Klasifikasi Ulasan Pengguna Untuk Pemeliharaan Perangkat Lunak Menggunakan IndoBERT-BiLSTM (Studi Kasus Mypertamina)

1st Muhammad Ardhy Satrio Jati
(Fakultas Teknik)
Universitas Lambung Mangkurat
Banjarmasin, Indonesia
ardhy.satrio27@gmail.com

2nd Andreyan Rizky Baskara
(Fakultas Teknik)
Universitas Lambung Mangkurat
Banjarmasin, Indonesia
andreyan.baskara@ulm.ac.id

Abstrak*—Ulasan aplikasi yang berguna memiliki peran penting bagi pengembang dalam pemeliharaan kualitas dan performa aplikasi. Namun, dengan meningkatnya penggunaan aplikasi dan jumlah ulasan yang diberikan, tidak semua ulasan bersifat konstruktif, beberapa ulasan hanya berisi cacian atau pujian tanpa memberikan saran atau masukan yang membangun. Memilah ribuan ulasan secara manual membutuhkan waktu dan tenaga yang besar. Oleh karena itu, dibutuhkan metode yang efisien dan efektif untuk mengidentifikasi ulasan yang berguna. Dalam penelitian ini peneliti menggunakan metode word embedding IndoBERT dan classifier BiLSTM untuk klasifikasi ulasan yang berguna. Hasil eksperimen menunjukkan bahwa Model IndoBERT-BiLSTM dengan konfigurasi learning rate sebesar 2e-5, dropout probability sebesar 0,2, dan batch size 16 mencapai hasil terbaik dengan nilai akurasi sebesar 95,49% dan menunjukkan peningkatan akurasi sebesar 1.16% dibandingkan dengan model fine-tuned IndoBERT.

***Kata Kunci*— Bi-LSTM, IndoBERT, Klasifikasi, Ulasan Aplikasi yang Berguna**

Abstract*—Useful app reviews have an important role for developers in maintaining the quality and performance of their applications. However, with the increasing usage of apps and the number of reviews received, not all reviews are constructive; some reviews consist of mere criticism or praise without providing constructive suggestions or feedback. Manually sorting through thousands of reviews is time-consuming and resource-intensive. Therefore, an efficient and effective method is required to identify useful reviews. In this study, researchers employed the word embedding method called Indonesian based on the Bidirectional Encoder Representations from Transformers (IndoBERT) and the classifier called Bidirectional Long Short-Term Memory (Bi-LSTM) for classifying useful reviews. The experimental results demonstrate that the IndoBERT-BiLSTM model, with a learning rate configuration of 2e-5, dropout probability of 0.2, and batch size of 16, achieved the best performance with an accuracy of 95.49% and showed a 1.16% improvement compared to the fine-tuned IndoBERT model.

***Keywords*— Bi-LSTM, Classification, IndoBERT, Useful App Reviews**

I. PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi telah memberikan dampak besar pada pertumbuhan dan evolusi aplikasi, yang kini memainkan peran sentral dalam mendukung berbagai aktivitas masyarakat. Aplikasi menjadi produk teknologi informasi yang terus berkembang dan memiliki peran yang signifikan dalam memfasilitasi beragam kegiatan masyarakat modern. Pada kuartal ketiga tahun 2022, laporan menunjukkan bahwa Google Play Store memiliki lebih dari 3,55 juta aplikasi seluler, mencerminkan pertumbuhan pesat dalam industri aplikasi dalam beberapa tahun terakhir [1]. Penetrasi luas teknologi, khususnya penggunaan *smartphone*, juga terlihat dari estimasi Statista untuk tahun 2023, di mana jumlah pengguna *smartphone* di seluruh dunia diprediksi mencapai 6,92 miliar, mencakup 86,29% dari populasi global [2].

Peningkatan penggunaan aplikasi Android berdampak pada meningkatnya jumlah ulasan yang diberikan oleh pengguna. Ulasan ini memiliki nilai penting bagi pengembang, memberikan wawasan berharga dalam mengevaluasi kualitas dan performa aplikasi. Informasi berharga seperti laporan bug, permintaan fitur, dan saran untuk meningkatkan pengalaman pengguna sering terkandung dalam ulasan tersebut [3]. Meski demikian, tidak semua ulasan memberikan masukan yang bermanfaat, beberapa hanya berupa pujian atau kritik tanpa detail yang memadai. Ulasan yang memberikan *insight* spesifik, seperti menyebut fitur yang disukai namun juga menunjukkan bug yang perlu diperbaiki, sangat membantu pengembang untuk fokus pada perbaikan yang diperlukan. Namun, jumlah ulasan yang besar menciptakan tantangan bagi pengembang dalam menilai berbagai ulasan yang beragam [4]. Meskipun kolom ulasan di App Store memudahkan pengguna dalam memberikan masukan, bagi pengembang, mengevaluasi ribuan ulasan yang beragam memerlukan waktu dan energi. Oleh karena itu, diperlukan metode efektif untuk menyaring ulasan yang bermanfaat dan membantu pengembang meningkatkan kualitas dan performa aplikasi mereka.

Dengan berkembangnya teknologi, banyak metode dan algoritma yang dapat digunakan untuk memilah ulasan aplikasi yang berguna untuk pemeliharaan perangkat lunak. Salah satunya adalah pendekatan melalui kecerdasan buatan seperti *Natural Language Processing* (NLP), yang memanfaatkan analisis teks untuk memahami dan mengekstrak informasi relevan [5]. Penggunaan *deep learning* dalam NLP juga telah berhasil mengungguli model-model sebelumnya dalam berbagai tugas [6]. Beberapa penelitian sebelumnya telah mengembangkan model *deep learning* yang menggunakan N-gram CNN sebagai *Word Embedding* serta Bi-LSTM *classifier* untuk mendeteksi ulasan spam. Kombinasi model ini menunjukkan performa unggul dalam deteksi ulasan spam dibandingkan metode lainnya, dengan nilai *Accuracy* mencapai 86,5%, F1 89,3%, *Precission* 86,1%, dan *Recall* 92,7 [7]. Studi lain menunjukkan bahwa model NLM seperti BERT memiliki nilai F1 klasifikasi yang lebih baik dalam mengklasifikasikan ulasan aplikasi [8].

