Terjemahan dari 10.22146/jnteti.v13i1.8532

Studi dan Analisis *Hyperparameter Tuning* IndoBERT Dalam Pendeteksian Berita Palsu

Anugerah Simanjuntak[1], Rosni Lumbantoruan[1], Kartika Sianipar[1], Rut Gultom[1], Mario Simaremare[1], Samuel Situmeang[1], Erwin Panggabean[2]

[1] Program Studi Sarjana Sistem Informasi, Fakultas Informatika dan Teknik Elektro, Institut Teknologi Del, Toba, Indonesia
[2] Program Studi Sarjana Teknik Informatika, Sekolah Tinggi Manajemen Informatika dan Komputer Pelita Nusantara, Medan, Indonesia

[Diserahkan: 10 Agustus 2023, Direvisi: 7 November 2023, Diterima: 13 Desember 2023]
Penulis Korespondensi: Rosni Lumbantoruan (email: rosni@del.ac.id)

INTISARI — Kemajuan teknologi komunikasi telah mempercepat penyebaran informasi. Namun, timbul kekhawatiran tentang penyebaran informasi palsu. Laporan terbaru dari Kementerian Komunikasi dan Informatika Republik Indonesia, yang mengungkapkan bahwa sekitar 800.000 situs web terlibat dalam penyebaran informasi palsu, menggarisbawahi keseriusan dalam masalah ini. Untuk mengatasi masalah ini, para peneliti berfokus pada pengembanganan teknik untuk mendeteksi dan memerangi berita palsu. Penelitian ini berfokus pada penggunaan IndoBERT-base-p1 untuk mendeteksi berita palsu dan bertujuan untuk meningkatkan kinerjanya melalui tiga metode dengan menyetel nilai *hyperparameter* dari model tersebut, yaitu: *Bayesian optimization*, *grid search*, dan *random search*. Setelah membandingkan hasil dari ketiga metode *hyperparameter tuning*, *Bayesian optimization* muncul sebagai pendekatan yang paling efektif, yang mencapai presisi 88,79%, *recall* 94,5%, dan *F1-score* 91,56% untuk label "*fake*". *Bayesian optimization* mengungguli metode *hyperparameter tuning* lainnya dan juga model yang menggunakan nilai *hyperparameter fine-tuning*. Penelitian ini menekankan pentingnya *hyperparameter tuning* dalam meningkatkan akurasi model dalam mendeteksi berita palsu. Dengan menggunakan *Bayesian optimization* dan mengoptimalkan *hyperparameter* yang ditentukan, model ini menunjukkan kinerja yang unggul dalam mengidentifikasi contoh-contoh berita palsu secara akurat serta memberikan kontribusi yang berharga dalam upaya melawan disinformasi di dunia digital.

KATA KUNCI — Berita Palsu, BERT, IndoBERT, *Hyperparameter Tuning*, Pemrosesan Bahasa Alami.

I. PENDAHULUAN

Internet telah menjadi bagian yang tidak terpisahkan dari kehidupan sebagian besar penduduk dunia. Indonesia, yang merupakan negara ketiga di Asia dengan jumlah pengguna internet terbanyak, yaitu 212,35 juta orang, pada 2021 [1], telah menjadikan internet sebagai sumber informasi yang sangat berguna. Sumber lain menyatakan bahwa teknologi informasi di Indonesia juga telah mengalami kemajuan yang signifikan, dengan jumlah pengguna internet pada 2019 mencapai 132,7 juta atau 51,6% dari total populasi Indonesia [2]. Namun, akselerasi penyebaran informasi sebagai dampak dari kemajuan teknologi komunikasi juga dapat menimbulkan misinformasi atau yang biasa disebut dengan "berita palsu", yang didefinisikan sebagai berita yang sengaja dibuat untuk menipu atau menyesatkan [3]. Serupa dengan bahasa yang menyinggung [4], motif umum untuk menyebarkan berita palsu termasuk menyesatkan pembaca, merusak reputasi, atau menghasilkan sensasi. Seperti negara-negara lain, Indonesia juga berjuang melawan informasi yang menyesatkan ini. Sebuah survei menunjukkan bahwa 38% hingga 61,1% masyarakat pedesaan dan 45,3% hingga 79,6% masyarakat perkotaan menjadi korban berita palsu [5]. Selain itu, Kementerian Komunikasi dan Informatika Republik Indonesia (Kominfo RI) menyatakan di situs resminya bahwa terdapat 800.000 situs penyebar hoaks yang beredar di Indonesia pada tahun 2017 [6]. Statistik ini dengan jelas menunjukkan prevalensi penyebaran berita palsu yang mengkhawatirkan di Indonesia.

Masalah berita palsu dapat diatasi dengan memverifikasi keakuratan berita secara manual melalui sumber-sumber alternatif atau melalui klasifikasi otomatis. Namun, metode-metode ini membutuhkan banyak usaha dan waktu. Sementara itu, model *transformer*, sebuah model bahasa *deep learning* yang didasarkan pada *self-attention*, telah menjadi sangat populer untuk mendeteksi berita palsu secara otomatis [7]. Dalam beberapa tahun terakhir, *transformer* dan berbagai modifikasinya telah mencapai peningkatan kinerja yang substansial dalam berbagai tugas pemrosesan bahasa alami (*natural language processing*, NLP) [7]. Salah satu contoh menonjol dari model representasi bahasa yang sudah terlatih yang telah menghasilkan kinerja luar biasa di berbagai tugas arsitektur khusus adalah *bidirectional encoder representations from transformers* (BERT) [8]. BERT dibuat untuk melatih representasi teks yang mendalam di kedua arah kiri dan kanan [9].

Berita palsu dapat ditemukan di situs berita legal maupun ilegal. Bahkan situs yang sebagian besar membagikan berita yang sah pun tidak terlepas dari penyebaran berita palsu. Oleh karena itu, penelitian ini menggunakan kumpulan informasi yang memiliki lebih banyak artikel berita asli daripada berita palsu. Hal ini membantu model secara akurat menemukan berita palsu dalam kelompok besar berita asli. Pada penelitian ini, diambil *dataset* dari penelitian yang telah dilakukan sebelumnya, yaitu *dataset* TurnBackHoax.ID [10]. Penelitian ini akan membandingkan metode BERT, *convolutional neural network* (CNN), *bidirectional long short-term memory* (BiLSTM), dan gabungan CNN-BiLSTM untuk mendeteksi berita palsu dalam bahasa Indonesia. Penelitian menunjukkan bahwa BERT mengungguli metode lainnya. IndoBERT, sebuah varian yang secara khusus dikembangkan untuk bahasa Indonesia, telah digunakan secara luas untuk berbagai tugas NLP, termasuk klasifikasi teks dan pemodelan bahasa. Namun, penelitian IndoBERT saat ini sebagian besar mengabaikan

manfaat kinerja dari *hyperparameter tuning* model. Maka, meskipun BERT memiliki kinerja yang kuat, penelitian ini terutama berkonsentrasi pada penyetelan nilai *hyperparameter* untuk model guna mengidentifikasi berita palsu.

