Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.6, December 2025, pp. 3270~3282
e-ISSN: 2548-6861

Evaluating the Impact of Random Over Sampling on IndoBERT Performance for Indonesian Sentiment Analysis

Dimas Ramadhan Alfinsyah [1]*, Bambang Pilu Hartato [2]*
* Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Amikom Yogyakarta
dimasramadhan29@students.amikom.ac.id [1], bambang.pilu@amikom.ac.id [2]

Article Info

Article history:

Received 2025-10-13
Revised 2025-11-03
Accepted 2025-11-08

Keyword:

Sentiment Analysis,
IndoBERT,
Random Over Sampler,
Imbalanced Data,
Model Evaluation.

ABSTRACT

Sentiment analysis is a prominent research area in natural language processing (NLP). For the Indonesian language, IndoBERT has emerged as a leading model due to its competitive performance. However, its effectiveness is strongly influenced by balanced class distribution. A common challenge arises because user reviews on digital platforms, such as the Google Play Store, often exhibit imbalanced classes. This study investigates the effectiveness of the Random Over Sampler (ROS) technique in improving IndoBERT's performance under imbalanced data conditions. The dataset consists of 13,821 user reviews of the IDN App collected from the Google Play Store between 2015 and July 2025. Prior to modeling, data preprocessing was performed, including punctuation removal, case folding, stopword removal, tokenizing, normalization, and stemming to ensure textual consistency. Reviews were categorized into two sentiment classes: positive (3–5 stars) and negative (1–2 stars). Two experimental scenarios were conducted: (1) IndoBERT without ROS and (2) IndoBERT with a balanced dataset using ROS. Model performance was evaluated using accuracy, precision, recall, and F1-score, with data split into 70% training, 20% validation, and 10% testing. Results showed a significant improvement after ROS implementation: 94.55% accuracy, 94.61% precision, 94.53% recall, and 94.54% F1-score. Confusion matrix analysis indicated improved classification of the minority class, reducing the error rate by 49%. However, learning curve analysis revealed potential overfitting due to ROS. Further research is needed to optimize ROS strategies for better performance and generalization.

I. PENDAHULUAN

Era transformasi digital telah mengubah cara konsumen berinteraksi dengan produk dan layanan, khususnya melalui platform aplikasi *mobile* yang menjadi bagian integral kehidupan sehari-hari. Dalam konteks ini, analisis sentimen memainkan peran krusial sebagai instrumen untuk memahami persepsi dan opini pengguna terhadap aplikasi digital [1]. Kemampuan untuk mengekstrak dan menginterpretasi sentimen dari ulasan pengguna tidak hanya memberikan wawasan berharga bagi pengembang aplikasi dalam meningkatkan kualitas produk, tetapi juga memungkinkan identifikasi aspek-aspek spesifik yang mempengaruhi kepuasan pengguna [2]. Lebih lanjut, analisis sentimen telah terbukti efektif dalam berbagai domain aplikasi, mulai dari platform game hingga aplikasi streaming musik, yang menunjukkan relevansi dan adaptabilitasnya dalam konteks digital modern [3][4]. Keberhasilan implementasi analisis sentimen pada berbagai platform digital menunjukkan potensi besar teknik ini dalam memberikan wawasan yang dapat ditindaklanjuti bagi pengembangan aplikasi *mobile* [5].

Meskipun analisis sentimen telah menunjukkan potensi yang signifikan, implementasinya pada data ulasan aplikasi digital menghadapi tantangan fundamental berupa ketidakseimbangan distribusi kelas. Permasalahan *imbalance* data secara konsisten ditemukan dalam berbagai studi analisis sentimen, di mana distribusi data yang tidak merata dapat

mengurangi kemampuan model untuk mengenali kelas minoritas secara akurat [6].

Sebagai gambaran, kami telah melakukan observasi secara acak terhadap 15 aplikasi yang terdapat pada Play Store. Hasil observasi kami sajikan pada Tabel 1. Perlu diketahui bahwa data yang disajikan pada Tabel 1 merupakan data yang diambil pada tanggal 8 September 202

TABEL I
DISTRIBUSI ULASAN 15 APLIKASI PADA PLAY STORE

No	Nama Aplikasi	Rating Bintang					Persen	
		1	2	3	4	5	Positif	Negatif
1	Roblox	188.564	43.133	70.754	148.079	1.277.470	86,59%	13,41%
2	Block Blast!	7.828	1.742	2.829	6.679	84.706	90,78%	9,22%
3	Super Bear Adventure	12.510	3.171	5.253	10.791	95.990	87,72%	12,28%
4	Mobile Legends: Bang Bang	251.229	28.932	31.061	55.328	434.450	65,02%	34,98%
5	Subway Surfers	51.330	16.768	42.424	68.524	473.454	89,56%	10,44%
6	Free Fire	509.661	98.606	109.697	140.685	1.548.851	74,73%	25,27%
7	Stickman Party	30.980	7.421	12.063	22.980	250.155	88,13%	11,87%
8	Ludo King	59.646	11.562	26.290	38.834	241.285	81,14%	18,86%
9	Pizza Ready!	2.501	829	1.144	1.565	6.019	72,38%	27,62%
10	School Party Craft	6.173	2.331	5.369	16.958	146.535	95,21%	4,79%
11	EA SPORTS FC	241.187	45.786	65.819	87.123	480.050	68,81%	31,19%
12	Worms Zone .io	66.897	22.758	50.582	86.576	572.835	88,79%	11,21%
13	SAKURA School Simulator	34.654	9.887	17.739	37.847	214.873	85,86%	14,14%
14	**Magic Chess**	**21.124**	**2.913**	**3.083**	**3.983**	**37.503**	**64,96%**	**35,04%**
15	8 Ball Pool	140.126	30.977	63.218	105.628	638.682	82,52%	17,48%

Untuk memudahkan proses analisis, kami mengubah sistem penilaian berbasis *rating* (bintang) menjadi penilaian berbasis sentimen. *Rating* 1 dan 2 kami kategorikan sebagai Sentimen Negatif, sedangkan *rating* 3 sampai dengan 5 kami kelompokkan sebagai Sentimen Positif. Sebagai contoh, *Magic Chess* mendapat 21.124 *rating* 1 dan 2.913 *rating* 2. Sehingga secara total *Magic Chess* mendapat 24.037 sentimen negatif. Sedangkan sentimen positif berjumlah 44.569 yang merupakan total penilaian untuk *rating* 3-5 untuk *Magic Chess*. Jika dibuat prosentase maka *Magic Chess* mendapat 35,04% sentimen negatif dan 64,96% sentimen positif.

Dari data yang ditampilkan pada Tabel 1, terlihat bahwa keseluruhan aplikasi memiliki ulasan positif dan negatif yang tidak berimbang. Sehingga hal tersebut semakin menguatkan argumen kami bahwa merupakan hal yang sangat sulit atau bahkan tidak mungkin untuk menemukan *dataset* nyata yang memiliki rasio yang benar-benar seimbang untuk setiap kelasnya. Jika *dataset* tersebut tidak diperlakukan khusus terlebih dahulu sebelum digunakan tentu saja akan mempengaruhi model yang dilatih. Model bisa saja mengalami bias terhadap kelas mayoritas. Hal tersebut selaras dengan apa yang dinyatakan pada penelitian [7] terutama dalam konteks *multiclass*.

Kondisi tersebut menjadi semakin kompleks pada aplikasi berbahasa Indonesia, di mana model seperti IndoBERT memerlukan distribusi data yang relatif seimbang untuk mencapai performa optimal dalam pemahaman konteks linguistik yang spesifik [8]. Dampak dari ketidakseimbangan ini dapat mengakibatkan model cenderung memprediksi kelas mayoritas dan mengabaikan kelas minoritas, sehingga mengurangi reliabilitas hasil analisis sentimen secara keseluruhan [9].

Penelitian oleh Mahmoudi & Salem [10] juga menegaskan bahwa meskipun arsitektur BERT unggul dalam tugas klasifikasi teks, performanya dapat menurun signifikan ketika data pelatihan tidak seimbang. Distribusi kelas yang condong membuat model lebih banyak belajar dari kelas mayoritas dan gagal mengenali pola yang merepresentasikan kelas minoritas. Secara teknis, IndoBERT seperti model Transformer pada umumnya sangat sensitif terhadap distribusi data yang tidak seimbang karena mekanisme optimasi yang berbasis *loss function*. Selama proses *fine-tuning*, model menggunakan *Cross-Entropy Loss* yang menghitung rata-rata kerugian di seluruh sampel. Ketika dataset timpang, strategi paling efisien untuk meminimalkan kerugian rata-rata adalah dengan memprioritaskan prediksi pada kelas mayoritas. Akibatnya, sinyal *loss* dari kelas minoritas menjadi relatif kecil dan sering diabaikan, menyebabkan model cenderung bias terhadap kelas mayoritas [10].