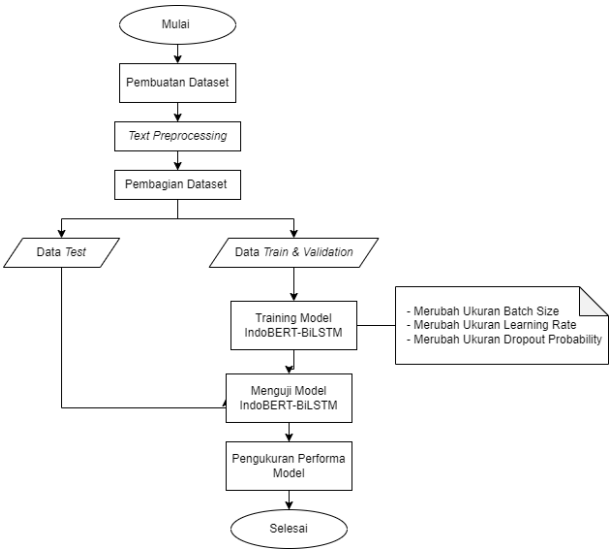
Pengembangan metode klasifikasi teks juga mencakup eksplorasi penggunaan LSTM pada model BERT seperti untuk mengidentifikasi berita palsu dengan memperhatikan judul dan kontennya, dan hasil penelitian ini terdapat peningkatan akurasi yang signifikan dibandingkan model *pre-trained* BERT [9]. Kombinasi metode BERT dan LSTM juga digunakan dalam penelitian pada bahasa Indonesia, contohnya

menggabungkan IndoBERT dan LSTM untuk mengklasifikasikan tweet menjadi beberapa kelas. Hasilnya menunjukkan presisi hingga 92,99% [10]. Penelitian lain membandingkan BERT-BiLSTM dengan *word embedding* tradisional seperti Word2Vec dan GloVe dalam model *deep learning* CNN dan LSTM, menunjukkan akurasi hingga 93% [11]. Pendekatan ini memanfaatkan dua arah (*Bidirectional*) dari BiLSTM untuk memahami pola kalimat, mengurangi kesalahan, dan meningkatkan kualitas klasifikasi.

Berdasarkan beberapa penelitian sebelumnya, penelitian ini menggunakan metode *word embedding* IndoBERT dan *classifier* BiLSTM untuk klasifikasi ulasan pengguna untuk pemeliharaan perangkat lunak, dengan data ulasan yang digunakan berasal dari ulasan aplikasi MyPertamina. Penggunaan *embedding* IndoBERT diharapkan dapat menghasilkan representasi kata yang lebih baik karena model telah dilatih pada bahasa Indonesia, sehingga memungkinkannya untuk lebih memahami bahasa Indonesia yang digunakan dalam ulasan aplikasi. Selain itu, dengan menggunakan metode BiLSTM sebagai *classifier*, diharapkan dapat meningkatkan akurasi klasifikasi ulasan yang berguna. Metode BiLSTM memungkinkan model untuk lebih memahami konteks ulasan dengan memproses informasi dalam dua arah (maju dan mundur) secara simultan [12]. Hasil penelitian ini diharapkan dapat membantu meningkatkan kualitas aplikasi MyPertamina dan memberikan manfaat bagi penggunanya.

II. METODOLOGI PENELITIAN

Tujuan dari penelitian ini adalah untuk mengevaluasi performa model klasifikasi IndoBERT-BiLSTM dalam klasifikasi ulasan pengguna untuk pemeliharaan perangkat lunak, dan membandingkannya dengan model pre-trained IndoBERT. Langkah-langkah yang dijalankan dalam penelitian ini dijelaskan pada Gambar 1.



Gambar 1. Prosedur Penelitian

A. Pembuatan Dataset

Data yang menjadi subjek penelitian ini dihimpun dari ulasan pengguna aplikasi MyPertamina di Google Play Store dengan menggunakan bahasa Indonesia. Setelah data ulasan aplikasi berhasil diambil dari Play Store, langkah selanjutnya adalah melakukan pembuatan dataset dengan melakukan proses pelabelan pada data ulasan. Proses pelabelan ini dilakukan secara manual oleh peneliti. Dalam proses ini, setiap ulasan dianalisis dan diberi label klasifikasi sesuai dengan isi ulasan tersebut. Penentuan label ulasan yang berguna atau tidak berguna mengikuti kriteria yang telah ditetapkan oleh penelitian sebelumnya yang berkaitan dengan ulasan aplikasi yang memiliki relevansi bagi pengembang dalam tugas pemeliharaan perangkat lunak [4] [13]. Pendekatan ini memberikan konsistensi dalam proses pelabelan data. Data yang telah diberi label ini kemudian digunakan sebagai dataset yang akan digunakan pada penelitian ini. Kriteria pelabelan ulasan yang digunakan dalam pelabelan data bisa dilihat pada Tabel I.