Meskipun BERT memiliki kinerja yang kuat dalam berbagai tugas NLP, termasuk mendeteksi berita palsu, penelitian dan pengembangan model yang sedang berlangsung secara konsisten berusaha untuk meningkatkan kinerjanya. *Hyperparameter tuning* adalah metode yang efektif untuk mencapai peningkatan tersebut. Penelitian sebelumnya telah menunjukkan bahwa *hyperparameter tuning* dapat secara signifikan meningkatkan kinerja model pembelajaran mesin, termasuk yang digunakan dalam pendeteksian berita palsu [11]–[13]. Peningkatan kinerja deteksi berita palsu melalui *hyperparameter tuning* memungkinkan pengembangan sistem yang lebih andal dan efektif untuk mengidentifikasi penyebaran berita palsu. Pemilihan *hyperparameter* yang optimal menghadirkan tantangan yang harus diatasi untuk mencapai hasil prediksi yang lebih baik [14].

Kontribusi penelitian ini adalah membandingkan tiga pendekatan yang paling umum digunakan untuk menyesuaikan nilai *hyperparameter*, yaitu: *Bayesian optimization*, *random search*, dan *grid search*. Setiap teknik dianalisis berdasarkan kinerjanya dan pemenuhan kriteria presisi (*precision*), *recall*, dan *F*1-*score*. Metode dengan kinerja tertinggi untuk semua metrik evaluasi akan dianggap sebagai metode *tuning* untuk studi kasus identifikasi berita palsu dalam bahasa Indonesia.

II. PENDEKATAN TERKAIT KLASIFIKASI TEKS

Bagian ini berfokus pada kemajuan terbaru dalam metode klasifikasi, yang mencakup BERT sebagai pendekatan umum dan model khusus yang dirancang untuk bahasa Indonesia, yang dikenal sebagai IndoBERT. Selain itu, dieksplorasi juga teknik-teknik populer untuk *hyperparameter tuning* dalam konteks tugas klasifikasi, yaitu *Bayesian optimization*, *random search*, dan *grid search.*

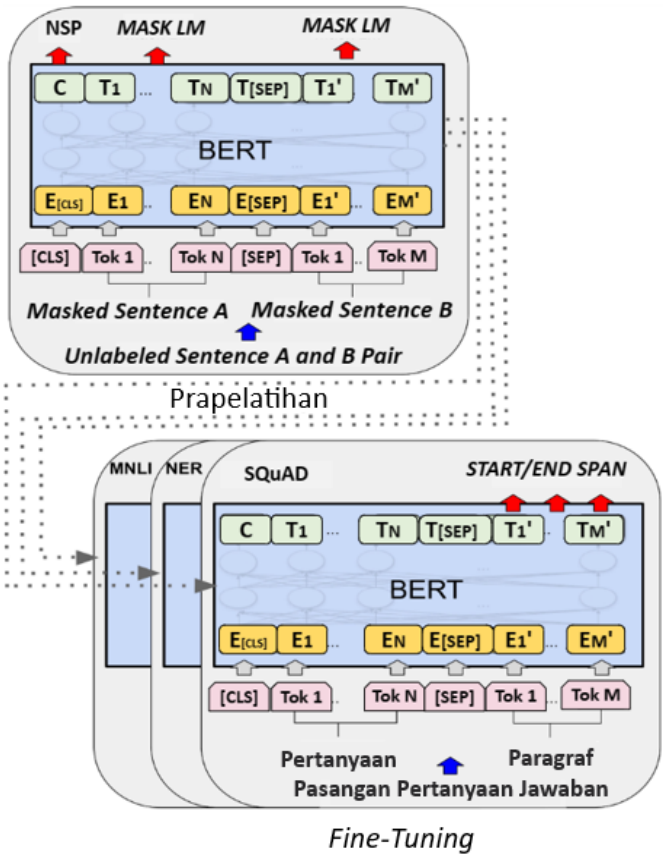
A. BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMERS (BERT)

BERT adalah model representasi bahasa yang dirancang sebagai transformator dua arah yang menangkap informasi dari teks dengan menggabungkan representasi dari token konteks kiri dan kanan di semua lapisan. BERT memahami hubungan kata secara dua arah dan menghasilkan vektor representasi untuk setiap kata berdasarkan hubungannya dalam sebuah kalimat [9].

BERT menggunakan mekanisme *self-attention*, yaitu menggabungkan beberapa vektor kata sebagai masukan dan menyertakan *cross-attention* di kedua arah antara dua kalimat [9]. BERT digunakan sebagai *sentence encoder* yang secara akurat merepresentasikan konteks dalam sebuah kalimat. BERT unggul dalam *transfer learning*, karena keluaran model BERT menyediakan model yang sudah terlatih yang dapat diadopsi untuk tugas-tugas klasifikasi teks, seperti yang ditunjukkan pada Gambar 1 [9]. Para peneliti menggunakan BERT sebagai *sentence encoder*, yang secara akurat mendapatkan representasi kontekstual dari sebuah kalimat [15]. Ada dua jenis model BERT yang digunakan untuk konteks tertentu, yang dijelaskan sebagai berikut [16].

1) BERT BASE

BERT base terdiri atas lapisan *embedding* dan 12 lapisan *encoder* dengan 12 *attention heads*. BERT base memiliki 110 juta parameter dan ukuran tersembunyi sebesar 768. Model ini



Gambar 1. Prapelatihan dan *fine-tuning* BERT.

berukuran lebih kecil dan terjangkau secara komputasi, tetapi mungkin tidak cocok untuk operasi penambangan teks yang kompleks.

2) BERT LARGE

BERT large adalah model yang lebih besar dengan kebutuhan komputasi yang lebih banyak. Model ini memiliki 24 lapisan, 16 *attention heads*, 340 juta parameter, dan ukuran tersembunyi sebesar 1.024. Model ini dapat menangani data teks yang lebih besar dan memberikan hasil yang lebih baik.

Karena BERT adalah model yang telah terlatih dengan membutuhkan masukan data dalam format tertentu, maka diperlukan hal-hal berikut.

- Token khusus [SEP] digunakan untuk menunjukkan akhir kalimat atau pemisahan antara dua kalimat.
- Token khusus [CLS] digunakan untuk klasifikasi yang terletak di awal teks
- Token yang sesuai dengan kosakata tetap yang digunakan dalam BERT.
- *Token ID* untuk token yang berasal dari *tokenizer* BERT.
- *Mask ID* yang terdiri atas nilai biner: 1 berarti model harus memperhatikan token; sedangkan 0 menunjukkan bahwa model tidak harus memperhatikan elemen *padding*.
- *Segment ID* digunakan membedakan berbagai kalimat.
- *Positional embeddings* digunakan untuk menunjukkan posisi token dalam urutan.