Berbagai penelitian telah mengusulkan sejumlah pendekatan untuk menangani ketidakseimbangan kelas. Pada tingkat *data-level*, pendekatan *under-sampling* dilakukan dengan mengurangi jumlah sampel dari kelas mayoritas agar distribusi data menjadi seimbang. Meskipun sederhana, metode ini berisiko menghilangkan informasi penting karena sebagian data dihapus [11]. Sebaliknya, metode *synthetic oversampling* menambah jumlah data pada kelas minoritas melalui duplikasi atau sintesis sampel baru. Teknik populer

seperti SMOTE, Borderline-SMOTE, dan ADASYN terbukti efektif dalam meningkatkan akurasi dan F1-score pada dataset yang tidak seimbang [12][13]. Temuan tersebut sejalan dengan hasil penelitian ini, di mana penerapan Random Over Sampling (ROS) pada IndoBERT berhasil meningkatkan performa model dari 89% menjadi 95% dengan F1-score yang lebih seimbang antar kelas. Selain metode berbasis data, pendekatan *cost-sensitive learning* dan *class-weighting* juga umum digunakan untuk menyesuaikan bobot kerugian pada kelas minoritas [14][15]. Pendekatan ini efektif diterapkan pada model berbasis *Transformer* seperti IndoBERT karena kompatibel dengan fungsi *cross-entropy loss* yang digunakan selama proses pelatihan.

Meskipun ada penelitian yang telah mengeksplorasi penerapan IndoBERT untuk analisis sentimen teks berbahasa Indonesia, masih terdapat perbedaan mendasar dalam konteks dan fokus penelitian yang dilakukan. Penelitian pembanding oleh Fathin dkk. [16] berfokus pada deteksi hoaks menggunakan data berita politik formal dari portal berita terverifikasi. Sebaliknya, penelitian ini menitikberatkan pada analisis sentimen dari ulasan pengguna di Google Play Store yang bersifat lebih informal dan penuh bahasa percakapan. Karakteristik data tersebut menimbulkan tantangan tersendiri karena mengandung banyak istilah gaul dan variasi penulisan. Oleh sebab itu, tahap *preprocessing* dalam penelitian ini lebih komprehensif, termasuk normalisasi menggunakan kamus *colloquial* dan proses *stemming* yang tidak diperlukan pada penelitian Fathin dkk. [16]. Selain itu, fokus analisis antara kedua penelitian juga berbeda. Fathin et al. menyoroti perbandingan performa antara ROS dan RUS, sedangkan penelitian ini berkontribusi pada pemahaman yang lebih mendalam mengenai dampak ROS itu sendiri, termasuk potensi *overfitting* akibat duplikasi data minoritas yang dianalisis melalui kurva pembelajaran (*learning curves*).

Secara khusus, teknik *Random Over Sampling* (ROS) telah menunjukkan efektivitas yang menjanjikan dalam mengatasi permasalahan ketidakseimbangan data pada berbagai domain penelitian. Penelitian [17] melaporkan bahwa ROS mampu meningkatkan akurasi model dari 78% menjadi 85%, sementara *Random Under Sampling* (RUS) justru menurunkan kinerja menjadi 74% dalam konteks data observasional kesehatan. Temuan serupa dikonfirmasi oleh Wongvorachan dkk. [11] yang menunjukkan bahwa *oversampling* dan SMOTE dapat meningkatkan *accuracy* hingga 12%, jauh melampaui *undersampling* yang hanya memberikan peningkatan 5% dalam domain *data mining* pendidikan. Penelitian Wibowo & Fatichah [18] pada *dataset* dengan rasio ketidakseimbangan 1:10 menunjukkan bahwa *oversampling* mampu meningkatkan F1-score dari 62% menjadi 79%, meski dengan risiko *overfitting* yang perlu dikelola. Studi Sir & Soepranoto [19] mengonfirmasi efektivitas *resampling* sederhana dengan peningkatan *accuracy* dari 83% menjadi 91%, sementara Bej dkk. [20] melaporkan bahwa *multi-schematic oversampling* dapat meningkatkan *recall* kelas minoritas hingga 18% dibandingkan metode konvensional. Meskipun hasil-hasil penelitian tersebut menunjukkan potensi positif ROS, sebagian besar masih terfokus pada domain kesehatan, pendidikan, atau data numerik, sehingga masih terdapat *gap* penelitian yang signifikan dalam penerapan ROS untuk analisis sentimen berbahasa Indonesia berbasis IndoBERT pada ulasan aplikasi digital.

Dengan meningkatkan representasi kelas minoritas, Random Over Sampling (ROS) memberikan kesempatan lebih besar bagi model Transformer untuk mempelajari pola unik dari kelas tersebut selama proses pelatihan [21]. Berbeda dengan metode oversampling sintetik seperti SMOTE atau ADASYN, ROS tidak menciptakan data baru, melainkan menduplikasi data asli sehingga terhindar dari potensi noise atau bias yang sering muncul pada pembuatan data sintetik [22]. Meskipun demikian, teknik ini memiliki risiko overfitting karena pengulangan sampel identik dapat membuat model terlalu menghafal data latih dan mengurangi kemampuan generalisasi terhadap data baru [21]. Untuk memitigasi risiko tersebut, beberapa strategi regulatif dapat diterapkan selama pelatihan IndoBERT. Salah satunya adalah penambahan *dropout layer* pada lapisan klasifikasi yang terbukti efektif dalam mengurangi *co-adaptation* antar neuron [23]. Selain itu, peningkatan nilai *weight decay* pada optimizer AdamW dapat membantu mengontrol pembaruan bobot agar efek regularisasi tetap terjaga [24]. Kombinasi keduanya berperan penting dalam menjaga stabilitas pelatihan dan meningkatkan kemampuan generalisasi model.

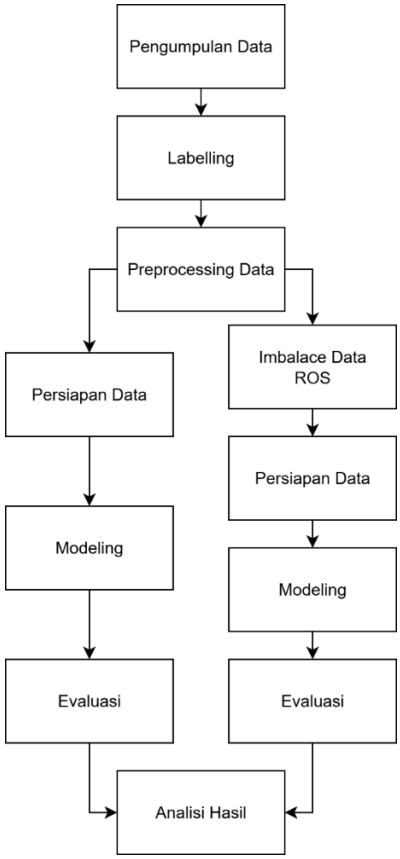
Penelitian ini bertujuan untuk menguji efektivitas ROS sebagai teknik *resampling* dalam meningkatkan performa IndoBERT pada analisis sentimen. Untuk mencapai tujuan tersebut kami menggunakan ulasan aplikasi IDN App sebagai bahan penelitian kami. Pemilihan IDN App didasarkan pada posisinya sebagai platform berita digital populer di Indonesia dengan basis pengguna yang luas [25]. Ulasan yang kami gunakan didapat dari proses *scraping* ulasan pengguna Google Play Store untuk periode 2015 - Juli 2025 untuk aplikasi IDN App. *Dataset* ulasan ini memiliki distribusi sentimen yang cenderung tidak seimbang. *Dataset* didominasi oleh ulasan positif [18]. Hal tersebut memberikan kondisi ideal untuk menguji efektivitas teknik ROS dalam mengatasi permasalahan *imbalance class***.**

Penelitian ini secara khusus mengevaluasi kemampuan ROS dalam mengatasi bias klasifikasi. ROS dievaluasi untuk mengatasi bias yang muncul akibat distribusi kelas yang tidak seimbang. Teknik *oversampling* adaptif sebelumnya terbukti efektif dalam berbagai konteks pembelajaran mesin, khususnya untuk menangani *imbalance class* [26]. Penelitian yang kami lakukan juga menganalisis performa IndoBERT sebelum dan sesudah penerapan ROS. Analisis dilakukan menggunakan beragam metrik evaluasi untuk mengukur efektivitas teknik *resampling*. Pengukuran dilakukan pada konteks analisis sentimen *binary class* berbahasa Indonesia [8]. Penelitian ini juga mengeksplorasi parameter optimal ROS untuk meningkatkan performa IndoBERT. Hasil penelitian diharapkan memberi kontribusi signifikan bagi pengembangan sistem analisis sentimen pada platform digital

di Indonesia. Kontribusi utama penelitian ini adalah menampilkan bukti empiris pengaruh ROS terhadap model IndoBERT dalam melakukan analisis sentimen terhadap ulasan pengguna berbahasa Indonesia pada platform Google Play Store.

II. METODE

Dalam melakukan eksperimen, kami melakukan prosedur seperti yang ditunjukkan oleh Gambar 1. Platform yang kami gunakan adalah Google Colab yang mengutilisasi GPU NVDIA T4. Penambahan GPU tersebut dimaksudkan untuk mempercepat proses *training* dan inferensi model.