TABEL I. KRITERIA IDENTIFIKASI LABEL ULASAN

Ulasan yang Berguna	Ulasan Tidak Berguna
Memberikan ide, saran, atau kebutuhan untuk meningkatkan atau memperbaiki aplikasi	Hanya mengungkapkan apresiasi atau sebaliknya
Melaporkan masalah, bug atau laporan kegagalan aplikasi	Menjelaskan pengalaman di mana aplikasi terbukti membantu
Meminta konten yang belum ada dalam aplikasi	Merujuk ke aplikasi lain, misalnya untuk dibandingkan
Meminta informasi atau bantuan dari pengguna lain atau pengembang	Informasi yang tidak memiliki makna atau tidak berhubungan dengan aplikasi

Setelah proses pemilahan dan pelabelan selesai, total dataset yang terkumpul berjumlah 10.410 data ulasan aplikasi, yang dibagi menjadi dua label, yaitu label satu untuk ulasan yang berguna, dan label nol untuk ulasan yang tidak berguna. Terdapat 5.205 data yang diberi label sebagai ulasan berguna, dan 5.205 data lainnya diberi label sebagai ulasan tidak berguna. Tabel II menampilkan contoh data ulasan beserta labelnya.

TABEL II CONTOH DATA BERLABEL

No	Kalimat	Label
1	Susah login, harus masukan verifikasi email, tapi kode verifikasi gak ada masuk ke email... aneh	1
2	Aplikasi My pertamina sangat membantu untuk pembelian bensin, sudah ada peningkatan kecepatan pembayaran dari scan QR loh, membantu juga agar kita tau perbulan nya kita habis berapa untuk beli bensin	0

B. Text Preprocessing

Text Preprocessing merupakan tahapan penting dalam pengolahan data yang bertujuan untuk membersihkan, mengubah, dan mempersiapkan data sebelum masuk ke tahap berikutnya. Dalam konteks penelitian ini, proses praproses teks meliputi serangkaian langkah, termasuk *noise removal, case folding, normalization, dan tokenizing*:

1) Noise Removal

Proses *noise removal* dilakukan untuk membersihkan dataset dari unsur-unsur yang tidak relevan atau tidak berkontribusi dalam proses klasifikasi. Karakter-karakter seperti tanda baca, emotikon, dan simbol-simbol khusus dapat berdampak pada performa model klasifikasi. Sebagai contoh, Tabel III menunjukkan contoh data setelah melalui tahap penghilangan *noise.*

TABEL III. PROSES NOISE REMOVAL

Sebelum	Sesudah
Baru instal dan pengen daftar tgl 1 July ini ,ternyata ga bisa dpt otp dan terjadi kesalahan...ðŸ˜…,metode	Baru instal dan pengen daftar tgl July ini ternyata ga bisa dpt otp dan terjadi kesalahan metode

Sebelum	Sesudah
pembayaran ga ada pilihan harus link aja...ðŸ¤”ðŸ¤	pembayaran ga ada pilihan harus link saja

2) *Case Folding*

Tahapan *case folding* bertujuan untuk mengubah semua huruf dalam teks menjadi huruf kecil (*lowercase*). Hal ini dilakukan untuk menghindari perbedaan penulisan yang tidak relevan dan memastikan kata-kata seragam dalam dataset. Dengan mengubah semua huruf menjadi huruf kecil, perbedaan dalam penulisan huruf besar dan kecil tidak akan mempengaruhi interpretasi kata-kata yang memiliki arti sama. Hal ini membantu memastikan konsistensi dalam dataset. Tabel IV menunjukkan contoh data setelah melalui tahap *case folding*:

TABEL IV. PROSES CASE FOLDING

Sebelum	Sesudah
Baru instal dan pengen daftar tgl July ini ternyata ga bisa dpt otp dan terjadi kesalahan metode pembayaran ga ada pilihan harus link aja¤	baru instal dan pengen daftar tgl july ini ternyata ga bisa dpt otp dan terjadi kesalahan metode pembayaran ga ada pilihan harus link aja

3) *Normalization*

Langkah normalisasi bertujuan untuk meningkatkan konsistensi dan akurasi teks dengan mengubah kata-kata slang atau informal menjadi bentuk formal yang umum digunakan dalam bahasa Indonesia. Dalam tahap normalisasi, digunakan kamus atau daftar kata-kata slang yang kemudian diubah menjadi bentuk formal yang dikenal dalam bahasa Indonesia. Kamus yang digunakan adalah kamus bahasa tidak baku yang berasal dari (https://raw.githubusercontent.com/nasalsabila/kamus-alay/master/colloquial-indonesian-lexicon.csv), yang berisi kata-kata slang beserta kata-kata formal yang menjadi penggantinya Tabel V menunjukkan contoh data setelah melalui tahap *normalization*:

TABEL V. PROSES CASE FOLDING

Sebelum	Sesudah
baru instal dan pengen daftar tgl july ini ternyata ga bisa dpt otp dan terjadi kesalahan metode pembayaran ga ada pilihan harus link aja¤	baru instal dan pengin daftar tanggal july ini ternyata enggak bisa dapat otp dan terjadi kesalahan metode pembayaran enggak ada pilihan harus link aja

4) *Tokenizing*

Tokenisasi adalah langkah untuk memecah teks menjadi bagian-bagian yang lebih kecil, seperti kata-kata dan frasa. Dalam penelitian ini, peneliti menggunakan BERTTokenizer, sebuah alat tokenisasi yang dioptimalkan untuk model berbasis BERT. Proses ini juga melibatkan penambahan token khusus, seperti [CLS] pada awal kalimat untuk menandai awal teks, dan token [SEP] sebagai penanda pemisah di akhir kalimat. Jika panjang kalimat kurang dari panjang maksimal, token [PAD] ditambahkan untuk melengkapi panjang. Setelah teks berhasil dipecah menjadi token, tahap selanjutnya adalah pengkodean menggunakan *tokenizer* yang disesuaikan dengan indeks *vocabulary* pre-trained IndoBERT. Dalam proses ini, setiap token diubah menjadi representasi numerik yang mencerminkan kata atau frasa tersebut. Contoh hasil tokenisasi dapat dilihat pada Tabel VI.