B. INDOBERT

IndoBERT adalah model berbasis *transformer* yang terinspirasi oleh BERT dan dilatih dalam korpus besar bahasa Indonesia. Model ini menggabungkan kosakata substansial lebih dari 220 juta kata yang bersumber dari referensi bahasa

Indonesia yang dapat diandalkan, seperti surat kabar *online* dan Korpus Web Indonesia. IndoBERT adalah *pre-trained model* yang dikembangkan dengan 2,4 juta langkah atau 180 *epoch*, sehingga menghasilkan kinerja yang kuat pada berbagai tugas NLP. *Pre-trained model* dilatih menggunakan *masked language modeling* (MLM) dan *next sentence prediction* (NSP). Model ini menggunakan arsitektur *transformer* dengan 12 lapisan dan 768 unit pemrosesan. IndoBERT dilatih dengan kosakata WordPiece bahasa Indonesia yang terdiri atas 31.923 token. Beberapa varian IndoBERT yang telah dikembangkan termasuk IndoBERT-base, IndoBERT-large, dan IndoBERT-lite. Varian dasar berisi 12 lapisan dan sekitar 125 juta parameter, sedangkan varian besar memiliki 24 lapisan dan sekitar 340 juta parameter [17].

C. INDOBERT-BASE

IndoBERT-base adalah model dasar dari IndoBERT, yang telah dilatih dengan korpus 5,5 miliar kata yang mencakup beberapa bentuk teks bahasa Indonesia. Model ini dapat digunakan untuk berbagai tugas NLP. Model ini terdiri atas 12 lapisan *transformator* dengan 12 *heads* per lapisan dan 110 juta parameter. Berikut ini jenis-jenis IndoBERT-base [17], [18].

1) INDOBERT-BASE-P1

Dengan menggunakan teknik *transfer learning*, model ini dilatih pada kumpulan data teks dalam bahasa Indonesia yang sangat besar dari berbagai sumber, termasuk artikel berita, Wikipedia, dan media sosial. Model ini juga dapat digunakan untuk melakukan berbagai tugas NLP.

2) INDOBERT-BASE-P2

Dibandingkan dengan IndoBERT-base-p1, model ini telah dilatih pada *dataset* yang lebih kompleks dan dapat menghasilkan keluaran yang lebih akurat. Hasilnya adalah model ini sesuai untuk tugas-tugas yang membutuhkan pemahaman bahasa yang lebih dalam, seperti klasifikasi dokumen dan analisis sentimen.

D. HYPERPARAMETER TUNING

Hyperparameter tuning umumnya melibatkan teknik-teknik seperti *grid search*, *random search*, dan *Bayesian optimization* [19].

1) GRID SEARCH

Grid search adalah metode tradisional untuk mengoptimalkan *hyperparameter* dengan mencari secara mendalam subset tertentu dari ruang *hyperparameter* algoritma pelatihan. Ruang parameter algoritma pembelajaran mesin dapat menyertakan nilai nyata atau tidak terbatas untuk parameter tertentu, sehingga perlu menentukan batasan untuk menerapkan *grid search*. *Hyperparameter* ditentukan menggunakan nilai minimum (batas bawah) dan maksimum (batas atas). *Grid search* merupakan pendekatan waktu eksponensial yang sulit diterapkan di area berdimensi tinggi. Namun, *hyperparameter* biasanya beroperasi secara independen satu sama lain, sehingga dapat dikelola dengan melakukan paralelisasi [19].

2) RANDOM SEARCH

Random search adalah metode untuk mengidentifikasi kumpulan konfigurasi *hyperparameter* yang optimal dengan mencoba kombinasi *hyperparameter* yang berbeda secara acak. Metode ini bekerja dengan mendefinisikan ruang pencarian *hyperparameter* dan memilih secara acak untuk menghasilkan kumpulan kombinasi *hyperparameter*. Tidak ada jaminan bahwa pendekatan ini akan menggabungkan nilai *hyperparameter* yang ideal untuk dicoba karena keacakannya. *Random search* bersifat efisien dan dapat menangani data berdimensi besar dengan baik. Alih-alih 100.000 sampel, hanya 1.000 sampel acak dari kumpulan *hyperparameter* yang akan diperiksa.

Keuntungan dari *random search* antara lain membutuhkan lebih sedikit waktu dan komputasi untuk mendapatkan hasil dibandingkan dengan metode pengoptimalan lainnya seperti *grid search*. Meskipun tidak menemukan set *hyperparameter* terbaik, *random search* menghasilkan model yang dapat mendekati kinerja model ideal. Di sisi lain, *random search* memiliki kekurangan seperti membutuhkan lebih banyak waktu karena pencarian acak di setiap iterasi, yang dapat memakan waktu dan membutuhkan banyak iterasi untuk menemukan kombinasi *hyperparameter* terbaik.

3) BAYESIAN OPTIMIZATION

Bayesian optimization adalah algoritma pencarian berdasarkan informasi, yang artinya temuan iterasi sebelumnya digunakan sebagai masukan pembelajaran untuk iterasi berikutnya dan hasil dari satu iterasi akan memengaruhi iterasi berikutnya. Model pengganti probabilistik yang menangkap opini tentang perilaku fungsi objektif yang tidak diketahui dan fungsi akuisisi yang mendefinisikan tingkat keoptimalan urutan kueri adalah komponen penting dari *Bayesian optimization*. Dalam praktiknya, fungsi akuisisi sering kali berbentuk *regret*, baik sederhana maupun kumulatif [19].

Optimasi dalam *deep learning* mengacu pada proses menemukan nilai parameter yang efisien untuk memaksimalkan keluaran dari masukan numerik yang diberikan. Pemilihan *hyperparameter* yang tepat memainkan peran penting dalam melatih BERT dan secara signifikan berdampak pada kinerja model yang dibangun. *Hyperparameter* yang ditentukan untuk model yang dibangun terdiri atas laju pembelajaran, ukuran *batch*, dan jumlah *epoch*. *Hyperparameter* ini memainkan peran penting dalam mengoptimalkan kinerja model BERT selama proses pelatihan.

E. EVALUASI MODEL

Mengingat tugas yang diselesaikan pada penelitian ini adalah klasifikasi berita ke dalam dua kelas, yaitu "*fake*" atau "tidak *fake*", maka digunakan metrik evaluasi untuk klasifikasi yang umum digunakan, yaitu akurasi (*accuracy*), presisi, dan *recall*. Selain itu, *F*1-*score* juga digunakan untuk menyeimbangkan kebutuhan akan perlunya presisi dan *recall*. Akurasi dihitung menggunakan (1).