Gambar 1. Alur Eksperimen

A. Pengumpulan Data

Pengumpulan data dalam penelitian ini berfokus pada ulasan pengguna aplikasi "IDN App" yang tersedia pada Google Play Store. Data dikumpulkan menggunakan teknik *web scraping* dengan pustaka google-play-scraper yang diintegrasikan dengan Google Colab [27]. Setiap ulasan mencakup informasi seperti ID, nama pengguna, gambar profil, teks ulasan, skor *rating*, versi aplikasi, jumlah *thumbs up*, serta waktu unggahan. Data mentah kemudian diorganisir ke dalam DataFrame Pandas dan disimpan dalam format CSV dengan nama *scrapped_data_idn_app_all.csv.*

Total 13.821 ulasan berhasil dikumpulkan sebagai *dataset* utama analisis sentimen. Tabel 2 menampilkan sebagian data hasil *scraping*. Ulasan negatif memiliki *rating* 1–2, sedangkan positif *rating* 3–5. Hal ini memperlihatkan distribusi persepsi pengguna yang beragam terhadap aplikasi IDN App.

TABEL II
HASIL SCRAPING DATA

userName	rating	at	content
Pengguna Google	1	13/07/ 2025 14:32	saya topup gold di idn via mbanking briva, setelah tagihan+pajak berhasil dibayar kan dan sukses, saya cek history pembayaran di apk brimo dan akun google berhasil tetapi di apk idn malah disuruh bayar lagi... gold tidak masuk... jadi gimana min.? walau jumlah tidak seberapa tapi bisa untuk top gift harian.. saya tunggu itikad baiknya terima kasih.
Pengguna Google	4	12/07/ 2025 14:13	aplikasi sudah bagus, tapi kadang jaringan nya suka jelek padahal udah pakai internet sendiri. kadang juga saat nonton live komentar nya tidak muncul. tolong gimana solusi nya
Pengguna Google	5	27/10/ 2015 12:47	Aplikasi yang luar biasa. Keren! It works 100% totally.

B. Labeling

Labeling atau pelabelan dilakukan berdasarkan skor *rating* pengguna. Skema pelabelan menggunakan pendekatan *binary classification*. Ulasan dengan skor *rating* 1 dan 2 dikategorikan sebagai ulasan 'negatif' dan diberi label 0. Sedangkan ulasan dengan skor *rating* 3, 4, dan 5 dikategorikan sebagai ulasan 'positif' dengan label 1 [28].

Pendekatan ini dipilih untuk memberikan pemisahan yang jelas terhadap orientasi opini yang diberikan oleh para pengguna. *Rating* 1-2 mencerminkan ketidakpuasan pengguna, sedangkan *Rating* 3-5 menunjukkan tingkat kepuasan yang dapat diterima hingga sangat baik. Tabel 3 menunjukkan beberapa *review* yang telah diberikan label.

TABEL III
HASIL LABELING

content	rating	label	category
saya topup gold di idn via mbanking briva, setelah tagihan+pajak berhasil dibayar kan dan sukses, saya cek history	1	0	negatif

pembayaran di apk brimo dan akun google berhasil tetapi di apk idn malah disuruh bayar lagi... gold tidak masuk... jadi gimana min.? walau jumlah tidak seberapa tapi bisa untuk top gift harian.. saya tunggu itikad baiknya terima kasih.			
aplikasi sudah bagus, tapi kadang jaringan nya suka jelek padahal udah pakai internet sendiri. kadang juga saat nonton live komentar nya tidak muncul. tolong gimana solusi nya	3	1	positif
Aplikasi yang luar biasa. Keren! It works 100% totally.	5	1	positif

C. *Preprocessing Data*

Tahap berikutnya adalah *Preprocessing Data* atau Pra Pengolahan Data. Tahapan ini dimaksudkan agar ulasan yang menjadi input siap untuk digunakan pada proses berikutnya. Secara teknis, tahap pra pengolahan yang kami lakukan memiliki sub-tahapan sebagai berikut:

1. *Punctuation Removal dan Case Folding*

Tahap *punctuation removal* melibatkan pembersihan teks dari karakter yang tidak diperlukan. Karakter tersebut meliputi tanda baca berlebihan, karakter khusus, angka, dan *whitespace* yang tidak relevan. Pembersihan ini penting karena karakter-karakter tersebut tidak memberikan makna signifikan pada proses sentimen analisis.

Selanjutnya, proses *case folding* dilakukan untuk mengonversi seluruh teks agar memiliki bentuk *case* yang sama untuk setiap hurufnya. Dalam kasus ini kami memilih menggunakan *lower case form*, di mana semua huruf dikonversi menjadi huruf non-kapital atau huruf kecil. Proses ini menghasilkan representasi data yang konsisten dan seragam. Tanpa *case folding*, model dapat menganggap kata yang sama dengan kapitalisasi berbeda sebagai entitas yang berbeda [1]. Tabel 4 menampilkan contoh bagaimana kami melakukan *punctuation removal* dan *case folding* pada suatu ulasan.

TABEL IV
PUNCTUATION REMOVAL DAN CASE FOLDING

Sebelum *Punctuation Removal* dan *Case Folding*	**Sesudah** *Punctuation Removal* dan *Case Folding*
App nyaa baguss bangett, bisaa buat nonton live**,** drama**,** dan masih banyak lagii**,** sukaa bangett sama app nyaa **11/10** pokoknyaa buat idn inii	**app** nyaa baguss bangett bisaa buat nonton live drama dan masih banyak lagii sukaa bangett sama app nyaa pokoknyaa buat idn inii

2. *Stopword Removal*

Stopword removal dilakukan untuk menghilangkan kata-kata yang tidak relevan dalam proses sentimen analisis [29]. Sama halnya dengan *punctuation removal*, tahap ini dilakukan karena kata-kata yang dihilangkan tersebut tidak memiliki makna signifikan dalam proses analisis sentimen. Sehingga jika kata-kata tersebut tidak dihapus justru akan membebani komputasi yang dilakukan tanpa memberikan *value* yang berarti.

Dalam eksperimen yang kami lakukan, kata hubung, kata depan, dan kata keterangan kami kategorikan sebagai *stopword*. Sehingga kata-kata yang masuk dalam kategori tersebut akan kami hilangkan dan tidak dimasukkan pada proses selanjutnya. Tabel 5 menampilkan contoh bagaimana suatu ulasan dihilangkan *stopwordnya*.

TABEL V
STOPWORD REMOVAL

Sebelum *Stopword Removal*	**Sesudah** *Stopword Removal*
app nyaa baguss bangett bisaa buat nonton live drama **dan masih banyak** lagii sukaa bangett **sama** app nyaa pokoknyaa **buat** idn inii	app nyaa baguss bangett bisaa nonton live drama lagii sukaa bangett app nyaa pokoknyaa idn inii

3. *Tokenizing*

Tokenizing adalah proses memecah suatu ulasan menjadi unit-unit kata. Setiap kata yang dipisahkan oleh spasi dianggap 1 token. Sehingga suatu ulasan akan berubah menjadi list kata/token. Langkah ini penting karena tokenisasi memungkinkan model memahami struktur teks pada level kata [5]. Tokenisasi juga memudahkan analisis lebih lanjut, seperti identifikasi kata kunci, perhitungan frekuensi, atau ekstraksi fitur. Tanpa tokenisasi, teks dianggap satu string panjang yang sulit diproses oleh NLP. Contoh bagaimana suatu ulasan diubah menjadi token-token kata ditunjukkan oleh Tabel 6.

TABEL VI
TOKENIZING

Sebelum *Tokenizing*	***Sesudah** Tokenizing*
app nyaa baguss bangett bisaa nonton live drama lagii sukaa bangett app nyaa pokoknyaa idn inii	**[**'app', 'nyaa', 'baguss', 'bangett', 'bisaa', 'nonton', 'live', 'drama', 'lagii', 'sukaa', 'bangett', 'app', 'nyaa', 'pokoknyaa', 'idn', 'inii'**]**

4. *Normalisasi dengan Kamus Colloquial Indonesian Lexicon*

Tidak semua kata dalam *dataset* merupakan kata baku, sehingga diperlukan proses normalisasi untuk menyamakan bentuk kata yang tidak lazim. Proses ini mengubah kata tidak baku atau bahasa gaul menjadi bentuk yang lebih umum, misalnya *"gk"* menjadi *"tidak"*, *"bgt"* menjadi *"banget"*, dan *"mantul"* menjadi *"mantap betul"*. Normalisasi membantu meningkatkan konsistensi data dan

memungkinkan model mengenali pola sentimen dengan lebih akurat. Langkah ini penting karena ulasan pengguna sering ditulis dengan gaya bahasa sehari-hari.

Kami menggunakan kamus colloquial indonesian lexicon untuk melakukan normalisasi kata. Kamus ini mengonversi kata-kata informal, singkatan, dan bahasa gaul menjadi bentuk formal yang standar [28]. Kamus tersebut berisi pasangan kata tidak baku dan padanan formalnya. Setiap token dibandingkan dengan entri kamus, dan jika cocok diganti dengan bentuk bakunya. Tabel 7 menampilkan contoh list token kata sebelum dan sesudah dilakukan normalisasi terhadap list token tersebut.

TABEL VII
HASIL NORMALISASI

Sebelum Normalisasi	**Sesudah** Normalisasi
['app', '**nyaa**', '**baguss**', '**bangett**', '**bisaa**', 'nonton', 'live', 'drama', '**lagii**', '**sukaa**', '**bangett**', 'app', 'nyaa', 'pokoknyaa', 'idn', '**inii**']	['app', '**nya**', '**bagus**', '**banget**', '**bisa**', 'menonton', 'live', 'drama', '**lagi**', '**suka**', '**banget**', 'app', 'nya', 'pokoknyaa', 'idn', '**ini**']

5. Stemming

Stemming adalah proses yang kami lakukan untuk mengubah kata menjadi bentuk dasarnya. Proses ini kami lakukan dengan mengubah kata ke bentuk dasar dengan menghilangkan imbuhan seperti awalan, akhiran, dan sisipan. Langkah ini penting agar kata-kata dengan bentuk berbeda dengan makna yang sama dapat diseragamkan sehingga data lebih konsisten. Sehingga model analisis sentimen dapat bekerja dengan lebih efektif. Karena *dataset* yang kami gunakan adalah *dataset* berbahasa Indonesia, maka kami menggunakan *Sastrawi Stemmer* sebagai *stemmer* kami [30]. Tabel 8 menunjukkan contoh proses *stemming* yang kami lakukan. Perhatikan kata "menonton". Kata tersebut menjadi kata "tonton". Dengan demikian kata "menonton" akan memiliki konteks yang sama dengan kata "ditonton", "menontoni", dan "menontonkan".

TABEL VIII
HASIL STEMMING

Sebelum Stemming	**Sesudah** Stemming
['app', 'nya', 'bagus', 'banget', 'bisa', '**menonton**', 'live', 'drama', 'lagi', 'suka', 'banget', 'app', 'nya', 'pokoknyaa', 'idn', 'ini']	['app', 'nya', 'bagus', 'banget', 'bisa', '**tonton**', 'live', 'drama', 'lagi', 'suka', 'banget', 'app', 'nya', 'pokoknyaa', 'idn', 'ini']

6. Mengembalikan Token menjadi kalimat utuh

Dikarenakan IndoBERT membutuhkan kalimat utuh daripada list token sebagai inputnya, maka list token diubah kembali menjadi bentuk kalimat utuhnya. Kami melakukannya dengan *pseudocode* berikut:

var Sentence ← ""

for each Token in TokenList do

Sentence ← Sentence + Token + " "

end for.

Tabel 9 menunjukkan hasil dari proses pengembalian token menjadi kalimat utuh. Kolom *Sebelum* menampilkan kumpulan token yang masih terpisah. Sedangkan kolom *Sesudah* menunjukkan token yang telah digabungkan kembali menjadi kalimat utuh. Dengan demikian kalimat tersebut dapat digunakan sebagai *input* oleh *IndoBERT*.

TABEL IX
HASIL MENGEMBALIKAN TOKEN MENJADI KALIMAT UTUH

Sebelum	**Sesudah**
['app', 'nya', 'bagus', 'banget', 'bisa', 'tonton', 'live', 'drama', 'lagi', 'suka', 'banget', 'app', 'nya', 'pokoknyaa', 'idn', 'ini']	**app nya bagus banget bisa tonton live drama lagi suka banget app nya pokoknyaa idn ini**

Implementasi pra pengolahan yang kami lakukan masih mengalami beberapa keterbatasan teknis. Jika diperhatikan, kami melakukan tahap normalisasi (2.3.4) setelah *stopword removal*. Memang secara teori bisa saja hal tersebut mempengaruhi performa model. Namun, dari hasil pengujian yang kami lakukan hal tersebut tidak memberikan dampak sama sekali terhadap model yang kami lakukan.

Selain itu, kamus Colloquial Indonesian Lexicon dan *stemmer* Sastrawi yang kami gunakan belum mampu menangani 100% variasi bahasa nonformal yang ada pada *dataset* kami. Sebagai contoh, kata "nya" yang seharusnya sudah tidak muncul lagi pada *dataset* pada kenyataannya masih tetap muncul.

Walaupun demikian, seharusnya hal tersebut tidak akan mempengaruhi performa IndoBERT secara signifikan. Hal ini didasarkan pada kemampuan arsitektur transformer IndoBERT yang sudah mampu memahami konteks kalimat meskipun terdapat variasi bentuk kata [31]. Kemampuan tersebut diperoleh karena IndoBERT sudah dilatih dengan korpus besar berbahasa Indonesia dengan berbagai gaya penulisan [32]. Dengan demikian, model seharusnya sudah cukup tangguh terhadap *noise* dan variasi bahasa yang tidak sepenuhnya teratasi pada fase *preprocessing*.

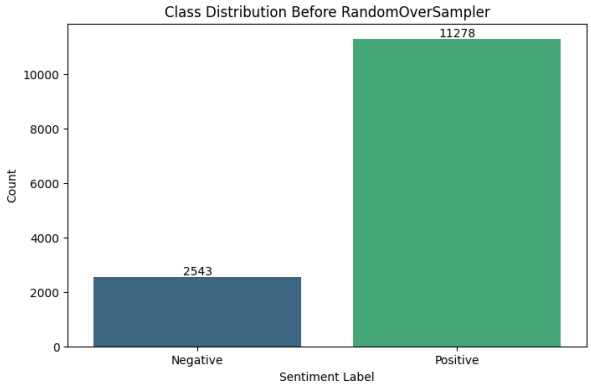
Dalam penelitian ini, alur penelitian hingga tahap *preprocessing* dilakukan dengan prosedur yang sama untuk seluruh data. Setelah tahap *preprocessing*, eksperimen bercabang menjadi dua perlakuan. Perlakuan pertama menggunakan *dataset* asli yang belum dilakukan penyeimbangan kelas sama sekali. Kami menyebutnya kondisi tanpa ROS. Perlakuan ke-dua menggunakan *dataset*

yang telah diseimbangkan dengan teknik *Random Over Sampler* atau kondisi dengan ROS. Secara lebih jelas, alur eksperimen ditunjukkan oleh Gambar 1.

ROS adalah teknik *resampling* untuk mengatasi ketidakseimbangan kelas dalam *dataset*. Secara teknis, metode ini bekerja dengan menduplikasi data pada kelas minoritas sehingga jumlahnya sama dengan kelas mayoritas [33]. Duplikasi dilakukan secara acak hingga terjadi keseimbangan antara kelas minoritas dan mayoritas. Dengan demikian, distribusi data menjadi seimbang tanpa menambah informasi baru.

7. Tanpa Random Over Sampler (ROS)

Gambar 2 menunjukkan distribusi label sebelum penerapan ROS. Terlihat ketidakseimbangan yang signifikan antara kelas positif dan negatif. Dari total 13.821 ulasan, 11.278 termasuk kelas positif dan 2.543 kelas negatif, dengan rasio ketidakseimbangan sekitar 4,4:1, yang menandakan dominasi kuat kelas positif.



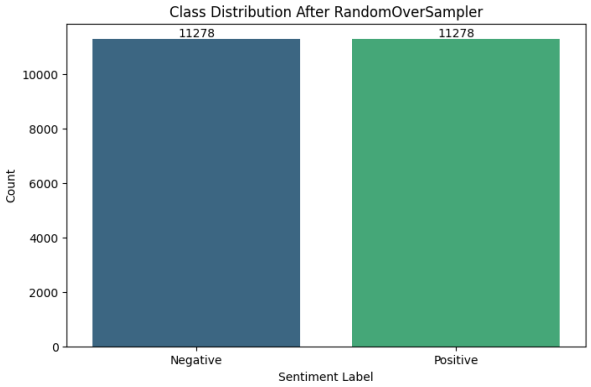
Gambar 2. Hasil Persebaran Label sebelum ROS

8. Dengan Random Over Sampler (ROS)

Gambar 3 memperlihatkan hasil persebaran label setelah penerapan ROS. Jumlah data antara kelas positif dan negatif telah seimbang. Jumlah komentar positif tetap 11.278, sementara komentar negatif diduplikasi hingga menyamai jumlah tersebut. Dengan demikian, distribusi data yang semula timpang berubah menjadi seimbang 1:1. Penerapan ROS dilakukan sebelum proses split data agar keseimbangan kelas tercapai pada keseluruhan dataset.

Secara teknis, implementasi ROS dimulai dengan menyiapkan data ulasan dan label dalam bentuk *array*. Data kemudian diubah ke format dua dimensi agar sesuai dengan kebutuhan *resampling*. Selanjutnya, ROS digunakan untuk menggandakan data pada kelas minoritas hingga jumlahnya setara dengan kelas mayoritas.

Misalnya, jika kelas Positif memiliki dua sampel dan kelas Negatif hanya satu sampel, ROS akan menyalin sampel kelas Negatif sehingga jumlahnya sama dengan kelas Positif. Apabila terdapat beberapa sampel pada kelas minoritas, pemilihan sampel untuk digandakan dilakukan secara acak hingga keseimbangan tercapai. Hasil *resampling* dikonversi kembali ke bentuk data tabular. Data ini disimpan agar konsisten dengan format awal dan siap dipakai untuk pelatihan model berikutnya.



Gambar 3. Hasil Persebaran Label sesudah ROS

D. Persiapan Data untuk Training

Persiapan data untuk training merupakan tahap krusial. IndoBERT menggunakan tokenizer khusus berbahasa Indonesia yang mampu menangani variasi kata dan struktur kalimat lokal. Proses tokenisasi ini mengubah teks ulasan menjadi token, lalu mengkonversinya ke ID numerik dengan penambahan token khusus [CLS] dan [SEP]. Token [CLS] digunakan sebagai penanda awal kalimat dan mewakili keseluruhan urutan untuk tugas klasifikasi. Token [SEP] digunakan sebagai pemisah antar kalimat atau penanda akhir urutan teks. Sebagai contoh, jika teks asli adalah *"mantap"*, proses tokenisasi akan menghasilkan token ['mantap'], yang kemudian dikonversi menjadi token ID [13729]. Urutan terakhir yaitu menjadi [CLS] 13729 [SEP]. Sehingga model dapat memproses konteks penuh dari ulasan tersebut untuk keperluan klasifikasi.

Semua urutan diseragamkan hingga 128 token melalui *padding* atau *truncating* [34]. Padding menambahkan token kosong pada urutan pendek agar panjangnya seragam dengan batas maksimum. *Truncating* digunakan untuk memotong urutan yang melebihi panjang yang telah ditentukan.

Setelah itu, *attention mask* dibuat untuk membedakan token asli dan token hasil *padding* sehingga model hanya memproses informasi yang relevan (token asli). Berikutnya, *dataset* dibagi menjadi *training set* (70%), *validation set* (20%), dan *test set* (10%). Pembagian ini dilakukan untuk menjaga evaluasi tetap objektif. Data selanjutnya dikonversi menjadi tensor PyTorch. Terakhir, data diorganisir dengan *DataLoader* agar pelatihan lebih efisien.

E. Modeling

Tahap *modeling* melibatkan inisialisasi dan konfigurasi model BERT untuk tugas klasifikasi sentimen biner. Kami menggunakan arsitektur *BertForSequenceClassification*

dengan model *pre-trained indolem/indobert-base-uncased* yang disesuaikan untuk dua label *output* (negatif dan positif) [8]. Model kemudian dijalankan di GPU untuk memanfaatkan akselerasi komputasi. Akselerasi ini sangat penting untuk pelatihan model bahasa besar. Sebagai *optimizer*, *AdamW* dipilih dengan *learning rate 2e-5* dan *epsilon 1e-8*. Nilai parameter tersebut direkomendasikan untuk *fine*-tuning BERT. Pelatihan dilakukan selama 10 epoch dengan batch size 16. Hyperparameter yang digunakan dapat dilihat pada Tabel 10. *Learning rate scheduler* diimplementasikan untuk mengelola laju pembelajaran secara dinamis [35]. Hal ini memastikan konvergensi yang optimal dan mencegah *overfitting*.

TABEL X
HYPERPARAMETER

Learning Rate	Epsilon	Epoch	Batch Size
2e-5	1e-8	10	16

Pada tahap ini, proses *modeling* tidak hanya mencakup inisialisasi model. *Modeling* juga melibatkan pelatihan dan validasi. Proses *training* dilakukan dengan menggunakan data latih sedangkan, validasi hasil pelatihan menggunakan data validasi untuk memantau kinerja model dan mencegah terjadinya bias.

F. Evaluasi dan Hasil

Evaluasi model dilakukan secara komprehensif untuk mengukur kinerja klasifikasi sentimen. Selama pelatihan, metrik utama seperti *Loss*, *Accuracy*, *F1-Score*, *Precision*, dan *Recall* dipantau secara berkelanjutan. Pemantauan dilakukan baik pada saat *training* maupun validasi di setiap *epoch*. Pemantauan ini bertujuan melacak perkembangan performa model sekaligus mendeteksi potensi *overfitting*.

Setelah proses pelatihan selesai, model kemudian dievaluasi secara menyeluruh. Evaluasi menggunakan data *test* yang belum pernah digunakan pada proses pelatihan ataupun validasi belumnya. Hasil evaluasi dapat mencerminkan kemampuan generalisasi yang dimiliki model.

Untuk mengukur performa, matrik pertama yang kami gunakan adalah *Confusion Matrix*. Matrik ini menampilkan jumlah prediksi benar dan salah pada setiap kelasnya. Hal tersebut mampu menggambarkan distribusi kesalahan yang dilakukan oleh model. Dari *confusion matrix* dapat dihitung metrik turunan seperti *Accuracy*, *Precision*, *Recall*, dan *F1-score*.

Dikarenakan kasus yang kami gunakan hanya bersifat biner, yaitu positif dan negatif, maka kami menggunakan istilah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) pada *confusion matrix* kami. TP adalah jumlah data positif yang berhasil diprediksi benar sebagai positif oleh model. TN adalah jumlah data negatif yang berhasil diprediksi benar sebagai negatif oleh model. FP adalah jumlah kasus negatif yang diidentifikasi sebagai kasus positif oleh model. Sebaliknya, FN adalah jumlah data positif yang diidentifikasi sebagai kelas negatif oleh model.

Setelah nilai *confusion matrix* diperoleh, kami menggunakannya untuk menghitung matrik lainnya, yaitu *Accuracy* (*acc*), *Precision* (*prec*), *Recall* (*rec*), dan *F1-score* (*F1*). Pemilihan metrik ini didasarkan pada karakteristik *imbalanced dataset*. *Accuracy* memberi gambaran umum tentang persentase prediksi benar, tetapi kurang informatif jika distribusi kelas tidak seimbang [19]. Oleh karena itu, *precision* dan *recall* digunakan untuk mengukur sejauh mana model mengidentifikasi kelas tertentu dengan benar. *Precision* menekankan pada tingkat ketepatan prediksi positif [18]. *Recall* mengukur kemampuan model dalam menangkap semua data positif [17]. *F1-score* menjadi ukuran harmonisasi antara *precision* dan *recall* serta representatif pada kondisi kelas yang tidak seimbang [26].

Persamaan masing-masing metrik dirumuskan sebagai berikut:

a. Accuracy

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Persamaan (1) menunjukkan persamaan yang kami gunakan untuk menghitung *Accuracy*, dengan TP adalah *True Positive*, TN adalah *True Negative*, FP adalah *False Positive*, dan FN adalah *False Negative*.

b. Precision

$$Prec = \frac{TP}{TP + FP} \quad (2)$$

Persamaan (2) menunjukkan persamaan yang kami gunakan untuk menghitung *Precision*, dengan TP adalah *True Positive*, dan FP adalah *False Positive*.

c. Recall

$$Rec = \frac{TP}{TP + FN} \quad (3)$$

Persamaan (3) menunjukkan persamaan yang kami gunakan untuk menghitung *Recall* dengan TP adalah *True Positive*, dan FN adalah *False Negative*.

d. F1-Score

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

Persamaan (4) adalah persamaan yang kami gunakan untuk menghitung F1-Score.

Hasil evaluasi pada *test dataset* disajikan dalam bentuk *classification report*. Laporan tersebut merinci nilai *accuracy, precision, recall,* dan *F1-score* untuk masing-masing kelas sentimen. Selain itu, *confusion matrix* divisualisasikan untuk memberikan gambaran lebih detail terhadap sebaran tebakan yang dilakukan oleh model.

G. Analisis Hasil

Analisis difokuskan untuk membandingkan kinerja IndoBERT tanpa ROS dan dengan ROS. Pada skenario tanpa ROS, model memiliki kecenderungan bias terhadap kelas mayoritas. Bias ini terjadi karena distribusi data yang tidak seimbang. Dengan penerapan ROS, kelas lebih seimbang. Keseimbangan ini mampu meningkatkan performa model secara keseluruhan.

Hasil evaluasi disajikan melalui *classification report*, *confusion matrix*, dan grafik evaluasi untuk memperjelas perbedaan performa antara model dengan dan tanpa ROS. Analisis tidak hanya menyoroti peningkatan performa, tetapi juga mempertimbangkan potensi *overfitting* akibat ROS. Risiko muncul karena duplikasi data minoritas, yang membuat model cenderung "menghafal" pola alih-alih melakukan generalisasi. Dengan demikian, tujuan analisis ini adalah memberikan pemahaman empiris tentang sejauh mana ROS meningkatkan performa IndoBERT. Peningkatan tersebut difokuskan pada penyelesaian masalah *imbalanced dataset* dalam analisis sentimen. Analisis sekaligus mengidentifikasi keterbatasan dari pendekatan ini.

III. HASIL DAN PEMBAHASAN

A. Performa Training dan Validation Model

Evaluasi performa IndoBERT sebelum ROS menunjukkan perbedaan signifikan antara hasil training dan validation. Tabel 11 menunjukkan training loss turun konsisten dari 0,3245 (*epoch* 1) menjadi 0,1048 (*epoch* 10). Penurunan ini diikuti dengan peningkatan *training accuracy* dari 0,8642 menjadi 0,9727. Sebaliknya, *validation loss* meningkat dari 0,2692 menjadi 0,4622. Sementara itu, *validation accuracy* cenderung stagnan dalam rentang 0,88–0,90. *Gap* yang signifikan antara *training* dan *validation performance* ini mengindikasikan terjadinya *overfitting*. Model IndoBERT terlalu fokus mempelajari pola spesifik dari data latih. Model ada indikasi mengalami kegagalan dalam melakukan generalisasi yang optimal terhadap data *validation* dalam konteks analisis sentimen.

TABEL XI
HASIL TRAINING DAN VALIDATION SEBELUM ROS

Epoch	train_loss	train_acc	val_loss	val_acc
1	0.324558	0.864275	0.269232	0.889451
2	0.253122	0.900558	0.33272	0.899205
3	0.217745	0.921956	0.30091	0.90065
4	0.192562	0.936634	0.307342	0.898121
5	0.170786	0.945111	0.3094	0.902697
6	0.155495	0.954104	0.402793	0.899446
7	0.147533	0.957412	0.404897	0.900891
8	0.124377	0.965474	0.434495	0.89764
9	0.110025	0.97116	0.468481	0.901614
10	0.104878	0.97271	0.462239	0.898724

Sementara itu, ketika ROS diterapkan pada *dataset* model IndoBERT dapat memberikan hasil yang lebih baik. Hal tersebut ditunjukkan oleh Tabel 12.

TABEL XII
HASIL TRAINING DAN VALIDATION SESUDAH ROS

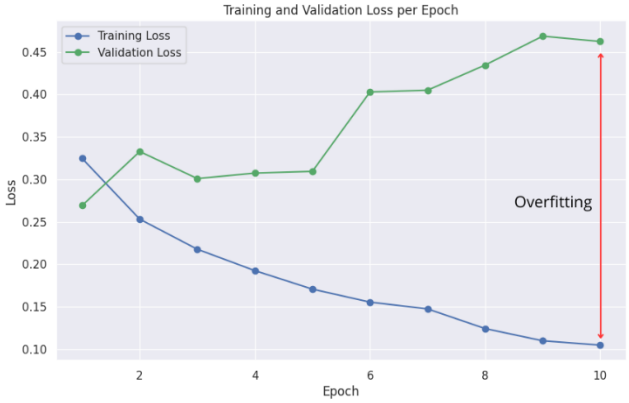
Epoch	train_loss	train_acc	val_loss	val_acc
1	0.366649	0.853379	0.26682	0.912204
2	0.265396	0.912091	0.275522	0.918853
3	0.223228	0.931978	0.195598	0.94013
4	0.185773	0.946672	0.19097	0.946129
5	0.160272	0.956552	0.198339	0.94768
6	0.142931	0.963456	0.272667	0.943026
7	0.120173	0.968902	0.253782	0.946794
8	0.106977	0.972259	0.24622	0.948345
9	0.0974252	0.974919	0.238546	0.952113
10	0.0878241	0.976503	0.255093	0.950561

Penerapan ROS sebagai teknik penyeimbangan data menunjukkan perbaikan substansial pada performa model (Tabel 12). *Training loss* tetap menunjukkan tren penurunan yang stabil dari 0,3666 menjadi 0,0878. Pada *train accuracy* meningkat dari 0,8533 hingga 0,9765. Yang lebih penting, *validation loss* berhasil dikendalikan dalam rentang yang lebih stabil antara 0,19–0,27. Hal ini disertai dengan peningkatan *validation accuracy* yang signifikan dari 0,9122 menjadi 0,9505. Hasil ini menunjukkan ROS dapat menyeimbangkan distribusi kelas, membuat IndoBERT unggul pada data training sekaligus lebih mampu melakukan generalisasi.

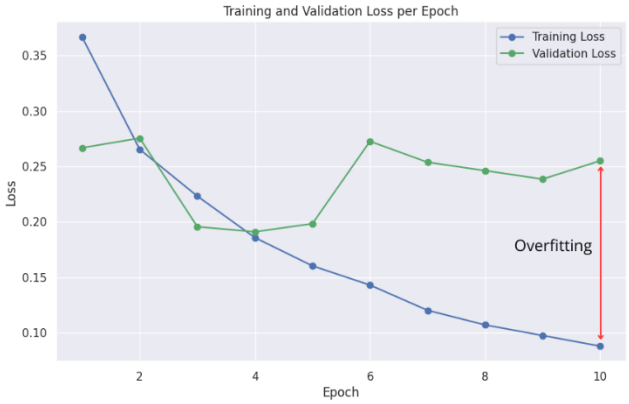
B. Analisis Grafik Loss

Gambar 4 menunjukkan *training loss* menurun secara konsisten dari 0,32 hingga 0,10, sedangkan *validation loss* justru meningkat dari 0,26 menjadi 0,46. Pola ini menandakan adanya *overfitting* karena model terlalu fokus pada data latih dan gagal menjaga kinerja validasi.

Sebaliknya Gambar 5 memperlihatkan pola *loss* lebih stabil. *Training loss* turun dari 0,36 menjadi 0,08, sedangkan *validation loss* berada pada fluktuasi yang terkendali. Hal ini menunjukkan bahwa ROS berhasil mengurangi kesenjangan antara *training* dan *validation*. Selain itu model dapat meningkatkan kemampuan generalisasi. Hal ini berarti ada peningkatan efektivitas kurang lebih sebesar 50%.



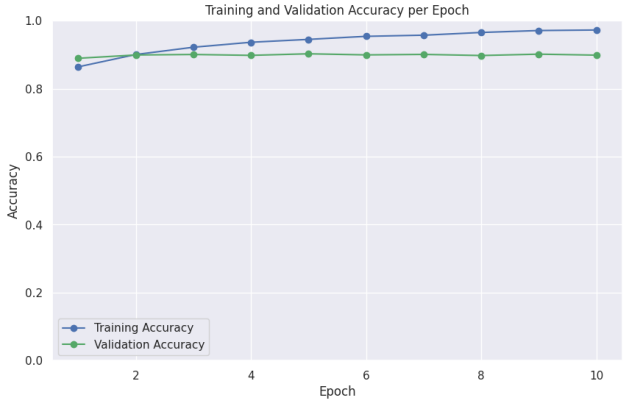
Gambar 4. Grafik Loss sebelum ROS



Gambar 5. Grafik Loss sesudah ROS

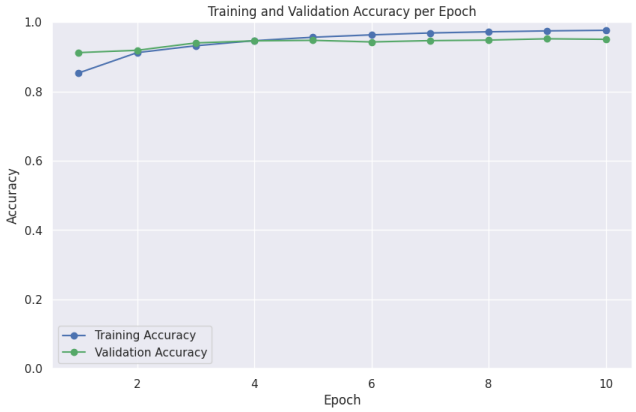
C. *Analisis Grafik Accuracy*

Gambar 6 menunjukkan *training accuracy* meningkat dari 0,86 menjadi 0,97 pada *epoch* ke-10. Namun, *validation accuracy* stagnan di kisaran 0,88–0,90 sejak *epoch* awal. Kondisi ini menegaskan adanya *overfitting*, karena performa latih naik, tetapi validasi tidak membaik.



Gambar 6. Grafik akurasi sebelum ROS

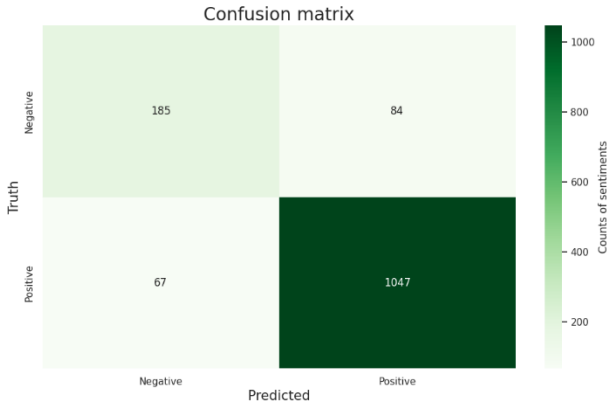
Pada Gambar 7 *training accuracy* meningkat stabil dari 0,85 hingga 0,97. Pada saat bersama, *validation accuracy* naik dari 0,91 menjadi 0,95. Model menunjukkan progres pembelajaran yang lebih baik. Gap akurasi berkurang dari 0.08 menjadi 0,04, atau meningkat 50% dibandingkan kondisi sebelumnya. Akurasi optimal tercapai lebih cepat, yakni pada *epoch* ke-8 dibanding *epoch* ke-10 sebelum ROS.



Gambar 7. Grafik akurasi sesudah ROS

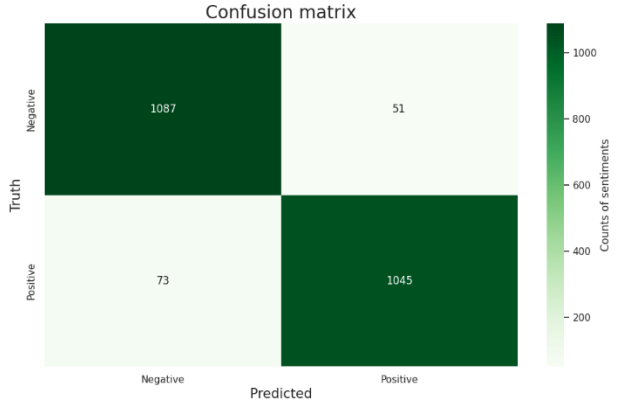
D. *Analisis Hasil Confusion Matrix*

Analisis *confusion matrix* pada Gambar 8 dan Gambar 9 menunjukkan perbedaan jelas sebelum dan sesudah penerapan ROS. Sebelum ROS, model cenderung bias terhadap kelas positif. Dari 269 data negatif, hanya 185 terklasifikasi benar dan 84 salah sebagai positif. Sementara dari 1.114 data positif, 1.047 terklasifikasi benar dan 67 salah sebagai negatif.



Gambar 8. Confusion Matrix sebelum ROS

Setelah ROS diterapkan, performa model meningkat terutama pada kelas negatif. Dari 1.138 data negatif, sebanyak 1.087 terklasifikasi dengan benar sehingga kesalahan prediksi jauh berkurang. Pada kelas positif, dari 1.118 data, sebanyak 1.045 berhasil diklasifikasikan dengan benar, meskipun masih ada sebagian kecil yang salah. Secara keseluruhan, jumlah misklasifikasi menurun dari 151 kasus (10,9%) menjadi 124 kasus (5,5%), yang berarti penurunan error rate sebesar 49,7%. Hasil ini menegaskan bahwa ROS tidak hanya mampu menyeimbangkan distribusi error antar kelas, tetapi juga secara signifikan meningkatkan akurasi klasifikasi.



Gambar 9. Confusion Matrix sesudah ROS

E. Hasil Performa IndoBERT

	precision	recall	f1-score	support
Negative	0.7419	0.6840	0.7118	269
Positive	0.9251	0.9425	0.9337	1114
accuracy			0.8923	1383
macro avg	0.8335	0.8133	0.8228	1383
weighted avg	0.8895	0.8923	0.8906	1383

Gambar 10. Hasil Performa sebelum ROS

Gambar 10 dan Gambar 11 menampilkan perbandingan performa model IndoBERT sebelum dan sesudah penerapan ROS. Sebelum ROS, model menunjukkan bias terhadap sentimen positif dengan *precision* 0,93 dan *recall* 0,94, sedangkan sentimen negatif hanya mencapai *precision* 0,73 dan *recall* 0,69. Ketidakseimbangan ini menghasilkan *F1-score* yang timpang antara kelas positif (0,93) dan negatif (0,71), dengan akurasi keseluruhan 0,89.

	precision	recall	f1-score	support
Negative	0.9312	0.9631	0.9469	1138
Positive	0.9611	0.9275	0.9440	1118
accuracy			0.9455	2256
macro avg	0.9461	0.9453	0.9454	2256
weighted avg	0.9460	0.9455	0.9455	2256

Gambar 11. Hasil Performa sesudah ROS

Implementasi ROS menghasilkan perbaikan yang signifikan dalam keseimbangan klasifikasi. Gambar 11 menunjukkan bahwa precision untuk sentimen negatif meningkat dari 0,73 menjadi 0,93 (peningkatan sekitar 28,77%). Sementara itu, recall meningkat dari 0,68 menjadi 0,96. Peningkatan ini sekitar 39,13%. Untuk sentimen positif, precision meningkat dari 0,92 menjadi 0,96 (peningkatan sekitar 2,15%). Recall sedikit menurun dari 0,94 menjadi 0,92. Penurunan ini sekitar 1,06%. Hal ini mengindikasikan *trade-off* yang optimal untuk mencapai keseimbangan klasifikasi. Perbandingan performa model antara sebelum dan sesudah diimplementasikannya ROS ditampilkan pada Tabel 15.

TABEL XIII
PERBANDINGAN SEBELUM DAN SESUDAH ROS

	Accuracy	Precision	Recall	F1-Score
Sebelum ROS	89%	83%	81%	82%
Sesudah ROS	94%	94%	94%	94%

F1-Score menunjukkan konvergensi yang ideal antara kedua kelas sentimen. Sebelum ROS, terdapat *gap* F1-Score sebesar 0,22 antara sentimen positif dan negatif. Sedangkan setelah ROS, kedua kelas mencapai F1-Score yang identik yaitu 0,94. Accuracy keseluruhan meningkat dari 0,89 menjadi 0,94. Hal ini menunjukkan peningkatan performa sekitar 6,74%. *Macro average* dan *weighted average* yang mencapai 0,94 mengkonfirmasi bahwa peningkatan performa tersebar merata pada kedua kelas dan tidak bias terhadap kelas mayoritas.

F. Pengujian Model dengan Dataset Baru

TABEL XIV
TABEL UJI DENGAN DATASET BARU

No	Ulasan	Groun dtruth	Model tanpa ROS	Model dengan ROS
1	kenapa baru install udah gak bisa login ngeblank	Negatif	Positif	Negatif
2	akun aku sering ke logout sendiri mana berkali kali lagi, tolong diperbaiki dong minn	Negatif	Positif	Negatif
…				
125	Tolong diperbaiki kembali bug bug yang ada di aplikasi ya, aplikasinya sering ngelag	Negatif	Positif	Negatif

Misklasifikasi pada *dataset* baru terjadi ketika hasil prediksi sentimen dari model sebelum penerapan ROS dan sesudah ROS tidak sesuai dengan label asli (*groundtruth*) yang ditentukan berdasarkan skor rating pengguna. Sebagai contoh, ulasan dari pengguna dengan isi "kenapa baru install udah gak bisa login ngeblank" memiliki skor 2 yang menunjukkan sentimen negatif. Namun, model sebelum ROS mengklasifikasikan ulasan tersebut sebagai positif, sedangkan model setelah ROS berhasil mengenalinya dengan benar sebagai negatif. Pola serupa juga ditemukan pada sejumlah ulasan lain, di mana penerapan ROS membantu model memperbaiki kesalahan prediksi terutama pada kelas minoritas. Beberapa hasil perbandingan prediksi antara kedua model ditampilkan pada Tabel 16.

IV. KESIMPULAN

Penelitian ini membuktikan efektivitas *Random Over Sampler* (ROS) dalam mengatasi ketidakseimbangan data dan meningkatkan performa model secara signifikan. Akurasi meningkat dari 89% menjadi 94%, dengan *F1-score* mencapai 0,94 sebagai hasil konvergensi optimal antar kelas sentimen. Peningkatan terbesar terjadi pada klasifikasi sentimen negatif, di mana *precision* naik dari 73% menjadi 94%, dan *recall* dari 69% menjadi 96%. Hasil ini menunjukkan bahwa ROS efektif dalam mengurangi bias model terhadap kelas mayoritas serta menyeimbangkan kinerja klasifikasi. Nilai *macro average* dan *weighted average* keduanya mencapai 95%, menegaskan bahwa teknik *oversampling* sederhana dapat meningkatkan performa model *transformer* pada data teks berbahasa Indonesia yang tidak seimbang.

Meskipun demikian, hasil juga mengindikasikan adanya risiko *overfitting* akibat duplikasi data kelas minoritas, yang dapat menimbulkan redundansi informasi dan menurunkan kemampuan generalisasi model. Oleh karena itu, penelitian lanjutan disarankan untuk mengeksplorasi strategi penyeimbangan yang lebih optimal, seperti kombinasi ROS dengan regularisasi, penggunaan metode lain seperti SMOTE atau ADASYN, serta evaluasi pada *dataset* yang lebih besar dan beragam. Pendekatan tambahan seperti *ensemble methods*, optimasi *hyperparameter*, *threshold optimization*, dan *cost-sensitive learning* juga berpotensi memperkuat penerapan IndoBERT pada sistem analisis sentimen di platform digital Indonesia.

DAFTAR PUSTAKA

[1] R. M. R. W. P. K. Atmaja and W. Yustanti, "Analisis Sentimen Customer Review Aplikasi Ruang Guru dengan Metode BERT (Bidirectional Encoder Representations from Transformers)," *Jeisbi*, vol. 02, no. 03, p. 2021, 2021.

[2] F. A. D. Aryanti, A. Luthfiarta, and D. A. I. Soeroso, "Aspect-Based Sentiment Analysis with LDA and IndoBERT Algorithm on Mental Health App: Riliv," *J. Appl. Informatics Comput.*, vol. 9, no. 2, pp. 361–375, 2025, doi: 10.30871/jaic.v9i2.8958.

[3] R. Kusnadi, Y. Yusuf, A. Andriantony, R. Ardian Yaputra, and M. Caintan, "Analisis Sentimen Terhadap Game Genshin Impact Menggunakan Bert," *Rabit J. Teknol. dan Sist. Inf. Univrab*, vol. 6, no. 2, pp. 122–129, 2021, doi: 10.36341/rabit.v6i2.1765.

[4] J. U. S. Lazuardi and A. Juarna, "Analisis Sentimen Ulasan Pengguna Aplikasi Joox Pada Android Menggunakan Metode Bidirectional Encoder Representation From Transformer (Bert)," *J. Ilm. Inform. Komput.*, vol. 28, no. 3, pp. 251–260, 2023, doi: 10.35760/ik.2023.v28i3.10090.

[5] Vidya Chandradev, I Made Agus Dwi Suarjaya, and I Putu Agung Bayupati, "Analisis Sentimen Review Hotel Menggunakan Metode Deep Learning BERT," *J. Buana Inform.*, vol. 14, no. 02, pp. 107–116, 2023, doi: 10.24002/jbi.v14i02.7244.

[6] M. A. Nugraha, M. I. Mazdadi, A. Farmadi, Muliadi, and T. H. Saragih, "Penyeimbangan Kelas SMOTE dan Seleksi Fitur Ensemble Filter pada Support Vector Machine untuk Klasifikasi Penyakit Liver," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 10, no. 6, pp. 1273–1284, 2023, doi: 10.25126/jtiik.2023107234.

[7] M. P. Pulungan, A. Purnomo, and A. Kurniasih, "Penerapan SMOTE untuk Mengatasi Imbalance Class dalam Klasifikasi Kepribadian MBTI Menggunakan Naive Bayes Classifier," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 11, no. 5, pp. 1033–1042, 2024, doi: 10.25126/jtiik.2024117989.

[8] Muhammad Bayu Nugroho, Akhmad Khanif Zyen, and Nur Aeni Widiastuti, "Multiclass Sentiment Analysis of Electric Vehicle Incentive Policies Using IndoBERT and DeBERTa Algorithms," *J. Appl. Informatics Comput.*, vol. 9, no. 3, pp. 910–919, 2025, doi: 10.30871/jaic.v9i3.9511.

[9] I. D. Apostolopoulos, "Investigating the Synthetic Minority Class Oversampling Technique (Smote) on an Imbalanced Cardiovascular Disease (Cvd) Dataset," *Int. J. Eng. Appl. Sci. Technol.*, vol. 04, no. 09, pp. 431–434, 2020, doi: 10.33564/ijeast.2020.v04i09.058.

[10] L. Mahmoudi and M. Salem, "BalBERT: A New Approach to Improving Dataset Balancing for Text Classification," *Rev. d'Intelligence Artif.*, vol. 37, no. 2, pp. 425–431, 2023, doi: 10.18280/ria.370219.

[11] T. Wongvorachan, S. He, and O. Bulut, "A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining," *Inf.*, vol. 14, no. 1, 2023, doi: 10.3390/info14010054.

[12] D. C. Li, Q. S. Shi, Y. S. Lin, and L. S. Lin, "A Boundary-Information-Based Oversampling Approach to Improve Learning Performance for Imbalanced Datasets," *Entropy*, vol. 24, no. 3, 2022, doi: 10.3390/e24030322.

[13] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J. Big Data*, vol. 11, no. 1, 2024, doi: 10.1186/s40537-024-00943-4.

[14] I. Araf, A. Idri, and I. Chairi, *Cost-sensitive learning for imbalanced medical data: a review*, vol. 57, no. 4. Springer Netherlands, 2024. doi: 10.1007/s10462-023-10652-8.

[15] Y. Feng, M. Zhou, and X. Tong, "Imbalanced classification: A paradigm-based review," *Stat. Anal. Data Min.*, vol. 14, no. 5, pp. 383–406, 2021, doi: 10.1002/sam.11538.

[16] M. A. Fathin, Y. Sibaroni, and S. S. Prasetyowati, "Handling Imbalance Dataset on Hoax Indonesian Political News Classification using IndoBERT and Random Sampling," *J. Media Inform. Budidarma*, vol. 8, no. 1, p. 352, 2024, doi: 10.30865/mib.v8i1.7099.

[17] C. Yang, E. A. Fridgeirsson, J. A. Kors, J. M. Reps, and P. R. Rijnbeek, "Impact of random oversampling and random undersampling on the performance of prediction models developed using observational health data," *J. Big Data*, vol. 11, no. 1, 2024, doi: 10.1186/s40537-023-00857-7.

[18] P. Wibowo and C. Fatichah, "An in-depth performance analysis of the oversampling techniques for high-class imbalanced dataset," *Regist. J. Ilm. Teknol. Sist. Inf.*, vol. 7, no. 1, pp. 63–71, 2021, doi: 10.26594/register.v7i1.2206.

[19] Y. A. Sir and A. H. H. Soepranoto, "Pendekatan Resampling Data Untuk Menangani Masalah Ketidakseimbangan Kelas," *J. Komput. dan Inform.*, vol. 10, no. 1, pp. 31–38, 2022, doi: 10.35508/jicon.v10i1.6554.

[20] S. Bej, K. Schulz, P. Srivastava, M. Wolfien, and O. Wolkenhauer, "A Multi-Schematic Classifier-Independent Oversampling Approach for Imbalanced Datasets," *IEEE Access*, vol. 9, pp. 123358–123374, 2021, doi: 10.1109/ACCESS.2021.3108450.

[21] S. F. Taskiran, B. Turkoglu, E. Kaya, and T. Asuroglu, "A comprehensive evaluation of oversampling techniques for enhancing text classification performance," *Sci. Rep.*, vol. 15, no. 1, pp. 1–20, 2025, doi: 10.1038/s41598-025-05791-7.

[22] D. A. Sani, "A Random Oversampling and BERT-based Model Approach for Handling Imbalanced Data in Essay Answer Correction," *J. Infotel*, vol. 16, no. 4, pp. 729–739, 2024, doi: 10.20895/infotel.v16i4.1224.

[23] M. Y. Ridho and E. Yulianti, "From Text to Truth: Leveraging IndoBERT and Machine Learning Models for Hoax Detection in Indonesian News," *J. Ilm. Tek. Elektro Komput. dan Inform.*, vol. 10, no. 3, pp. 544–555, 2024, doi: 10.26555/jiteki.v10i3.29450.

[24] X. Wang and L. Aitchison, "How to set AdamW's weight decay as you scale model and dataset size," 2025, [Online]. Available:

http://arxiv.org/abs/2405.13698
[25] W. Utomo, "IDN App." Accessed: Jul. 30, 2025. [Online]. Available: https://www.idn.app/about
[26] J. B. Wang, C. A. Zou, and G. H. Fu, "AWSMOTE: An SVM-Based Adaptive Weighted SMOTE for Class-Imbalance Learning," *Sci. Program.*, vol. 2021, 2021, doi: 10.1155/2021/9947621.
[27] A. R. Putra and D. E. Ratnawati, "Analisis Sentimen Berbasis Aspek pada Aplikasi Mobile Menggunakan Naïve Bayes berdasarkan Ulasan Pengguna Playstore (Studi Kasus : Jconnect Mobile)," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 12, no. 2, pp. 293–300, 2025, doi: 10.25126/jtiik.2025127556.
[28] H. Imaduddin, F. Y. A'la, and Y. S. Nugroho, "Sentiment Analysis in Indonesian Healthcare Applications using IndoBERT Approach," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 8, pp. 113–117, 2023, doi: 10.14569/IJACSA.2023.0140813.
[29] R. R. Suryono, "Sentiment Classification of Indonesian-Language Roblox Reviews Using IndoBERT with SMOTE Optimization," vol. 9, no. 4, pp. 1868–1877, 2025.
[30] E. Eskiyaturrofikoh and R. R. Suryono, "Analisis Sentimen Aplikasi X Pada Google Play Store Menggunakan Algoritma Naïve Bayes Dan Support Vector Machine (Svm)," *JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.*, vol. 9, no. 3, pp. 1408–1419, 2024, doi: 10.29100/jipi.v9i3.5392.
[31] B. Wilie *et al.*, "IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding," pp. 843–857, 2024, doi: 10.18653/v1/2020.aacl-main.85.
[32] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP," *COLING 2020 - 28th Int. Conf. Comput. Linguist. Proc. Conf.*, pp. 757–770, 2020, doi: 10.18653/v1/2020.coling-main.66.
[33] M. F. Ashidiq, L. Muflikhah, and B. D. Setiawan, "Deteksi Nefropati Diabetik Pada Pasien Diabetes Melitus Menggunakan Regresi Logistik," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. 2548–964, 2025, [Online]. Available: http://j-ptiik.ub.ac.id
[34] M. Pota, M. Ventura, R. Catelli, and M. Esposito, "An effective bert-based pipeline for twitter sentiment analysis: A case study in Italian," *Sensors (Switzerland)*, vol. 21, no. 1, pp. 1–21, 2021, doi: 10.3390/s21010133.
[35] R. Nihalani and K. Shah, "Enhancing Grammatical Error Detection using BERT with Cleaned Lang-8 Dataset," 2024.