TABEL VI. PROSES NOISE REMOVAL

Tahapan	Hasil
Tokenisasi	'baru', 'instal', 'dan', 'pengin', 'daftar', 'tanggal', 'july', 'ini', 'ternyata', 'enggak', 'bisa', 'dapat', 'ot', '##p', 'dan', 'terjadi', 'kesalahan', 'metode', 'pembayaran', 'enggak', 'ada', 'pilihan', 'harus', 'link', 'aja'
Penambahan token khusus	'[CLS]', 'baru', 'instal', 'dan', 'pengin', 'daftar', 'tanggal', 'july', 'ini', 'ternyata', 'enggak', 'bisa', 'dapat', 'ot', '##p', 'dan', 'terjadi', 'kesalahan', 'metode', 'pembayaran', 'enggak', 'ada', 'pilihan', 'harus', 'link', 'aja', [SEP], [PAD] , [PAD], ..., [PAD]
Encoding	2, 1202, 1339, 259, 1305, 7650, 816, 7650, 186, 5969, 301, 259, 271, 4634, 92, 2468, 1339, 731, 92, 3043, 4017, 41, 308, 13065, 79, 3043, 469, 6878, 79, 366, 308, 7734, 1339, 2174, 1389, 2526, 2088, 6595, 90, 8866, 551, 2557, 9409, 1997, 318, 180, 3298, 62, 406, 1107, 21192, 599, 3072, 419, 229, 15951, 41, 7835, 3975, 259, 3290, 26, 1339, 2557, 18348, 422, 1320, 6878, 26, 28098, 8676, 2174, 3, 0, 0, ..., 0
Attention mask¤	1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, ..., 0

C. *Pembagian Dataset*

Setelah melalui tahap preprocessing, dataset yang terdiri dari 10.410 data siap untuk dikelompokkan menjadi tiga set data yang berbeda, yaitu data latih (*training*), data validasi (*validation*), dan data uji (*testing*). Proses pembagian dataset ini dilakukan menggunakan pustaka scikit-learn yang menyediakan alat-alat yang mempermudah dalam mengelompokkan data. Dalam pembagian dataset ini mengikuti perbandingan proporsi yang umum digunakan, yaitu 70% untuk set latih, 20% untuk set validasi, dan 10% untuk set uji. Informasi detail tentang pembagian dataset dapat ditemukan dalam Tabel VII, yang memberikan gambaran tentang jumlah data yang ada dalam masing-masing set data yang telah dibuat. Hal ini bertujuan untuk memastikan bahwa model dapat diuji dengan baik dan memiliki performa yang baik pada data yang belum pernah dilihat sebelumnya.

TABEL VII. PEMBAGIAN DATASET

Rincian Data	Training (70%)		Validation (20%)		Testing (10%)	
Label	1	0	1	0	1	0
Jumlah	3644	3645	1040	1040	521	520
Total	7289		2080		1041	
	10.410					

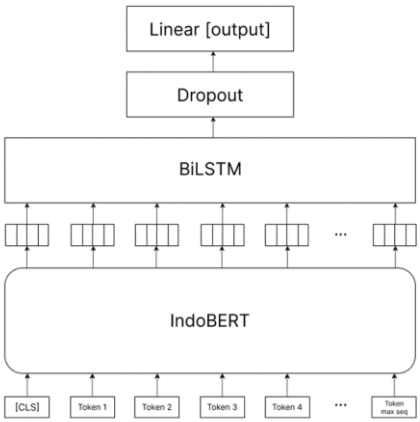
D. *Training Model*

Proses pelatihan model dilaksanakan menggunakan data latih (set *training*) untuk menyesuaikan parameter-parameter yang diperlukan, eperti bobot (*weights*) dan bias, agar model dapat mengoptimalkan fungsi kerugian (*loss function*). Langkah berikutnya adalah menerapkan proses *backpropagation*, yang bertujuan untuk memperbaiki kesalahan dalam model selama proses pelatihan. Selanjutnya, set data validasi (set *validation*) digunakan untuk mengevaluasi performa model selama proses pelatihan, serta untuk mencegah terjadinya *overfitting* atau *underfitting* pada model. Teknik seperti *early stopping* atau regularisasi juga diterapkan pada set validasi. Setelah tahap pelatihan selesai dan *hyperparameter* telah ditentukan, set data uji (set testing) digunakan untuk menguji performa model pada data baru yang belum pernah dilihat sebelumnya. Melalui pengujian

pada data uji, kita dapat mengestimasi kinerja model pada data baru dan menganalisis keefektifan model tersebut.

1) Model IndoBERT- BiLSTM

Pada tahap pelatihan model IndoBERT-BiLSTM, masukan berupa kalimat yang telah melewati proses tokenisasi akan diterima oleh model IndoBERT yang menghasilkan keluaran berupa *last hidden state* dari model IndoBERT untuk setiap token dalam bentuk sekuens. Kemudian, sekuens tersebut dimasukkan ke dalam layer BiLSTM untuk menangkap informasi secara sekuensial. Sekuens tersebut kemudian diteruskan ke lapisan BiLSTM untuk menangkap informasi secara sekuensial. Hasil akhir dari lapisan BiLSTM, yaitu *final hidden state*, melewati lapisan *dropout* untuk mencegah *overfitting* dan meningkatkan kemampuan generalisasi. Akhirnya, lapisan output melakukan transformasi linear untuk menghasilkan *logits*, yang merupakan keluaran dari model untuk tugas klasifikasi. Arsitektur model IndoBERT-BiLSTM dijelaskan secara visual dalam Gambar 2.