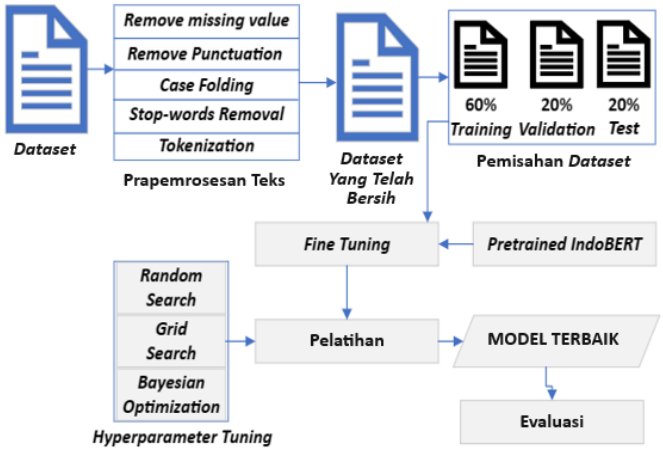
$$Akurasi = \frac{(TP+TN)}{(TP+TN+FP+FN)}. \quad (1)$$

Akurasi dihitung dengan menghitung rasio prediksi benar positif, termasuk prediksi benar *true positive* (TP) dan prediksi benar *true negative* (TN), terhadap keseluruhan hasil positif yang diperiksa, termasuk prediksi benar sebelumnya dan prediksi *false positive* (FP) serta *false negative* (FN). Presisi dihitung menggunakan (2).

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

Recall melakukan perhitungan untuk mendapatkan semua dokumen yang dianggap relevan oleh sistem. *Recall* dihitung menggunakan (3).

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



Gambar 2. Metodologi penelitian.

*F*1-*score* dievaluasi dengan kisaran 0 sebagai nilai terburuk dan 1 sebagai nilai terbaik. *F*1-*score* dihitung dengan (4).

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

III. METODOLOGI

Pada bagian ini, akan diberikan penjelasan rinci tentang data yang digunakan dalam penelitian ini, prapemrosesan, dan model BERT untuk klasifikasi, seperti yang diilustrasikan pada Gambar 2. Prosesnya dimulai dengan pengumpulan *dataset*, diikuti dengan prapemrosesan data dan pemisahan *dataset*. Selanjutnya adalah tahap *pre-trained IndoBERT* dan diakhiri dengan tahap *hyperparameter tuning*.

A. HIMPUNAN DATA

Dalam penelitian ini, digunakan *dataset* dari penelitian sebelumnya, yaitu TurnBackHoax.ID. *Dataset* berisi data yang terdiri atas 1.116 baris dalam bahasa Indonesia. Jumlah data berita "*fake*" sebanyak 433 data dan jumlah berita "real" sebanyak 683 data. Terdapat tiga atribut pada *dataset*, yaitu "label" yang terdiri atas dua label: nol (0) untuk kategori berita "real" dan satu (1) untuk kategori berita "*fake*"; "*headline*" berisi judul berita; dan "*body*" berisi teks isi berita. Pada pembuatan model, *dataset* perlu dipisah untuk mengevaluasi kinerja model pembelajaran mesin. Setelah itu, kinerja model dievaluasi dengan terlebih dahulu membagi *dataset* menjadi data pelatihan, data validasi, dan data pengujian, masing-masing sebesar 60%, 20%, dan 20%.

B. PRAPEMROSESAN DATA

Tahap prapemrosesan teks adalah proses mengubah format data yang tidak terstruktur menjadi data terstruktur untuk menyajikan data dalam format kata yang jelas. Tahap prapemrosesan data akan menghilangkan nilai yang hilang, yang berarti bahwa setiap data kosong atau *null* dalam kumpulan data akan dihapus pada tahap ini. Kemudian, tanda baca dihapus untuk menghilangkan properti seperti "@% -"" " agar tidak mengganggu proses penghitungan dalam aplikasi algoritma. Selanjutnya, dilakukan pengubahan huruf kapital menjadi huruf kecil (*lowercasing*) pada semua kata untuk memastikan bahwa kata-kata seperti "yANG", "yang", dan "Yang" memiliki arti yang sama.

Tahap *stopword removal* adalah proses penghapusan kata-kata yang masuk daftar kata tidak penting.. Korpus dari Sastrawi digunakan dalam prosedur *stopword removal* ini untuk menghasilkan daftar *stopword* dalam bahasa Indonesia. Kata-kata seperti "yang", "di", "ke", "adalah", "dan", dan "dari" merupakan contoh *stopword* dalam Sastrawi. *Dataset* ini memiliki total 1.676.646 kata sebelum penghapusan *stopword*. Data berkurang menjadi 1.672.798 kata setelah *stopword* dihapus, dengan penurunan sebesar 0,23%. Hal ini dilakukan untuk mengurangi kata-kata yang tidak penting dari pernyataan sambil menjaga agar maknanya tetap utuh. Tahap terakhir dalam prapemrosesan adalah tokenisasi, yang membagi teks menjadi kata-kata individual. Tujuan tahap ini adalah menemukan token untuk setiap kata dalam sebuah kalimat.

C. FINE-TUNING

Proses *fine-tuning* dimulai dengan menginisialisasi *pre-trained model* IndoBERT. Variasi model IndoBERT yang akan digunakan dalam penelitian ini adalah IndoBERT-base-p1. Pemilihan model didasarkan pada keterbatasan ukuran *dataset* yang digunakan dalam penelitian. IndoBERT-base memiliki lebih sedikit lapisan dan parameter yang membuatnya lebih cepat dalam melatih model [17].

Pada proses *fine-tuning*, arsitektur model dimodifikasi sesuai dengan tugas khusus yang harus diselesaikan. Dalam penelitian ini, tugas khusus tersebut adalah klasifikasi teks. Pada proses *fine-tuning* akan diberikan urutan teks dari *dataset* sebagai masukan ke model IndoBERT. Panjang urutan maksimum yang digunakan adalah 512 [9] . Jika sebuah urutan teks melebihi panjang maksimum 512 token, teks perlu dipotong atau dibagi menjadi bagian yang lebih kecil agar sesuai dengan batasan.

Dalam proses *fine-tuning* ini, ditetapkan nilai *hyperparameter* dari penelitian sebelumnya untuk ketiga *hyperparameter*: laju pembelajaran, ukuran *batch*, dan jumlah *epoch*. Setelah melalui proses *fine-tuning* yang cukup, model IndoBERT akan memiliki pengetahuan yang disesuaikan dengan tugas spesifik yang diberikan. Representasi teks yang dihasilkan oleh model yang telah disesuaikan dapat digunakan untuk melakukan inferensi pada data baru.

D. HYPERPARAMETER TUNING

Pada penelitian terdahulu, model yang dihasilkan tidak mempertimbangkan perlunya melakukan *hyperparameter tuning*, sehingga nilai parameter yang digunakan bersifat statis [18]. Dalam penelitian ini, model pada penelitian sebelumnya [18] dibandingkan dengan model setelah *hyperparameter tuning*. *Hyperparameter tuning* BERT dilakukan guna menemukan kombinasi *hyperparameter* yang optimal untuk meningkatkan kinerja model. Untuk perbandingan model sebelum dan sesudah *hyperparameter tuning*, dipilih metode yang paling sederhana dan paling umum digunakan, yaitu *grid search*, *random search*, dan *Bayesian optimization* [18]. *Grid search* dipilih karena mudah diimplementasikan dan cocok untuk semua jenis model, *random search* dipilih karena lebih efisien untuk ruang pencarian besar dan banyak *hyperparameter*, sedangkan *Bayesian optimization* dipilih karena memberikan optimasi global untuk fungsi kotak hitam, sehingga mengurangi evaluasi kesalahan validasi yang diperlukan. *Framework* yang digunakan untuk menentukan nilai *hyperparameter* secara otomatis adalah Optuna [20]. Optuna dibangun secara dinamis, sehingga lebih berpeluang mendapatkan parameter terbaik yang mungkin tidak didapatkan dengan metode *hyperparameter* lainnya [21].

Hyperparameter yang digunakan dalam penelitian ini adalah laju pembelajaran, ukuran *batch*, dan jumlah *epoch*. Ketiga parameter tersebut merupakan *hyperparameter* terpenting dalam BERT [17]. Laju pembelajaran berfungsi untuk mengontrol kecepatan model belajar selama proses

pelatihan. Jika laju pembelajaran terlalu tinggi, model mungkin gagal konvergen. Sebaliknya, jika laju pembelajaran terlalu rendah, model mungkin membutuhkan waktu terlalu lama untuk konvergen. Ukuran *batch* berfungsi untuk menentukan jumlah sampel yang digunakan pada setiap iterasi pelatihan. Ukuran *batch* yang lebih besar dapat menghasilkan waktu pelatihan yang lebih cepat, tetapi juga dapat menyebabkan kemampuan dan akurasi generalisasi menjadi buruk. Lalu, jumlah *epoch* digunakan untuk menentukan banyaknya pelatihan yang dilakukan terhadap model di seluruh *dataset*.

E. EVALUASI

Pada penelitian ini, evaluasi dijadikan sebagai dasar atau tolok ukur untuk membandingkan kinerja ketiga model setelah menerapkan *hyperparameter tuning*. Perhitungan dan metrik evaluasi yang digunakan adalah akurasi, presisi, *recall*, dan *F*1-*score*. Dalam penelitian ini, nilai akurasi diperlukan untuk menilai ketepatan sistem dalam mengklasifikasikan data secara akurat. Namun, akurasi saja tidak cukup untuk menunjukkan kemampuan model. Jika semua kalimat yang diklasifikasikan berada dalam kategori negatif, pengklasifikasi yang selalu memprediksi kalimat sebagai negatif akan memiliki akurasi yang sangat tinggi. Oleh karena itu, diperlukan pengukuran lain seperti presisi, *recall*, dan *F*1-*score*. Nilai presisi menunjukkan model yang dibangun memiliki tingkat kebenaran yang tinggi atau rendah, sedangkan nilai *recall* diperlukan untuk menilai sensitivitas model. Model dengan sensitivitas tinggi menunjukkan bahwa prediksi model tersebut sangat relevan, begitu pula sebaliknya. *F*1-*score* digunakan untuk menentukan nilai komparatif rata-rata terbobot dari presisi dan *recall*, yang mewakili keseluruhan kinerja sistem.

IV. EKSPERIMEN

Pada subbagian ini, kinerja ketiga metode *hyperparameter tuning* dalam menentukan nilai *hyperparameter* model yang optimal untuk laju pembelajaran, ukuran *batch*, dan jumlah *epoch* dibandingkan satu dengan yang lain. Eksperimen ini menggunakan IndoBERT-base-p1 yang telah dilatih sebelumnya, yang diperoleh dari repositori Indo Benchmark pada Hugging Face [19], dengan Adam sebagai pengoptimal. Perangkat keras yang digunakan untuk mengembangkan dan menjalankan eksperimen ini adalah komputer dengan prosesor Intel(R) Xeon(R) CPU @ 2,20 GHz dan RAM dengan kapasitas 8 GB.

A. FINE-TUNING

Proses selanjutnya adalah *fine-tuning*. Model IndoBERT yang telah diinisialisasi sebelumnya [18] dilatih ulang dengan data tertentu. Pada penelitian ini, model tersebut [18] disebut sebagai *model with fine-tuning*. Nilai-nilai *hyperparameter* untuk laju pembelajaran, ukuran *batch*, dan jumlah *epoch* yang digunakan dalam proses *fine-tuning* disajikan pada Tabel I.

Hyperparameter dilatih bersama dengan model BERT, lalu akurasi validasi akan dihitung. Selanjutnya, model dievaluasi dan dari hasil evaluasi tersebut akan ditentukan kombinasi *hyperparameter* dengan kinerja terbaik. Hasilnya mencakup *hyperparameter* terbaik atau model terbaik, yang terdiri atas laju pembelajaran, ukuran *batch*, jumlah *epoch*, dan *validation loss* terbaik.

B. HYPERPARAMETER TUNING

Setelah tahap *fine-tuning*, dilakukan *hyperparameter tuning*, yang dimulai dengan memilih rentang nilai untuk setiap *hyperparameter* berdasarkan penelitian sebelumnya. Tiga teknik *hyperparameter tuning* akan dibandingkan, yaitu *grid search*, *random search*, dan *Bayesian optimization*. Percobaan dilakukan sebanyak tiga kali untuk menguji setiap metode. Rentang *hyperparameter* yang digunakan pada ketiga jenis *hyperparameter tuning* tersebut dapat dilihat pada Tabel II.

TABEL I
NILAI *HYPERPARAMETER* DARI *FINE-TUNING*

Hyperparameter	**Rentang Nilai**
Laju pembelajaran	2×10^{-5}
Ukuran *batch*	16
Jumlah *epoch*	10

TABEL II
RENTANG NILAI *HYPERPARAMETER TUNING*

Hyperparameter	**Rentang Nilai**
Laju pembelajaran	2×10^{-5}, 3×10^{-5}, 5×10^{-5}
Ukuran *batch*	16, 32
Jumlah *epoch*	10

Dalam penelitian ini, digunakan Optuna sebagai *framework* untuk *hyperparameter tuning*, yang mencoba 36 kombinasi *hyperparameter* yang berbeda. Dengan memperhatikan semua kemungkinan *hyperparameter* yang diberikan, tampak bahwa jumlah keseluruhan kombinasi adalah $3 \times 6 \times 2 = 36$ (laju pembelajaran × jumlah *epoch* × ukuran *batch*). Optuna digunakan dengan arah "meminimalkan", yang bertujuan untuk menemukan *hyperparameter* yang memberikan nilai terendah pada fungsi objektif. Meskipun *local optima* mengacu pada titik terendah dalam ruang *hyperparameter* tertentu dan *global optima* adalah nilai terendah di seluruh ruang, Optuna menggunakan berbagai teknik agar tidak terjebak pada *local optima* dan mendekati *global optima*. Jadi, tujuan utama dalam "meminimalkan" dengan Optuna adalah mendekati atau mencapai *global optima*. Untuk mencegah terjadinya *overfitting*, penghentian awal akan ditambahkan. Teknik ini melibatkan pemantauan *validation loss* selama pelatihan dan akan menghentikan proses pelatihan ketika *validation loss* tidak membaik. Dari hasil pelatihan, akan didapatkan model terbaik dari kombinasi nilai *hyperparameter* terbaik dari rentang nilai yang ditentukan.

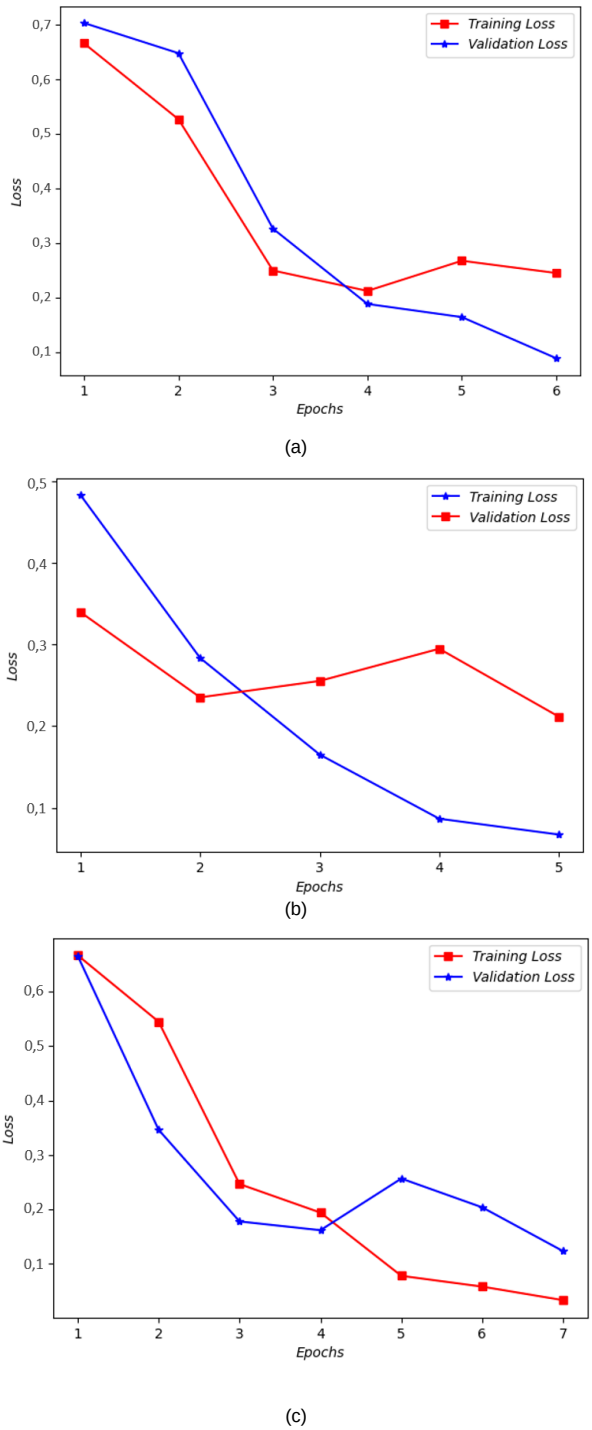
V. HASIL DAN PEMBAHASAN

Pada bagian ini, hasil ketiga metode *hyperparameter tuning* dibandingkan. Lalu, kinerjanya dibandingkan dengan kinerja sebelum diatur, dalam hal presisi, *recall*, *F*1-*score*, dan akurasi.

A. PERBANDINGAN PELATIHAN METODE HYPERPARAMETER TUNING

Gambar 3 menunjukkan kinerja model selama pelatihan dalam hal *training loss* dan *validation loss*, dengan sumbu x menunjukkan jumlah *epoch* model pelatihan dan sumbu y menunjukkan tingkat *loss* yang terkait pada setiap *epoch*. *Training loss* pada awalnya memiliki nilai yang tinggi. Namun, seiring bertambahnya jumlah *epoch*, baik *training loss* maupun *validation loss* mulai berkurang. Di sini juga digunakan penghentian awal, yaitu pelatihan dihentikan ketika terdeteksi adanya *overfitting*.

Gambar 3(a) menunjukkan bahwa untuk pendekatan *grid search*, pelatihan berhenti pada *epoch* ke-6. Model masih mengalami sedikit *overfitting*, yaitu jika diamati grafik *validation loss* pada awalnya memiliki nilai *loss* yang tinggi, kemudian menurun seiring dengan bertambahnya jumlah *epoch*. Namun, setelah beberapa *epoch*, nilai *loss* mulai naik kembali. Hal ini mengindikasikan bahwa model mendapatkan "*fit*" yang lebih baik pada data pelatihan seiring dengan



Gambar 3. Perbandingan *training loss* dan *validation loss* untuk beberapa pendekatan, (a) *grid search*, (b) *random search*, (c) *Bayesian optimization*.

bertambahnya jumlah *epoch*, tetapi tidak berhasil menggeneralisasi hal yang telah dipelajari ke data validasi.

Gambar 3(b) menunjukkan bahwa untuk pendekatan *random search*, pelatihan berhenti pada *epoch* ke-6. Model masih mengalami sedikit *overfitting*. Jika diamati, grafik *validation loss* awalnya memiliki nilai *loss* yang tinggi dan menurun pada beberapa *epoch* pertama. Setelah beberapa *epoch*, nilai *loss* pada data validasi mulai meningkat kembali, sedangkan nilai *loss* pada data pelatihan terus menurun. Hal ini menunjukkan bahwa model mengalami *overfitting*, sehingga ketika diterapkan pada data validasi, kinerjanya menurun. Dari segi *loss*-nya, kinerja *random search* ini merupakan yang paling buruk di antara ketiga teknik yang ada, yang menunjukkan model yang tetap *overfitting* karena model tidak

TABEL III
NILAI *HYPERPARAMETER* YANG OPTIMAL

Metode	Laju Pembelajaran	Jumlah *Epoch*	Ukuran *Batch*
Grid search	5×10^{-5}	8	32
Random search	5×10^{-5}	9	32
Bayesian optimization	2×10^{-5}	8	16

TABEL IV
PERBANDINGAN KINERJA BEBERAPA PENDEKATAN DALAM PRESISI

Metode	Presisi			
	Label "Real"	*Pening-katan*	*Label "Fake"*	*Pening-katan*
Grid Search	0,9462	2,50%	0,8661	1,89%
Random Search	0,9541	1,77%	0,8462	3,53%
Bayesian Optimization	0,9726	-	0,8879	-

TABEL V
PERBANDINGAN KINERJA BEBERAPA PENDEKATAN DALAM *RECALL*

Metode	*Recall*			
	Label "Real"	*Pening-katan*	*Label "Fake"*	*Pening-katan*
Grid search	0,9336	0,83%	0,8899	4,90%
Random search	0,9204	2,03%	0,9083	3,33%
Bayesian optimization	0,9425	-	0,9450	-

TABEL VI
PERBANDINGAN KINERJA BEBERAPA PENDEKATAN DALAM *F1-SCORE*

Metode	*F1-Score*			
	Label "Real"	*Pening-katan*	*Label "Fake"*	*Pening-katan*
Grid search	0,9399	1,64%	0,8899	2,29%
Random search	0,9369	1,91%	0,8761	3,46%
Bayesian optimization	0,9573	-	0,9156	-

dapat menggeneralisasi pola dari data pelatihan ke data baru dengan baik.

Gambar 3(c) menunjukkan bahwa untuk *Bayesian optimization*, pelatihan awal berhenti pada *epoch* ke-7. Model ini tidak menunjukkan adanya *overfitting*. Hal ini dibuktikan dengan grafik *validation loss*, yang pada awalnya memiliki nilai *loss* yang tinggi, lalu menurun seiring dengan bertambahnya *epoch*. Hal ini berarti kinerja model pada data pelatihan dan data validasi berkorelasi, mengindikasikan bahwa model dapat mempelajari pola pada data pelatihan dan menggeneralisasi dengan baik data validasi. Tampak pula bahwa *Bayesian optimization* memiliki kinerja yang lebih baik dan lebih konsisten daripada dua metode lainnya.

B. PERBANDINGAN KINERJA METODE HYPERPARAMETER TUNING

Sebelumnya, model telah dilatih dan nilai *hyperparameter* yang optimal untuk ketiga teknik yang digunakan telah ditemukan. Nilai *hyperparameter* optimal yang dihasilkan oleh masing-masing metode disajikan pada Tabel III.

Dengan nilai *hyperparameter* yang dipilih pada Tabel III, selanjutnya kinerja ketiga teknik *hyperparameter tuning* ini dibandingkan, dalam hal presisi, *recall*, dan *F1-Score*, seperti ditunjukkan pada Tabel IV sampai Tabel VI. Tabel IV menggambarkan perbandingan kinerja ketiga teknik ini dalam hal presisi. Seperti dapat dilihat, *Bayesian optimization* mengungguli dua teknik lainnya dalam mengidentifikasi label asli dan palsu pada sebuah berita.

TABEL VII
PERBANDINGAN KINERJA *FINE-TUNING* DENGAN *HYPERPARAMETER TUNING* DALAM PRESISI

Metode	Presisi	Peningkatan
Model dengan *fine-tuning*	0,8632	-
Model dengan *grid search*	0,8661	0,33%
Model dengan *random search*	0,8462	-2,01%
Model dengan *Bayesian optimization*	0,8879	2,78%

TABEL VIII
PERBANDINGAN KINERJA *FINE-TUNING* DENGAN *HYPERPARAMETER TUNING* DALAM *RECALL*

Metode	*Recall*	Peningkatan
Model dengan *fine-tuning*	0,9266	-
Model dengan *grid search*	0,8899	-4,12%
Model dengan *random search*	0,9083	1,95%
Model dengan *Bayesian optimization*	0,9450	-

TABEL IX
PERBANDINGAN KINERJA *FINE-TUNING* DENGAN *HYPERPARAMETER TUNING* DALAM *F*1-*SCORE*

Metode	*F*1-*Score*	Peningkatan
Model dengan *fine-tuning*	0,8938	-
Model dengan *grid search*	0,8899	-0,44%
Model dengan *random search*	0,8761	-2,02%
Model dengan *Bayesian optimization*	0,9156	2,38%

TABEL X
PERBANDINGAN KINERJA *FINE-TUNING* DENGAN *HYPERPARAMETER TUNING* DALAM AKURASI

Metode	Akurasi	Peningkatan
Model dengan *fine-tuning*	0,9313	-
Model dengan *grid search*	0,9194	-1,29%
Model dengan *random search*	0,9164	-1,63%
Model dengan *Bayesian optimization*	0,9432	1,26%

Tabel V menggambarkan perbandingan kinerja metode-metode tersebut dalam hal *recall*. Di sini, *Bayesian optimization* mengungguli *random search* dan *grid search* dalam mengklasifikasikan data berita palsu.

Sementara itu, pada Tabel VI dapat dilihat perbandingan kinerja ketiga metode ini dalam hal *F*1-*score*. Tabel tersebut menunjukkan bahwa *Bayesian optimization* secara konstan mengungguli dua metode lainnya dalam keseluruhan metrik evaluasi dalam mengklasifikasikan data berita palsu, diikuti oleh *grid search* dan yang terakhir adalah *random search*. Meskipun *random search* berkinerja baik pada data pelatihan, model ini cenderung memiliki skor evaluasi yang rendah pada data yang tidak teramati.

C. PERBANDINGAN KINERJA MODEL SEBELUM DAN SESUDAH HYPERPARAMETER TUNING

Pada bagian ini, dievaluasi kinerja IndoBERT-base-p1 sebelum dan sesudah penerapan *hyperparameter tuning* dalam mengklasifikasikan berita palsu, khususnya dalam mengklasifikasikan berita dengan label "*fake*" dengan benar. Kinerja IndoBERT-base-p1 setelah proses *tuning* dengan *grid search*, *random search*, dan *Bayesian optimization* menggunakan nilai *hyperparameter* optimal pada Tabel III dibandingkan menggunakan nilai *hyperparameter fine-tuning* pada Tabel II. Tabel VII sampai Tabel X masing-masing mengilustrasikan evaluasi model sebelum dan sesudah *hyperparameter tuning* dalam hal presisi, *recall*, *F*1-*score*, dan akurasi.

TABEL XI
PERBANDINGAN WAKTU PROSES ANTARA MODEL DENGAN *FINE-TUNING* DAN MODEL DENGAN *HYPERPARAMETER TUNING*

Metode	Waktu (menit)	Penambahan Waktu
Model dengan *fine-tuning*	70	-
Model dengan *grid search*	88	25,71%
Model dengan *random search*	77	10,00%
Model dengan *Bayesian optimization*	84	20,00%

Dari tabel hasil dapat dilihat bahwa model setelah *hyperparameter tuning*, khususnya dengan teknik *Bayesian optimization*, memiliki kinerja yang melampaui model dengan *fine-tuning* untuk semua metrik evaluasi yang dinilai. *Bayesian optimization* dianggap lebih baik daripada *fine-tuning* saja untuk IndoBERT-base-p1 karena secara efektif mencari *hyperparameter* yang optimal untuk model. *Fine-tuning* melibatkan penyesuaian bobot model yang telah dilatih sebelumnya pada tugas-tugas tertentu, tetapi sering kali membutuhkan penyempurnaan *hyperparameter* yang cermat untuk mencapai kinerja terbaik pada tugas target, seperti deteksi berita palsu dalam kasus ini.

Dengan demikian, terbukti bahwa *hyperparameter tuning* pada penelitian ini dapat meningkatkan kinerja model dalam mengklasifikasikan berita palsu. Selain itu, optimasi *hyperparameter* dapat melibatkan interaksi nonlinear antarparameter, sehingga menantang metode tradisional seperti *grid search* atau *random search* untuk mengeksplorasi ruang secara efektif. *Bayesian optimization*, di sisi lain, menggunakan pemodelan probabilistik dan inferensi Bayesian untuk menangkap hubungan nonlinear ini dengan lebih baik, sehingga lebih cocok untuk tugas-tugas *hyperparameter* tuning yang kompleks. Metode *grid search* menjadi yang terbaik kedua untuk presisi dan memiliki kinerja yang sedikit berbeda dalam hal *recall* dan *F*1-*score* dibandingkan dengan model dengan *fine-tuning*. Namun, dalam hal akurasi, *fine-tuning* merupakan terbaik kedua.

D. PERBANDINGAN MODEL BERDASARKAN WAKTU SEBELUM DAN SESUDAH HYPERPARAMETER TUNING

Tabel XI menampilkan perbandingan efisiensi dari IndoBERT-base-p1 dengan metode *fine-tuning* dan setelah melakukan *hyperparameter tuning* menggunakan *grid search*, *random search*, dan *Bayesian optimization* berdasarkan waktu yang dibutuhkan untuk memperoleh nilai *hyperparameter* yang optimal hingga menyelesaikan klasifikasi. Terlihat jelas bahwa apabila dibandingkan dengan model *fine-tuning*, model dengan *hyperparameter tuning* membutuhkan waktu tambahan, yang digunakan untuk menemukan nilai *hyperparameter* yang optimal. Namun, tambahan waktu ini hanya dibutuhkan saat pertama sekali pelatihan dilakukan. Selanjutnya, model dapat digunakan untuk tugas klasifikasi. Dengan demikian, pertambahan waktu ini dapat ditoleransi dengan kenaikan kinerja dari model dengan *hyperparameter tuning*.

VI. KESIMPULAN DAN SARAN

Berdasarkan hasil dan pembahasan, dapat disimpulkan bahwa metode *hyperparameter tuning Bayesian optimization* menunjukkan kinerja yang unggul pada model IndoBERT-base-p1 dengan menggunakan *dataset* TurnBackHoax.ID dibandingkan dengan metode lain, termasuk model tanpa *hyperparameter tuning*. Secara khusus, metode ini mencapai

*F*1-*score* tertinggi dalam mengidentifikasi data berita yang dilabeli "*fake*" dengan benar, dengan nilai parameter optimal 16 untuk ukuran *batch*, 2×10^{-5} untuk laju pembelajaran, dan 8 untuk jumlah *epoch*.

Berdasarkan perbandingan eksperimental antara metode *fine-tuning* pada model IndoBERT dengan tiga metode *hyperparameter tuning* lainnya, terlihat jelas *bahwa Bayesian optimization* lebih unggul dalam mengoptimalkan nilai evaluasi pada saat *fine-tuning*. Kesimpulan ini didukung oleh fakta bahwa nilai evaluasi yang diperoleh melalui *hyperparameter tuning Bayesian optimization* lebih tinggi jika dibandingkan dengan nilai evaluasi yang diperoleh dari *fine-tuning* IndoBERT-base-p1 pada *dataset* TurnBackHoax.ID. Sebaliknya, hasil evaluasi dari *hyperparameter tuning grid search* dan *random search* tidak menunjukkan adanya peningkatan hasil evaluasi pada saat dilakukan *fine-tuning*. Hal ini terjadi karena nilai evaluasi yang dihasilkan setelah dilakukan *hyperparameter tuning* menggunakan *grid search* dan *random search* mengalami penurunan jika dibandingkan dengan nilai evaluasi yang dihasilkan pada saat proses *fine-tuning* awal.

Terlepas dari peningkatan yang dihasilkan dengan menggunakan *Bayesian optimization*, model ini membutuhkan waktu tambahan untuk menemukan nilai *hyperparameter* yang optimal. Oleh karena itu, penelitian berikutnya sebaiknya mempertimbangkan evaluasi perbandingan efisiensi dan efektifitas dari *evolutionary algorithm*, seperti *gradient-based optimization* yang bekerja dengan memperbarui nilai *hyperparameter* secara bertahap hingga ditemukannya solusi optimal.

KONTRIBUSI PENULIS

Konseptualisasi, metodologi dan model, Rosni Lumbantoruan, Anugerah Simanjuntak, Kartika Sianipar, dan Rut Gultom; validasi, Rosni Lumbantoruan, Samuel Situmeang, Mario Simaremare, dan Erwin Panggabean; analisis formal, Rosni Lumbantoruan dan Anugerah Simanjuntak; investigasi, Kartika Sianipar, Rut Gultom, dan Samuel Situmeang; sumber daya; Anugerah Simanjuntak, Kartika Sianipar, dan Rut Gultom; kurasi data, Anugerah Simanjuntak, Kartika Sianipar, dan Rut Gultom; penulisan—persiapan draft awal, Rosni Lumbantoruan, Anugerah Simanjuntak, Rut Gultom, dan Kartika Sianipar; penulisan—penelaahan dan penyuntingan, Rosni Lumbantoruan dan Samuel Situmeang; visualisasi, Anugerah Simanjuntak; supervisi, Rosni Lumbantoruan; admin proyek, Rosni Lumbantoruan.

UCAPAN TERIMA KASIH

Penelitian ini didukung oleh Lembaga Penelitian dan Pengabdian kepada Masyarakat (LPPM) Institut Teknologi Del. Terima kasih disampaikan atas dukungan yang diberikan untuk mendanai publikasi dari penelitian ini. Komitmen Institut Teknologi Del melalui LPPM untuk mempromosikan dan memfasilitasi kegiatan penelitian telah berperan penting dalam keberhasilan penyelesaian proyek ini.

REFERENSI

[1] V.B. Kusnandar (2021) "Pengguna internet Indonesia peringkat ke-3 terbanyak di Asia," [Online], https://databoks.katadata.co.id/datapublish/2021/10/14/pengguna-internet-indonesia-peringkat-ke-3-terbanyak-di-asia, tanggal akses: 10-Jan-2023.

[2] M.A. Rahmat dan I.S. Areni, "Hoax web detection for news in Bahasa using support vector machine," *2019 Int. Conf. Inf. Commun. Technol. (ICOIACT)*, 2019, hal. 332-336, doi: 10.1109/ICOIACT46704.2019.8938425.

[3] A. Thota, P. Tilak, S. Ahluwalia, dan N. Lohia, "Fake news detection: A deep learning approach," *SMU Data Sci. Rev.*, vol. 1, no. 3, hal. 1-20, 2018.

[4] R. Lumbantoruan dkk., "Analysis comparison of FastText and Word2vec for detecting offensive language," *2022 IEEE Int. