Hate*”, sekitar 81% benar-benar merupakan data *hate*. Sementara itu, nilai *recall* sebesar 81,43% menggambarkan bahwa dari seluruh data aktual yang termasuk ke dalam kelas “*Hate*”, sekitar 81% berhasil dikenali dengan benar oleh model. Nilai *F1-score* sebesar 81,39% mengindikasikan adanya keseimbangan yang baik antara presisi dan *recall* dalam kinerja model. Selain itu, terdapat nilai lainnya yaitu AUC-ROC sebesar 0,8961 memperlihatkan kemampuan model yang tinggi dalam membedakan antara dua kelas (*Hate* dan *NonHate*) mendekati nilai maksimum yaitu 1.

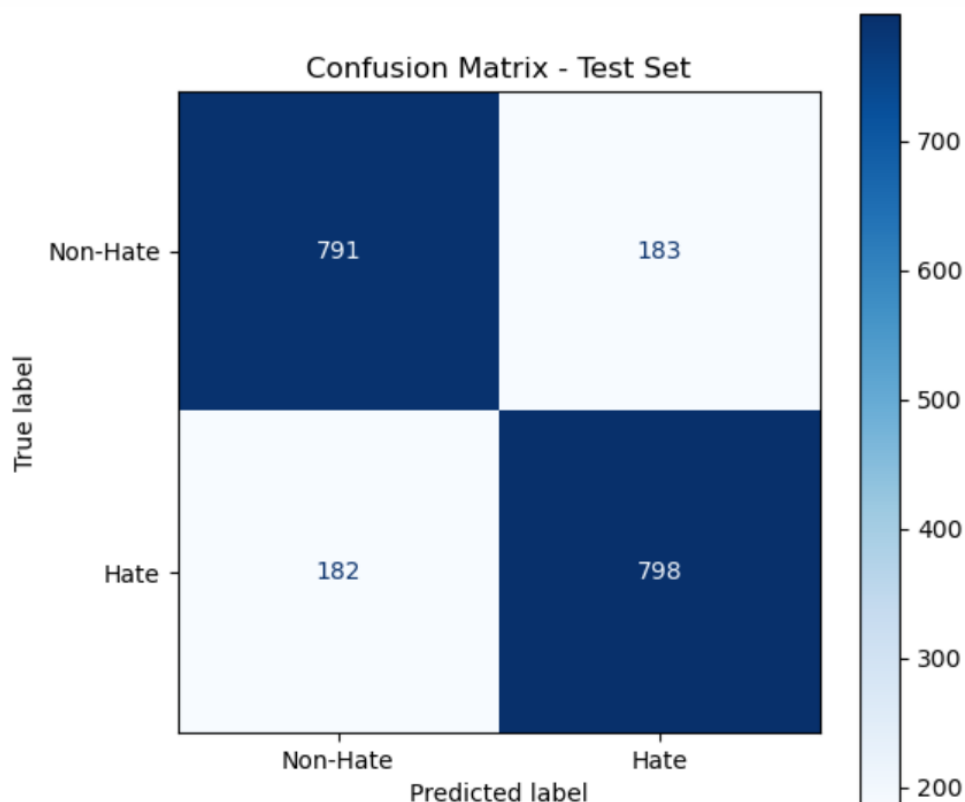


Gambar 4.1. Grafik *loss* dan akurasi *training* dan *validation* model bert+xcgboost

Tabel 4.7. Hasil evaluasi model pada data uji

Class	Precision	Recall	F1-score	Support
Non-hate	0.81	0.81	0.81	974
Hate-speech	0.81	0.81	0.81	980
Accuracy			0.81	1954

Berdasarkan Gambar 4.2 mengenai distribusi kelas dengan *confusin matrix*, terdapat 980 data untuk kelas “*Hate*” dan 974 data untuk kelas “*NonHate*”, menunjukkan distribusi kelas yang seimbang. Model berhasil melakukan prediksi yang benar terhadap 1.589 dari total 1.954 data. Analisis terhadap *confusion matrix* menunjukkan bahwa model mengklasifikasikan 791 data kelas “*NonHate*” dengan benar, namun keliru mengklasifikasikan 183 data kelas tersebut sebagai “*Hate*”. Sebaliknya, dari kelas “*Hate*”, sebanyak 798 data berhasil diklasifikasikan dengan benar, sedangkan 182 sisanya salah diklasifikasikan sebagai “*NonHate*”.



Gambar 4.2. *Confusion matrix* model bert+*xgboost*

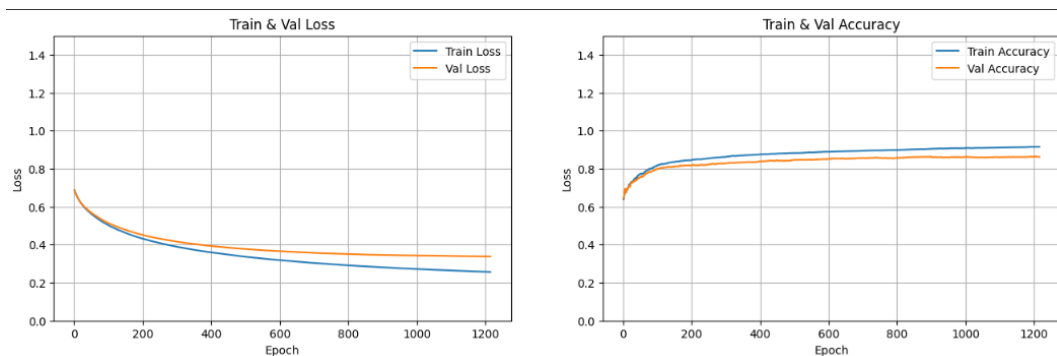
Secara keseluruhan, model menunjukkan performa yang konsisten dalam membedakan antara ujaran kebencian (*hate speech*) dan non-kebencian. Hal ini

tercermin dari distribusi prediksi yang seimbang serta nilai-nilai metrik evaluasi yang seragam antar kelas. Meskipun masih terdapat sejumlah kesalahan klasifikasi, khususnya dalam membedakan kasus-kasus ambiguitas antar kelas, hasil yang diperoleh mengindikasikan bahwa pendekatan yang digunakan mampu menangkap pola-pola relevan dalam data dengan tingkat akurasi yang baik.

B Hasil Evaluasi XGBoost TF-IDF

Pada Gambar 4.3 disajikan hasil evaluasi model yang diterapkan data validasi, diperoleh tingkat akurasi sebesar 85,57%, yang memberitahukan hasil prediksi yang benar terhadap seluruh data yang melebihi 85% yang dimana lebih baik dari model sebelumnya. Dalam kesalahan prediksi, nilai *loss* dalam pelatihan tercatat pada angka 0.2628, sedangkan pada *validation loss* menyentuh angka sebesar 0,3333 mengindikasikan bahwa model belajar dan memvalidasi dengan baik. Adapun matrices performa lainnya dari model XGBoost TF-IDF menghasilkan *metrics* seperti pada Tabel 4.8.

Berdasarkan hasil tabel tersebut, model menunjukkan performa yang baik dalam membedakan antara ujaran kebencian (*hate speech*) dan non-kebencian. Untuk kelas *hate speech*, model mencatat nilai presisi sebesar 88%, yang berarti model memprediksi 88% benar pada kategori *hate-speech*. Nilai *recall* untuk kelas ini mencapai 83%, menunjukkan bahwa dari seluruh data aktual yang termasuk dalam kelas *hate speech*, 83% berhasil diidentifikasi secara tepat. *F1-score* sebesar 0,85 mencerminkan keseimbangan yang baik antara presisi dan *recall* dalam mengenali kelas tersebut. Sementara itu, untuk kelas *nonhate*, model meraih presisi sebesar 83% dan *recall* sebesar 89%, dengan *F1-score* sebesar 0,86. Hal ini mengindikasikan bahwa model memiliki sedikit keunggulan dalam mengenali data *nonhate* dibandingkan *hate speech*. Secara keseluruhan, akurasi model mencapai 86% pada data uji, yang menunjukkan bahwa sebagian besar data berhasil diklasifikasikan dengan benar. Selain itu, nilai AUC-ROC sebesar 0,9347 menunjukkan bahwa model memiliki kemampuan diskriminatif yang sangat baik dalam membedakan antara dua kelas, mendekati nilai maksimum 1.



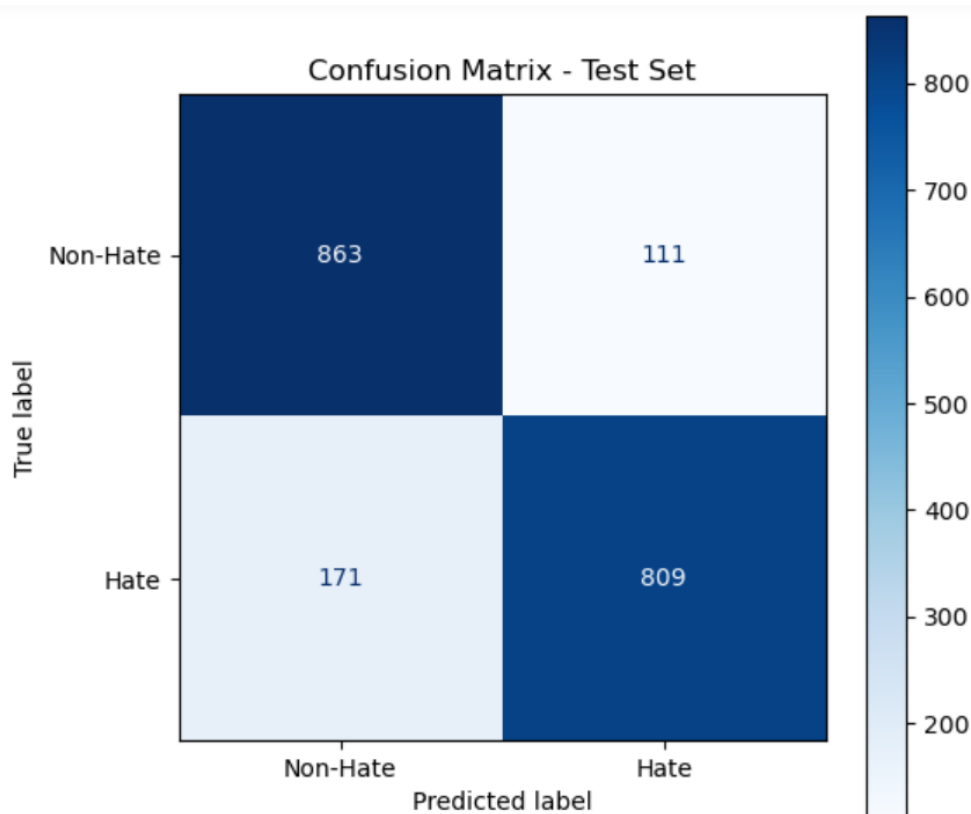
Gambar 4.3. Grafik *loss* dan akurasi *training* dan *validation* model xgboost tf-idf

Tabel 4.8. Hasil evaluasi model pada data uji

Class	Precision	Recall	F1-score	Support
Non-hate	0.83	0.89	0.86	974
Hate-speech	0.88	0.83	0.85	980
Accuracy			0.86	1954

Berdasarkan hasil evaluasi terhadap data uji yang terdiri dari 1.954 sampel, model XGBoost berhasil membuat 1.672 prediksi yang benar, yang menghasilkan tingkat akurasi sebesar 85,57%. Analisis lebih lanjut ditunjukkan melalui *confusion matrix*, di mana sebanyak 863 sampel dari kelas "NonHate" berhasil diklasifikasikan dengan benar, sementara 111 sampel lainnya keliru diklasifikasikan sebagai "Hate". Untuk kelas "Hate", sebanyak 809 sampel berhasil diidentifikasi secara tepat, sedangkan 171 sampel salah diklasifikasikan sebagai "NonHate".

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 4.4. *Confusion matrix* model xgboost tf-idf

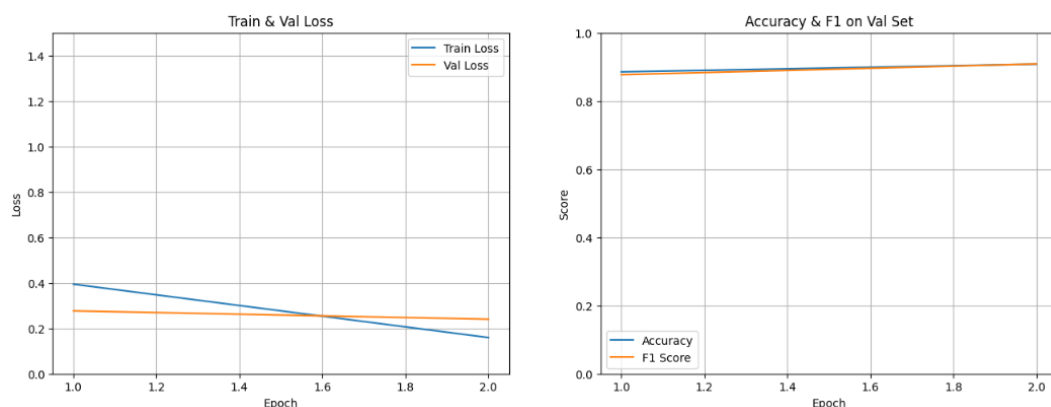
Secara keseluruhan, hasil evaluasi menunjukkan bahwa model memiliki kemampuan klasifikasi yang seimbang, dengan kinerja yang baik dalam mengenali kedua kelas, baik *hate speech* maupun *nonhate*. Jumlah prediksi yang benar yang tinggi serta distribusi kesalahan yang tidak terlalu dominan pada salah satu kelas mengindikasikan bahwa model mampu membedakan ujaran kebencian dan non-kebencian secara efektif. Kinerja ini memperkuat bahwa pendekatan XGBoost dengan representasi fitur TF-IDF memberikan performa yang andal dalam tugas klasifikasi biner. Selain itu, performa model ini menunjukkan sedikit peningkatan dibandingkan pendekatan *hybrid* sebelumnya yang menggabungkan BERT dengan XGBoost.

C Hasil Evaluasi BERT Finetuned

Gambar 4.5 menyajikan hasil evaluasi model BERT *Finetuned* selama proses pelatihan terhadap data validasi. Model menunjukkan stabilitas akurasi pada kisaran 88–89% selama dua *epoch* terakhir, yang mengindikasikan bahwa model memiliki kemampuan prediksi yang sangat baik, dengan tingkat keberhasilan

klasifikasi mendekati 90%. Selisih antara *training loss* (0,1678) dan *validation loss* (0,2806) relatif kecil, yang menunjukkan bahwa model mampu mempelajari pola data secara efektif baik pada data pelatihan maupun data validasi. Hasil ini menunjukkan bahwa model BERT *Finetuned* merupakan pendekatan terbaik dibandingkan tiga model lainnya yang telah dievaluasi sebelumnya.

Berdasarkan Tabel 4.9, performa model dalam membedakan ujaran kebencian (*hate speech*) dan non-kebencian tercatat sangat baik. Untuk kelas "Hate", model mencatat nilai presisi sebesar 90% dan *recall* sebesar 88%, menghasilkan *F1-score* sebesar 0,89. Nilai ini mencerminkan keseimbangan yang solid antara presisi dan *recall*, yang mengindikasikan bahwa model tidak hanya akurat dalam memprediksi *hate-speech*, tetapi juga sensitif terhadap keberadaannya dalam data. Sementara itu, untuk kelas "NonHate", model meraih presisi sebesar 89% dan *recall* sebesar 90%, dengan *F1-score* yang juga mencapai 0,89. Nilai metrik yang seimbang ini menunjukkan bahwa model bekerja konsisten dalam mengenali kedua kategori kelas. Secara keseluruhan model mencapai akurasi 89% untuk pengklasifikasian dengan benar. Selain itu, terdapat faktor penilaian AUC-ROC sebesar 0.9584 menunjukan model sangat percaya diri dalam membedakan antara dua kelas tersebut.



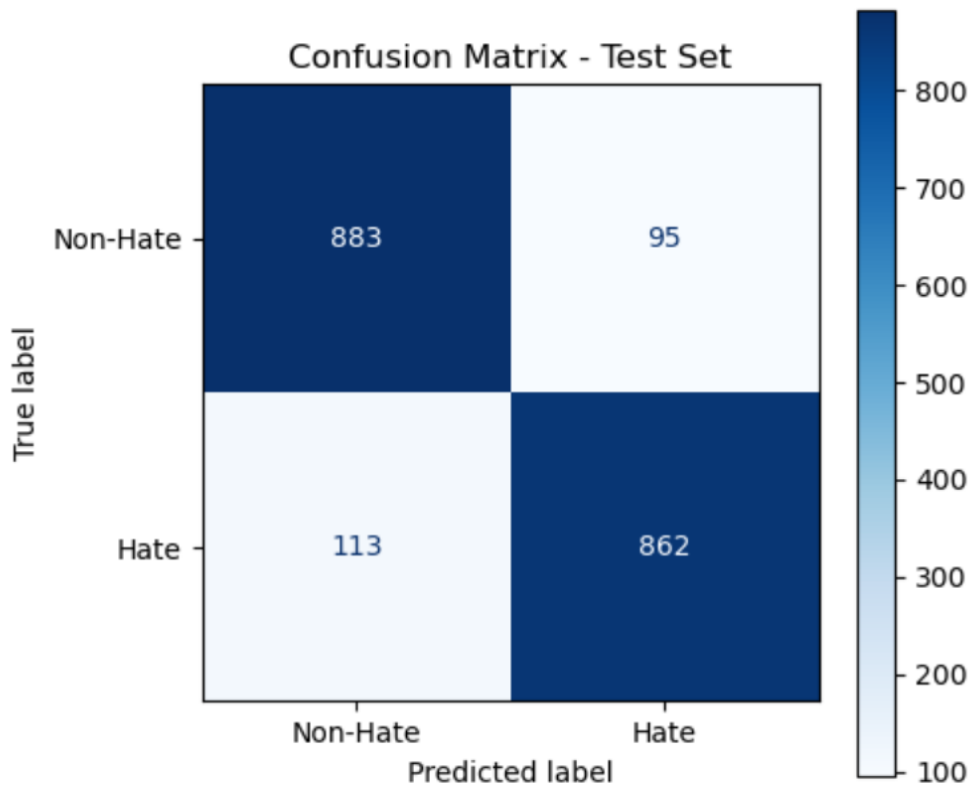
Gambar 4.5. Grafik *loss* dan akurasi *training* dan *validation* model bert finetuned

Evaluasi terhadap data uji yang terdiri dari 1.953 sampel memperkuat hasil tersebut. Model BERT *Finetuned* berhasil mengklasifikasikan sebanyak 1.737 sampel dengan benar, menghasilkan akurasi sebesar 88,99%. Dari *confusion matrix* pada Gambar 4.6, diketahui bahwa sebanyak 883 sampel kelas "NonHate" diklasifikasikan dengan tepat, sementara 95 sampel salah diklasifikasikan sebagai "Hate". Pada kelas "Hate", terdapat 862 prediksi benar dan 113 prediksi yang keliru diklasifikasikan sebagai "NonHate". Keseimbangan kesalahan ini memperlihatkan

bahwa model tidak bias terhadap salah satu kelas, dan mampu menangani distribusi kelas yang seimbang dengan baik.

Tabel 4.9. Hasil evaluasi model pada data uji

Class	Precision	Recall	F1-score	Support
Non-hate	0.89	0.90	0.89	978
Hate-speech	0.90	0.88	0.89	976
Accuracy			0.89	1954



Gambar 4.6. Confusion matrix model bert finetuned

Secara keseluruhan, model BERT *Finetuned* menunjukkan performa klasifikasi terbaik dibandingkan seluruh model yang telah diuji dalam mengenali dua kategori, yaitu *Hate speech* dan *Non Hate*. Model ini unggul dari segi akurasi, keseimbangan metrik antar kelas, serta kemampuan generalisasi yang baik tanpa menunjukkan bias terhadap salah satu kelas. Dengan demikian, model BERT *Finetuned* dapat dianggap sebagai pendekatan paling andal dan efektif dalam tugas klasifikasi ujaran kebencian pada penelitian ini.

4.4.1 Rangkuman dan Evaluasi Akhir Antar Model

Pada bagian ini disajikan rangkuman hasil evaluasi akhir dari tiga pendekatan model yang digunakan dalam tugas klasifikasi *hate speech*, yaitu kombinasi BERT + XGBoost, XGBoost dengan fitur TF-IDF, dan BERT *finetuned*. Evaluasi dilakukan berdasarkan metrik performa klasifikasi yang mencakup akurasi, presisi, *recall*, dan *F1-score*. Masing-masing metrik dihitung untuk dua kategori kelas, yaitu *Hate-Speech* dan *NonHate*, serta dihitung nilai rata-rata untuk memberikan gambaran umum kinerja tiap model. Tabel 4.10 merangkum hasil evaluasi tersebut.

Tabel 4.10. Evaluasi model untuk klasifikasi *hate-speech* dan *non-hate*

Skenario	Sentimen	Akurasi	Metrics (%)		
			Precision	Recall	F1-Score
BERT + XGBoost	Hate-Speech	81%	81%	81%	81%
	Non-Hate		81%	81%	91%
	Avg.		81%	81%	81%
XGBoost TF-IDF	Hate-Speech	85%	83%	89%	86%
	Non-Hate		88%	83%	85%
	Avg.		86%	86%	86%
BERT Finetuned	Hate-Speech	90%	89%	90%	89%
	Non-Hate		90%	89%	89%
	Avg.		90%	90%	89%

Berdasarkan Tabel 4.10, model BERT *Finetuned* menunjukkan kinerja terbaik secara keseluruhan dengan akurasi sebesar 90% dan skor evaluasi rata-rata yang seimbang pada semua metrik (*precision*, *recall*, dan *F1-score* sebesar 89–90%). Model XGBoost TF-IDF menempati posisi kedua setelah BERT *finetuned* dengan akurasi 85% dan skor metrik yang cukup kompetitif, khususnya pada kelas *Hate Speech* yang menunjukkan *recall* sebesar 89%. Sementara itu, model BERT + XGBoost menunjukkan performa paling rendah di antara ketiganya, dengan akurasi hanya 81% dan skor metrik yang relatif seragam namun lebih rendah.

Hasil ini menunjukkan bahwa penggabungan representasi vektor *embeddings* dari BERT dengan model pembelajaran klasik seperti XGBoost tidak secara otomatis menghasilkan peningkatan performa. Salah satu penyebabnya

adalah kurang optimalnya proses integrasi representasi vektor kontekstual dengan arsitektur model pembelajaran klasik yang tidak dirancang untuk memanfaatkan konteks semantik mendalam sebagaimana halnya jaringan neural transformer. Selain itu, pendekatan ini juga tidak melibatkan pelatihan ulang parameter BERT, sehingga potensi penuh dari representasi bahasa yang kontekstual tidak dapat dimanfaatkan secara maksimal.



BAB 5

SIMPULAN DAN SARAN

Bab kesimpulan dan saran berisikan kesimpulan mengenai penelitian yang dilakukan mengenai implementasi model *hybrid* BERT + XGBoost dalam klasifikasi ujaran kebencian (*hate-speech*) tweet pada platform X. Selain itu, disajikan saran-saran untuk pengembangan dan implementasi lanjutan berdasarkan hasil analisis penelitian.

5.1 Simpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

1. Model *hybrid* BERT + XGBoost berhasil diimplementasikan dalam mengklasifikasi konten tweet ujaran kebencian pada platform X dengan tingkat *accuracy* dan *F1-score* yang memadai sebesar 81%. Meskipun tergolong cukup baik, performa model ini tidak melampaui model BERT *finetuned* secara langsung maupun model XGBoost klasik berbasis TF-IDF. Hal ini menunjukkan bahwa integrasi model *hybrid* tidak menjamin peningkatan performa, khususnya dalam konteks klasifikasi konten ujaran kebencian bahasa Indonesia.
2. Pengujian dilakukan terhadap tiga model algoritma klasifikasi, yaitu: (1) model *hybrid* BERT + XGBoost, (2) XGBoost dengan fitur TF-IDF, dan (3) BERT *finetuned*. Seluruh model diuji pada dataset yang sama dan melalui proses *preprocessing* yang seragam. Hasil evaluasi menunjukkan bahwa model BERT *finetuned* memberikan kinerja terbaik secara keseluruhan dibandingkan dua model lainnya.
3. Model BERT *finetuned* mencapai *accuracy* sebesar 88,99%, *precision* 0,8926, *recall* 0,8862, dan *F1-score* 0,8893 pada data uji yang terdiri dari 1.954 sampel, dengan 1.737 sampel diklasifikasikan dengan benar. Keunggulan ini mencerminkan kemampuan model dalam memahami konteks linguistik dan nuansa semantik dalam ujaran kebencian, menjadikannya solusi yang efektif untuk klasifikasi berbasis makna.

4. Meskipun model yang dikembangkan menunjukkan performa cukup baik, tantangan tetap ada dalam menangani kompleksitas bahasa alami, seperti sarkasme, idiom, serta konteks sosial dan budaya yang khas dalam bahasa Indonesia. Hal ini menandakan perlunya pengembangan lanjutan untuk meningkatkan sensitivitas model terhadap variasi linguistik.

5.2 Saran

Berdasarkan hasil penelitian dan simpulan yang telah dipaparkan, terdapat beberapa saran yang dapat diberikan untuk pengembangan lebih lanjut maupun implementasi lanjutan :

1. Penyeretaan Label “Normal” Secara Eksplisit. Menyertakan label khusus untuk kategori “normal” agar model dapat secara eksplisit membedakan antara ujaran kebencian dan komentar netral. Hal ini akan memperjelas pemetaan kelas saat pelabelan maupun saat evaluasi, serta membantu dalam interpretasi hasil klasifikasi multi-label.
2. Integrasi Pendekatan Rule-Based sebagai Sistem Pendukung. Pendekatan *rule-based* dapat diintegrasikan sebagai sistem pelengkap dalam proses inferensi, terutama untuk menangani pola ujaran kebencian yang eksplisit dan berulang. Meskipun model telah dilatih menggunakan data berlabel, penerapan aturan berbasis kata atau frasa kunci dapat menjadi filter awal atau lapisan tambahan yang memperkuat deteksi, khususnya pada kasus-kasus yang belum terwakili di data latih.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

DAFTAR PUSTAKA

- [1] S. Hinduja and J. W. Patchin, *Bullying Beyond the Schoolyard: Preventing and Responding to Cyberbullying*, 2nd ed. Corwin Press, 2014.
- [2] C.-H. Lee and M. Sanchez, "Cyberbullying: Prevalence, causes, and consequences," *International Journal of Cyber Criminology*, vol. 12, no. 1, pp. 78–95, 2018.
- [3] D. Jurafsky and J. H. Martin, "Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition," in *Pearson Education*, 2009.
- [4] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing Management*, vol. 24, no. 5, pp. 513–523, 1988.
- [5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [6] I. T. Jolliffe, "Principal component analysis," *Springer Series in Statistics*, 2002.
- [7] A. Ray and S. Kumar, "Cyberbullying detection on twitter using machine learning," *Journal of Computational Social Science*, 2024.
- [8] E. Hendrayani and A. Pratama, "Digital hate speech propagation on social media," *Social Media Studies*, 2024.
- [9] I. Alfina *et al.*, "Indotoxic2024: Dataset for hate speech in indonesian," *arXiv preprint arXiv:2406.19349*, 2024.
- [10] S. Paul and S. Saha, "Cyberbert: Bert for cyberbullying identification," *Multimedia Systems*, vol. 28, no. 6, pp. 1897–1904, 2022.
- [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers," *arXiv preprint arXiv:1810.04805*, 2019.
- [12] M. Mozafari, R. Farahbakhsh, and N. Crespi, "Hate speech detection and racial bias mitigation," *PLOS ONE*, vol. 15, no. 8, p. e0237861, 2020.
- [13] A. Alamsyah, A. Wibowo, and A. Suryani, "Hoax news detection analysis using indobert deep learning methodology," in *2022 4th International Conference on Cybernetics and Intelligent System (ICORIS)*. IEEE, 2022, pp. 1–6.


- [14] L. R. Hazim and O. Ata, "Textual authenticity in the ai era: Evaluating bert and roberta with logistic regression and neural networks for text classification," in *2024 International Symposium on Electronics and Telecommunications (ISETC)*, 2024, pp. 1–6.
- [15] D. O. Otieno, A. Siami Namin, and K. S. Jones, "The application of the bert transformer model for phishing email classification," in *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2023, pp. 1303–1310.
- [16] J. Yadav, D. Kumar, and D. Chauhan, "Cyberbullying detection using pre-trained bert model," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2020, pp. 1096–1100.
- [17] F. S. Amalia and Y. Suyanto, "Offensive language and hate speech detection using bert model," *Indonesian Journal of Computing and Cybernetics Systems*, vol. 15, no. 2, pp. 129–136, 2021.
- [18] M. Babaeianjelodar, G. P. Prudhvi, S. Lorenz, K. Chen, S. Mondal, S. Dey, and N. Kumar, "Explainable and high-performance hate and offensive speech detection," *arXiv preprint arXiv:2206.12983*, 2022.
- [19] S. Liang, "Comparative analysis of svm, xgboost and neural network on hate speech classification," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 5, pp. 3506–3512, 2021.
- [20] Twitter Inc., "Twitter usage statistics," 2023.
- [21] T. R. Edison, "Social media trends 2023: A global perspective," *Journal of Digital Communication*, vol. 12, no. 2, pp. 45–60, 2023.
- [22] G. Ray, C. D. McDermott, and M. Nicho, "Cyberbullying on social media: Definitions, prevalence, and impact challenges," *Journal of Cybersecurity*, vol. 10, no. 1, p. tyae026, 2024.
- [23] A. Matamoros-Fernández and J. Farkas, "Racism, hate speech, and social media: A systematic review and critique," *Television New Media*, vol. 22, no. 4, pp. 406–430, 2021.
- [24] T. Davidson, D. Warmesley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 11, no. 1, pp. 512–515, 2017.
- [25] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the First Instructional Conference on Machine Learning*, 2003.
- [26] J. Brownlee, "The transformer model," *Machine Learning Mastery*, n.d.


- [27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *Advances in Neural Information Processing Systems*, vol. 26, pp. 3111–3119, 2013.
- [28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [29] J. Hassannataj Joloudari, S. Hussain, M. A. Nematollahi, R. Bagheri, F. Fazl, R. Alizadehsani, R. Lashgari, and A. Talukder, "Bert-deep cnn: state of the art for sentiment analysis of covid-19 tweets," *Social Network Analysis and Mining*, vol. 13, p. 14, 07 2023.
- [30] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "Indolem and indobert: A benchmark dataset and pre-trained language model for indonesian nlp," *arXiv preprint arXiv:2011.00677*, 2020.
- [31] S. Cahyawijaya, G. I. Winata, B. Wilie, K. Vincentio, X. Li, A. Kuncoro, S. Ruder, Z. Y. Lim, H. Lovenia, and P. Fung, "Indonlg: Benchmark and resources for evaluating indonesian natural language generation," *arXiv preprint arXiv:2104.08200*, 2021.
- [32] A. Wibowo, "Pembangunan model bahasa indobert untuk pemrosesan bahasa alami berbahasa indonesia," Master's thesis, Universitas Gadjah Mada, 2020.
- [33] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. CRC Press, 2012.
- [34] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [35] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.
- [36] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.
- [37] L. Mason, J. Baxter, P. L. Bartlett, and M. R. Frean, "Boosting algorithms as gradient descent," in *Advances in Neural Information Processing Systems*, vol. 12, 1999, pp. 512–518.
- [38] B. Clark and F. Lee, "What is gradient boosting?" IBM Think, Apr. 2025.
- [39] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.
- [40] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in Neurorobotics*, vol. 7, p. 21, 2013.

- [41] Y. Wang, Z. Pan, J. Zheng, L. Qian, and L. Mingtao, "A hybrid ensemble method for pulsar candidate classification," *Astrophysics and Space Science*, vol. 364, 08 2019.
- [42] H. Zhang, F. Cheng, and T. Hu, "Performance analysis of xgboost and other machine learning algorithms for credit risk evaluation," *Journal of Machine Learning Research*, vol. 19, no. 1, pp. 1234–1256, 2018.
- [43] D. M. W. Powers, "Evaluation: From precision, recall and f-measure to roc, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.
- [44] M. O. Ibrohim and I. Budi, "Multi-label hate speech and abusive language detection in indonesian twitter," in *Proceedings of the Third Workshop on Abusive Language Online (ALW3)*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 46–57.
- [45] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.
- [46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, pp. 2825–2830, 2011.






Lampiran 1. Hasil Persentase Turnitin

 Page 1 of 69 - Cover Page Submission ID trn:old::1:3302334737



UMN libtii TI 5


Arrafi Aji Pamungkas - Skripsi

 ARRAFI AJI PAMUNGKAS
 2025 GENAP - SKRIPSI INFORMATIKA
 Universitas Multimedia Nusantara


Document Details

Submission ID	trn:old::1:3302334737
Submission Date	Jul 25, 2025, 7:27 PM GMT+7
Download Date	Jul 25, 2025, 7:33 PM GMT+7
File Name	Arrafi_Aji_Pamungkas_SKRIPSI.pdf
File Size	1.2 MB

57 Pages
13,342 Words
82,307 Characters



U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

 Page 1 of 69 - Cover Page Submission ID trn:old::1:3302334737




19% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text




Top Sources

- 13%  Internet sources
- 11%  Publications
- 12%  Submitted works (Student Papers)

UMN

UNIVERSITAS
MULTIMEDIA
NUSANTARA

Top Sources

13%  Internet sources
11%  Publications
12%  Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	kc.umn.ac.id	<1%
2	Student papers	University of New South Wales	<1%
3	Internet	jurnal.stkipgritlungagung.ac.id	<1%
4	Student papers	Coventry University	<1%
5	Internet	docplayer.info	<1%
6	Student papers	Universitas Muhammadiyah Purwokerto	<1%
7	Student papers	UIN Sultan Syarif Kasim Riau	<1%
8	Student papers	Politeknik Manufaktur Negeri Bangka Belitung	<1%
9	Publication	Aghi Kalam Ibrahim, Metty Mustikasari, Irwan Bastian. "ANALISIS SENTIMEN PAD..."	<1%
10	Student papers	Monash University	<1%
11	Internet	repository.uin-suska.ac.id	<1%

12	Student papers	Sriwijaya University	<1%
13	Internet	jurnal.itg.ac.id	<1%
14	Internet	id.123dok.com	<1%
15	Internet	xgboosting.com	<1%
16	Student papers	Queen's University of Belfast	<1%
17	Publication	Karimah Mutisari Hana, Adiwijaya, Said Al Faraby, Arif Bramantoro. "Multi-label ...	<1%
18	Student papers	Universitas Maritim Raja Ali Haji	<1%
19	Student papers	Konsorsium Perguruan Tinggi Swasta Indonesia II	<1%
20	Student papers	Universitas Jember	<1%
21	Publication	Windy Livia Azzahra, Evangs Malloa. "Analisis Sentimen terhadap RSUD Salatiga ...	<1%
22	Student papers	Aalto Yliopisto	<1%
23	Publication	Adib Ulinuha El Majid, Reflan Nuari. "Performance Comparison Of BERT Metrics a...	<1%
24	Student papers	Imperial College of Science, Technology and Medicine	<1%
25	Internet	123dok.com	<1%

26	Publication	Bowden Jenkins, Nicole. "Topic-Based Classification of MAUDE Adverse Problem R..."	<1%
27	Student papers	Brunel University	<1%
28	Publication	Natasa Kleanthous, Abir Hussain. "Machine Learning in Farm Animal Behavior usi..."	<1%
29	Student papers	Tarumanagara University	<1%
30	Internet	dspace.uii.ac.id	<1%
31	Internet	arxiv.org	<1%
32	Internet	essay.utwente.nl	<1%
33	Internet	medium.com	<1%
34	Student papers	unimal	<1%
35	Student papers	Fakultas Teknik	<1%
36	Internet	dspace.cuni.cz	<1%
37	Publication	Fajar Budi Kurniawan, Lia Farokhah. "Aplikasi Cerdas Prediksi Kelulusan Mahasis..."	<1%
38	Student papers	LTI	<1%
39	Student papers	National University of Singapore	<1%

40	Student papers	Universidad Carlos III de Madrid - EUR	<1%
41	Internet	repository.ub.ac.id	<1%
42	Student papers	Associatie K.U.Leuven	<1%
43	Student papers	ICTS	<1%
44	Publication	Mohamed Zul Fadhl Khairuddin, Suresh Sankaranarayanan, Khairunnisa Hasikin,...	<1%
45	Student papers	UPN Veteran Yogyakarta	<1%
46	Student papers	Universitas Bhayangkara Jakarta Raya	<1%
47	Internet	adsii.or.id	<1%
48	Student papers	Deakin University	<1%
49	Publication	Ibnu Zahy' Atha Illah, Wahyu Syaifullah Jauharis Sapu, Aviolla Terza Damaliana. "I...	<1%
50	Internet	journal.mediapublikasi.id	<1%
51	Student papers	Consortio CIXUG	<1%
52	Student papers	Gisma University of Applied Sciences GmbH	<1%
53	Publication	Iqbal Hanan Junaidi, Sopingi, Sri Sumarlinda. "Sistem Retrieval E-Arsip Tirta Asat...	<1%

54	Student papers	University of Lancaster	<1%
55	Internet	blog.enterprisedna.co	<1%
56	Internet	repo.darmajaya.ac.id	<1%
57	Internet	repository.ar-raniry.ac.id	<1%
58	Internet	www.soloaja.com	<1%
59	Publication	Berliana Wahyu Nurlita, Sri Winarno, Adhitya Nugraha, Almas Najib Imam Mutt...	<1%
60	Publication	Febby Apri Wenando, Nooraini Yusoff, Nurul Izrin, Sulistiawati R. N. Ahmad, M. Sa...	<1%
61	Student papers	Institut Bisnis dan Teknologi Indonesia (INSTIKI)	<1%
62	Publication	Kartikadyota Kusumaningtyas, Irmma Dwijayanti, Alfirna Rizqi Lahitani, Muham...	<1%
63	Student papers	Universitas Negeri Semarang - ITh	<1%
64	Student papers	University College London	<1%
65	Student papers	poltekssn	<1%
66	Student papers	yfcnu	<1%
67	Student papers	Kaplan College	<1%

68	Publication	Mitra Unik, M. Eddo Aldo Fahrurrozi, Harmaini Harmaini, Chandra Kusuma. "Impl...	<1%
69	Publication	Muhamad Ziaul Haq, Cut Susan Octiva, Ayuliana Ayuliana, Uli Wildan Nuryanto, D...	<1%
70	Publication	Putu Jeevallucas Jnanamaitriya Surya Gautama, Argo Wibowo, Jong Jek Siang. "An...	<1%
71	Student papers	Universitas Budi Luhur	<1%
72	Student papers	University of Leeds	<1%
73	Student papers	University of Ulster	<1%
74	Student papers	Fachhochschule Salzburg GmbH	<1%
75	Student papers	MIT Academy of Engineering	<1%
76	Student papers	Prague College	<1%
77	Publication	Repan, Barry Ceasar Octariadi, Sucipto. "Penerapan Algoritma (Naive Bayes) Unt...	<1%
78	Student papers	Universitas Muslim Indonesia	<1%
79	Student papers	Universitas Pendidikan Indonesia	<1%
80	Internet	academy.hsoub.com	<1%
81	Internet	nustat.github.io	<1%

82	Publication	Bagus Tri Yulianto Darmawan, Bassamtiano Renaufalgi Irnawan, Yoshimi Suzuki. ...	<1%
83	Publication	Ferris Tita Sabillillah, Sri Winarno, Ryandhika Bintang Abiyi. "Implementasi BERT ...	<1%
84	Publication	Retno Waluyo, Andhar Siraj Munir. "Optimasi Prediksi Kematian pada Gagal Jantu...	<1%
85	Student papers	University of Sunderland	<1%
86	Internet	python.hotexamples.com	<1%
87	Internet	repository.unpar.ac.id	<1%
88	Internet	repository.upbatam.ac.id	<1%
89	Internet	tunasbangsa.ac.id	<1%
90	Publication	Alma Bryan Fitri Finika, Septi Andryana, Ratih Titi Komalasari. "Algoritma Fisher-Y...	<1%
91	Internet	course-net.com	<1%
92	Student papers	AUT University	<1%
93	Student papers	KTO Karatay Aniversitesi	<1%
94	Publication	Nailong Zhang. "A Tour of Data Science - Learn R and Python in Parallel", CRC Pre...	<1%
95	Publication	Taslim Taslim, Susi Handayani, Fajrizal Fajrizal. "Kinerja Komparatif Optimasi Alg...	<1%

96	Student papers	Telkom University	<1%
97	Student papers	Universidad Europea de Madrid	<1%
98	Internet	ejournal.itn.ac.id	<1%
99	Internet	ejournal1.unud.ac.id	<1%
100	Internet	isl.anthropomatik.kit.edu	<1%
101	Internet	maldiniyogi.blogspot.com	<1%
102	Internet	www.cl.uni-heidelberg.de	<1%
103	Internet	www.mdpi.com	<1%
104	Internet	www.slideshare.net	<1%
105	Publication	Aini Suri Talita, Aristiawan Wiguna. "Implementasi Algoritma Long Short-Term M...	<1%
106	Publication	Aryo Sasi Kirono, Yessica Nataliani. "Perbandingan Algoritma Machine Learning d...	<1%
107	Publication	Lely Kurniawati, Dadang Priyanto, Neny Sulistia Ningsih, Moch Syahrir, Ria Risma...	<1%
108	Internet	repository.dinamika.ac.id	<1%
109	Internet	www.jstage.jst.go.jp	<1%

Lampiran 2. Formulir Bimbingan

Form Bimbingan Skripsi Program Studi Informatics Semester Genap 2024/2025



Nama : ARRAFI AJI PAMUNGKAS
NIM : 00000064717
Angkatan : 2021
Dosen Pembimbing : Aditiyawan, S.Komp., M.Si. (Pembimbing)

No	Tanggal	Jam	Keterangan	Tanggal Approval
1	10 Februari 2025	15:59	Bimbingan pembahasan pertama mengenai topik dan judul penelitian serta penggunaan model algoritma	12 Juni 2025 9:48
2	20 Februari 2025	15:15	Bimbingan mengenai Bab 1 mengenai key point pengambilan latar belakang	12 Juni 2025 9:48
3	28 Februari 2025	15:14	Bimbingan mengenai Bab 2 mengenai landasan teori serta konsultasi penambahan metodologi	12 Juni 2025 9:48
4	07 Maret 2025	14:15	Pengajuan Bab 3 mengenai metodologi penelitian yang digunakan serta bimbingan revisi bab sebelumnya	12 Juni 2025 9:48
5	29 Mei 2025	15:00	Bimbingan mengenai hasil dan evaluasi penelitian serta revisi mengenai Bab 4	12 Juni 2025 9:48
6	06 Juni 2025	15:00	Bimbingan mengenai Bab 5	12 Juni 2025 9:48
7	11 Juni 2025	13:00	Pergantian judul dan review laporan final skripsi	12 Juni 2025 9:48
8	20 Maret 2025	15:00	Bimbingan mengenai program dan model serta revisi pada bab sebelumnya	12 Juni 2025 9:48
9	24 Maret 2025	13:00	Konsultasi usulan dengan pendekatan yang berbeda mengenai judul dan algoritma serta bimbingan Bab 4	12 Juni 2025 9:48