Gambar 2. Arsitektur Model IndoBERT-BiLSTM

Proses pelatihan dilakukan dalam beberapa skenario yang berbeda untuk menguji parameter-parameter yang bervariasi. Parameter yang diuji meliputi *dropout probability*, mengacu pada studi sebelumnya [14], serta *learning rate* dan *batch size*, sesuai rekomendasi konfigurasi untuk model berbasis BERT [15]. Setiap skenario melibatkan 20 epoch untuk memastikan model memiliki waktu cukup untuk mempelajari pola bahasa Indonesia. Parameter yang digunakan dalam pelatihan model IndoBERT-BiLSTM, mencakup variasi *learning rate, dropout probability*, dan *batch size* yang dieksplorasi dalam pelatihan. Daftar parameter yang digunakan pada pelatihan model IndoBERT-BiLSTM dicantumkan dalam Tabel VIII.

TABEL VIII. PARAMETER PELATIHAN MODEL INDOBERT-BILSTM

Parameter	Konfigurasi
Epoch	20
Max sequence	128
LSTM layer input size	768
LSTM layer hidden size	768
Loss Function	binary cross-entropy
Optimizer	Adam
Learning Rate	{1e-6, 2e-5, 5e-5}
Dropout Probability	{0.1, 0.2}
Batch Size	{16, 32}

2) Model Fine-tuned IndoBERT

Pelatihan model IndoBERT dilakukan dalam 20 epoch untuk memastikan model memiliki cukup waktu untuk memahami dan menyesuaikan dengan pola bahasa Indonesia. Teknik *early stopping* juga diterapkan untuk memungkinkan pelatihan berhenti lebih awal jika tidak ada peningkatan signifikan dalam performa model pada data validasi. Pelatihan model IndoBERT dilakukan dengan mempertimbangkan *hyperparameter* yang diambil dari penelitian yang dilakukan oleh Fajri Koto dan rekan-rekan dalam studi tentang analisis sentimen [16]. Rincian parameter yang digunakan dalam pelatihan model IndoBERT disajikan dalam Tabel IX.

TABEL IX. PARAMETER PELATIHAN MODEL INDOBERT-BILSTM

Parameter	Konfigurasi
Epoch	20
Max Sequence	128
Loss Function	binary cross-entropy
Optimizer	Adam
Learning Rate	2e-5
Dropout Probability	0.1
Batch Size	16

E. Pengujian Model

Proses pengujian dilakukan dengan membandingkan kinerja model IndoBERT-BiLSTM dan model IndoBERT yang telah difine-tuning untuk membuktikan bahwa penggabungan antara model IndoBERT dan Bidirectional-LSTM adalah pilihan terbaik dalam mengklasifikasikan ulasan aplikasi yang berguna dalam bahasa Indonesia. Nilai-nilai seperti akurasi, presisi, *recall*, dan *F1-score* dihitung menggunakan *Confusion Matrix*. Pengujian ini menggunakan data dari set pengujian (test set).

Pengujian mencakup perbandingan berbagai konfigurasi ukuran batch, *learning rate*, dan juga ukuran *dropout probability*. Proses ini bertujuan untuk mengevaluasi bagaimana variasi parameter ini mempengaruhi kinerja model. Beberapa parameter yang diuji meliputi ukuran batch dengan nilai 16 dan 32, serta variasi *learning rate* yang akan diuji pada rentang nilai tertentu. Selain itu, ukuran *dropout probability* juga dieksplorasi untuk mengamati dampaknya terhadap performa model. Pengujian ini memberikan informasi penting mengenai konfigurasi parameter optimal yang dapat meningkatkan kinerja model dalam tugas klasifikasi ulasan aplikasi. Dengan membandingkan berbagai kombinasi parameter, penelitian ini dapat memilih konfigurasi terbaik yang memberikan hasil terbaik.

F. Evaluasi Model

Pada tahap evaluasi model, yang dilakukan setelah tahap pengujian model, kinerja model dinilai dengan menggunakan *confusion matrix*, alat pendukung yang diperlukan dalam mengevaluasi efektivitas model dalam melakukan klasifikasi. Matriks ini menyajikan rincian data dengan membuat tabel klasifikasi yang benar dan salah oleh *classfier*, membandingkan label aktual dengan label yang diklasifikasikan sehingga menghasilkan analisis rinci terhadap performa *classifier* [17]. Dalam evaluasi menggunakan *confusion matrix*, terdapat empat kemungkinan hasil; *true positives* (TP) untuk kasus positif yang diklasifikasikan dengan benar, *false negatives* (FN) untuk kasus negatif yang dilabeli secara salah, *true negatives* (TN) untuk kasus negatif yang diidentifikasi secara akurat, dan *false positives* (FP) untuk kasus positif yang diklasifikasikan dengan salah. Hasil ini memainkan peran penting dalam menentukan keakuratan dan kemanjuran pengklasifikasi. Dengan menggunakan nilai-nilai tersebut, metrik kinerja model seperti *precision, recall, f1 score*, dan *accuracy* dapat dihitung dari *confusion matrix*.

Nilai *Accuracy* menunjukkan proporsi data yang diklasifikasikan dengan benar dibandingkan dengan keseluruhan dataset, yang menawarkan penilaian keseluruhan dari klasifikasi model di semua kelas. Persamaan (1) merupakan rumus untuk menghitung nilai *accuracy.*

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

Nilai *Precission* mewakili proporsi data positif yang diklasifikasikan dengan benar dari semua data positif yang diprediksi, bertujuan untuk mengukur kemampuan model untuk mengklasifikasikan kelas yang relevan secara akurat. Persamaan (2) merupakan rumus untuk menghitung nilai *precission.*

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

Nilai *Recall* menunjukkan proporsi data positif yang diklasifikasikan dengan benar dari semua data positif yang sebenarnya, yang menunjukkan kapasitas model untuk mengidentifikasi dengan benar semua kelas yang relevan. Persamaan (3) merupakan rumus untuk menghitung nilai *recall.*

$$Recall = \frac{TP}{P} \quad (3)$$

F1 score merupakan kombinasi dari metrik *precission* dan *recall*, yang memadatkannya menjadi satu skor. Skor ini menyajikan perspektif yang seimbang pada kinerja model dan memberikan bobot yang lebih besar pada nilai yang rendah. Persamaan (4) merupakan rumus untuk menghitung nilai *f1 score*.

$$F_1 = 2 \times \frac{Precission \times Recall}{Precission + Recall} \quad (4)$$

Keterangan
TP : *True positive*
FP : *False positive*
TN : *True negative*
FN : *False negative*

III. HASIL DAN PEMBAHASAN

A. Training dan Pengujian Model

Pelatihan model dilakukan dengan menggunakan set pelatihan dari dataset. Selama fase pelatihan, strategi seperti *early stopping* juga diterapkan untuk mengoptimalkan efisiensi. Hal ini melibatkan proses pelatihan yang memungkinkan proses pelatihan dihentikan sebelum waktunya jika terjadi penurunan yang signifikan dalam kinerja model, sehingga penggunaan waktu dan sumber daya menjadi lebih efektif. Pelatihan model dilakukan dalam beberapa skenario untuk mengevaluasi berbagai parameter sesuai dengan konfigurasi yang telah ditentukan. Setiap skenario terdiri dari 20 epoch, memberikan waktu yang cukup bagi model untuk memahami dan beradaptasi dengan pola data dalam bahasa Indonesia.

Setelah tahap pelatihan yang melibatkan beragam skenario parameter seperti yang telah ditentukan sebelumnya, model menjalani tahap pengujian. Dalam tahap pengujian model, dilakukan perhitungan nilai akurasi, *precision, recall, f1 score* dengan menggunakan *confusion matrix.* Tahap pengujian menggunakan set tes yang terdiri dari 1041 data dari dataset. Hasil pengujian untuk setiap konfigurasi model disajikan pada Tabel X.

TABEL X. HASIL PENGUJIAN MODEL

Model	B. Size	Accuracy	F1-score	Recall	Precision
IndoBERT +BiLSTM, lr = 1e-6, dp=0.1	16	0.94524	0.94523	0.94545	0.94524
	32	0.94236	0.94235	0.94290	0.94235
IndoBERT +BiLSTM, lr = 1e-6, dp=0.2	16	0.94525	0.94524	0.94545	0.94524
	32	0.94525	0.94524	0.94532	0.94525
IndoBERT +BiLSTM, lr = 2e-5, dp=0.1	16	0.93372	0.93369	0.93441	0.93374
	32	0.94428	0.94428	0.94446	0.94428
IndoBERT +BiLSTM, lr = 2e-5, dp=0.2	16	0.95485	0.95485	0.95486	0.95486
	32	0.94525	0.94524	0.94553	0.94523
IndoBERT +BiLSTM, lr = 5e-5, dp=0.1	16	0.93564	0.93564	0.93566	0.93565
	32	0.95005	0.95004	0.95028	0.95006
IndoBERT +BiLSTM, lr = 5e-5, dp=0.2	16	0.93468	0.93466	0.93510	0.93466
	32	0.94333	0.94331	0.94370	0.94331
IndoBERT	16	0.94333	0.94330	0.94393	0.94331

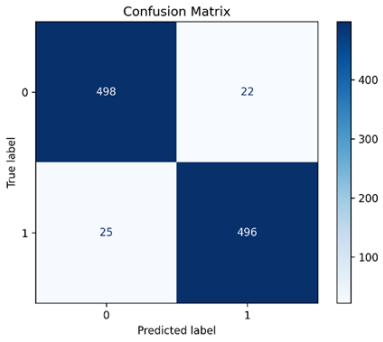
Hasil pengujian memperlihatkan bahwa model IndoBERT-BiLSTM, dengan *learning rate* (lr) 2e-5, *dropout probability* (dp) 0.2, dan ukuran batch 16, menunjukkan performa terbaik untuk akurasi, *F1-score, recall*, dan *precision*. Sebaliknya, model IndoBERT-BiLSTM dengan learning rate (lr) 2e-5, *dropout probability* (dp) 0.1, dan ukuran batch 32 mempunyai nilai terendah dalam hal akurasi, *F1-score*, *recall*, dan *precision*. Data-data tersebut menunjukkan bahwa model IndoBERT-BiLSTM dengan *learning rate* (lr) 2e-5, *dropout probability* (dp) 0.2, dan ukuran batch 16 mengungguli konfigurasi lain dalam pengujian ini.

B. Evaluasi Model

Hasil pengujian model IndoBERT-BiLSTM, yang dikonfigurasikan dengan *learning rate* (lr) 2e-5 dan probabilitas *dropout* (dp) 0,2, menunjukkan kinerja yang mengesankan. Model ini mencapai tingkat *Accuracy* sebesar 95,49%, *F1-score* sebesar 95,49%, *Precision* sebesar 95,49%, dan *Recall* sebesar 95,49%., melampaui kinerja model IndoBERT dengan akurasi 94,33%, F1-score 94,33%, recall 94,40%, dan presisi 94,33%, serta konfigurasi lain yang telah diujikan. Sehingga, model ini merupakan model dengan performa terbaik dalam mengklasifikasikan ulasan yang berguna untuk pemeliharaan perangkat lunak. Rincian tambahan mengenai konfigurasi model terbaik dapat ditemukan di Tabel XI.

TABEL XI. PARAMETER PELATIHAN MODEL INDOBERT-BILSTM

Parameter	Konfigurasi
Epoch	20
Max sequence	128
LSTM layer input size	768
LSTM layer hidden size	768
Loss Function	binary cross-entropy
Optimizer	Adam
Learning Rate	2e-5
Dropout Probability	0.2
Batch Size	16



Gambar 3. Confussion Matrix Model IndoBERT-BiLSTM

Pada tahap pengujian, model IndoBERT-BiLSTM menjalani uji coba dengan menggunakan dataset uji yang terdiri dari 1041 data yang belum pernah dilihat sebelumnya. Hasil klasifikasi dari model ini digunakan untuk membangun *Confusion Matrix*, yang ditampilkan pada Gambar 3. Berdasarkan data tersebut, model secara akurat mengklasifikasikan 994 data uji dari total 1041 data uji. Informasi lengkap mengenai jumlah hasil klasifikasi dalam dataset uji dapat dilihat pada Tabel XII di bawah ini.

TABEL XII. HASIL KLASIFIKASI PADA TEST SET

Label	Benar	Salah	Total	Persentase Kesalahan
1 (ulasan yang berguna)	496	22	518	2.11%
0 (ulasan tidak berguna)	498	25	523	2.40%
Total	994	47	1041	4.51%

Selain pengujian menggunakan test set, model ini juga diuji dengan 100 data baru yang tidak termasuk dalam dataset pelatihan, validasi, atau uji, sehingga memberikan gambaran yang lebih komprehensif dengan data yang sepenuhnya baru. Hasil pengujian pada 100 data baru ini menunjukkan tingkat keberhasilan klasifikasi sebesar 96%. Informasi terperinci mengenai hasil klasifikasi pada 100 data baru ini dapat ditemukan pada Tabel XIII.

TABEL XIII. HASIL KLASIFIKASI PADA 100 DATA BARU

Label	Benar	Salah	Total	Persentase Kesalahan
1 (ulasan yang berguna)	49	3	50	1.0%
0 (ulasan tidak berguna)	47	1	50	3.0%
Total	96	4	100	4.0%

Berdasarkan hasil pengujian model pada test set, model IndoBERT-BiLSTM mencapai akurasi 95,49% dalam mengategorikan ulasan menjadi berguna dan tidak berguna, yang menunjukkan kemahirannya dalam mengidentifikasi ulasan berguna yang menawarkan wawasan berharga bagi pengembang aplikasi. Beberapa contoh hasil klasifikasi yang benar pada set pengujian dapat ditemukan di Tabel XIV. Selain klasifikasi yang benar, terdapat juga contoh-contoh kesalahan klasifikasi, seperti yang ditunjukkan pada Tabel XV.

TABEL XIV. HASIL KLASIFIKASI YANG BENAR PADA TEST SET

No	Teks	Aktual	Klasifikasi	Ket
1	daftarnya sulit selalu kembali ke menu aplikasi buruk pertamina zonk	Useful	Useful	Benar
2	informasi yang di berikan cukup mudah cara membacanya dan untuk metode pembayarannya cukup mudah	Not_useful	Not_useful	Benar
3	pertamina akan bangkrut dan yang ada di belakangnya akan dibeginikan ingat doa orang terzalimi mudah diijabah	Not_useful	Not_useful	Benar
4	aplikasi yang tidak berguna sangat menyusahkan sudah daftar terus masuk apk malah keluar sendiri	Useful	Useful	Benar

TABEL XV. HASIL KLASIFIKASI YANG SALAH PADA TEST SET

No	Teks	Aktual	Klasifikasi	Ket
1	aplikasi sangat banyak bug dan malah menyusahkan pengguna sehingga bukannya memudahkan tapi malah menyebabkan antrian di spbu	Useful	Not_useful	Salah
2	kesulitan mengisi minyak karena atm jauh dan tidak ada cash semoga dengan ini semua kendala teratasi	Not_useful	Useful	Salah
3	aplikasi terkutuk loginnya seperti menangkap koruptor susah setengah mati	Useful	Not_useful	Salah

Walaupun model ini memiliki akurasi yang baik, masih ada beberapa aspek yang perlu mendapat perhatian lebih lanjut. Tabel XV mengilustrasikan beberapa contoh di mana model tidak berhasil mengklasifikasikan data, yang mengindikasikan adanya kasus-kasus yang menantang untuk analisis yang akurat dari model. Kalimat-kalimat tertentu bahkan tidak dapat diklasifikasikan dengan tepat oleh model, mungkin karena kerumitan bahasa dan variasi ulasan pengguna, yang menyebabkan beberapa situasi sulit untuk diprediksi secara akurat. Model ini juga mengalami kesulitan dalam mengenali makna yang ambigu dan konteks kiasan dalam kalimat ulasan.

Untuk menangani ketidakakuratan dalam klasifikasi, beberapa langkah perbaikan dapat dilakukan, seperti memperluas data pelatihan untuk meningkatkan keragaman dan representasi dari berbagai skenario ulasan, sehingga memungkinkan model untuk belajar secara lebih efektif dan komprehensif, dengan mengenali pola-pola yang relevan. Selain itu, mengevaluasi dan menyesuaikan konfigurasi model dapat dilakukan sebagai strategi perbaikan. Eksplorasi lebih lanjut yang melibatkan penggabungan model IndoBERT dengan metode pengklasifikasi lainnya, seperti *Convolutional Neural Network* (CNN) atau *Bidirectional Gated Recurrent Unit* (Bi-GRU), dapat memberikan wawasan baru untuk meningkatkan kinerja model dalam menangani skenario klasifikasi yang kompleks dan sulit. Dengan melakukan langkah-langkah ini, diharapkan model dapat mencapai performa yang lebih baik dalam mengklasifikasikan ulasan yang berguna dan tidak berguna dalam konteks aplikasi yang beragam.

IV. KESIMPULAN

Konfigurasi terbaik dari model IndoBERT-BiLSTM (dengan *learning rate* 2e-5, *dropout probability* 0.2, dan ukuran batch 16) mencapai tingkat akurasi yang memuaskan yaitu sebesar 95.49%. Pencapaian ini melampaui kinerja model *fine-tuned* IndoBERT sebesar 1.16%. Namun demikian, model ini masih menunjukkan keterbatasan dalam

mengklasifikasikan kalimat-kalimat yang ambigu atau kiasan secara tepat, sehingga membutuhkan tingkat pemahaman yang lebih dalam.

Saran untuk penelitian berikutnya adalah dengan menyertakan data ulasan dari berbagai jenis aplikasi di luar MyPertamina, yang bertujuan untuk menghasilkan model klasifikasi ulasan yang lebih universal yang mampu menangani kasus-kasus rumit dalam klasifikasi, termasuk kalimat-kalimat yang bersifat samar atau kiasan, yang tersebar luas di seluruh platform. Meningkatkan dataset dengan jumlah data ulasan yang lebih besar menjadi langkah penting untuk meningkatkan kinerja dan akurasi model, yang saat ini hanya terbatas pada 10.410 dataset. Dataset yang lebih besar diharapkan dapat meningkatkan kemampuan model dalam mengategorikan kalimat secara efektif. Selain itu, dengan mempertimbangkan penerapan klasifikasi multi-kelas untuk mengelompokkan ulasan ke dalam kelompok yang lebih relevan akan memberikan banyak wawasan bagi para pengembang. Selain itu, mengeksplorasi penggabungan model IndoBERT dengan metode pengklasifikasi yang lebih beragam, seperti *Convolutional Neural Network* (CNN) atau *Bidirectional Gated Recurrent Unit* (Bi-GRU), diharapkan dapat meningkatkan performa dan kemampuan model dalam menghadapi berbagai jenis data ulasan aplikasi. Dengan demikian, penelitian selanjutnya diharapkan dapat memberikan kontribusi yang signifikan terhadap evolusi dan peningkatan kualitas model klasifikasi ulasan aplikasi untuk pemeliharaan perangkat lunak.

REFERENCES

[1] Statista, "Number of available apps in the Google Play Store from 2nd quarter 2015 to 3rd quarter 2022," *Statista*. Accessed: Mar. 02, 2023. [Online]. Available: https://www.statista.com/statistics/289418/number-of-available-apps-in-the-google-play-store-quarter/

[2] Statista, "Number of smartphone subscriptions worldwide from 2016 to 2021, with forecasts from 2022 to 2027," *Statista*. Accessed: Mar. 02, 2023. [Online]. Available: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

[3] F. Palomba *et al.*, "User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps," in *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, IEEE, Sep. 2015, pp. 291–300. doi: 10.1109/ICSM.2015.7332475.

[4] D. Pagano and W. Maalej, "User feedback in the appstore: An empirical study," in *2013 21st IEEE International Requirements Engineering Conference (RE)*, IEEE, Jul. 2013, pp. 125–134. doi: 10.1109/RE.2013.6636712.

[5] Jalaj Thanaki, *Python Natural Language Processing*. Packt, 2017.

[6] A. Vaswani *et al.*, "Attention Is All You Need," *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6000–6010, 2017.

[7] Y. Liu, L. Wang, T. Shi, and J. Li, "Detection of spam reviews through a hierarchical attention architecture with N-gram CNN and Bi-LSTM," *Inf Syst*, vol. 103, p. 101865, Jan. 2022, doi: 10.1016/j.is.2021.101865.

[8] A. Araujo, M. Golo, B. Viana, F. Sanches, R. Romero, and R. Marcacini, "From Bag-of-Words to Pre-trained Neural Language Models: Improving Automatic Classification of App Reviews for Requirements Engineering," in *Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2020)*, Sociedade Brasileira de Computação - SBC, Oct. 2020, pp. 378–389. doi: 10.5753/eniac.2020.12144.

[9] N. Rai, D. Kumar, N. Kaushik, C. Raj, and A. Ali, "Fake News Classification using transformer based enhanced LSTM and BERT," *International Journal of Cognitive Computing in Engineering*, vol. 3, pp. 98–105, Jun. 2022, doi: 10.1016/j.ijcce.2022.03.003.

[10] T. Iskandar Zulkarnain Maulana Putra, Suprapto, and A. Farhan Bukhori, "Model Klasifikasi Berbasis Multiclass Classification dengan Kombinasi Indobert Embedding dan Long Short-Term Memory untuk Tweet Berbahasa Indonesia," *Jurnal Ilmu Siber dan Teknologi Digital (JISTED)*, vol. 1, no. 1, pp. 1–28, 2022, doi: 10.35912/jisted.v1i1.1509.

[11] M. P. David Junggu, Kusrini, and Sudarmawan, "Peningkatan Akurasi Klasifikasi Sentimen Ulasan Makanan Amazon Dengan Bidirectional LSTM Dan Bert Embedding," *Inspiration : Jurnal Teknologi Informasi dan Komunikasi*, vol. 10, no. 1, pp. 9–20, 2020.

[12] X. Yao, "Attention-based BiLSTM Neural Networks for Sentiment Classification of Short Texts," in *Proceedings of Information Science and Cloud Computing — PoS(ISCC 2017)*, Trieste, Italy: Sissa Medialab, Feb. 2018, p. 014. doi: 10.22323/1.300.0014.

[13] S. Panichella, A. Di Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "How can i improve my app? Classifying user reviews for software maintenance and evolution," in *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, IEEE, Sep. 2015, pp. 281–290. doi: 10.1109/ICSM.2015.7332474.

[14] D. Lu, "daminglu123 at SemEval-2022 Task 2: Using BERT and LSTM to Do Text Classification," in *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2022, pp. 186–189. doi: 10.18653/v1/2022.semeval-1.22.

[15] Google Research, "google-research/bert TensorFlow code and pre-trained models for BERT." https://github.com/google-research/bert (accessed Jun. 01, 2023).

[16] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP," in *Proceedings of the 28th International Conference on Computational Linguistics*, Stroudsburg, PA, USA: International Committee on Computational Linguistics, 2020, pp. 757–770. doi: 10.18653/v1/2020.coling-main.66.

[17] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit Lett*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.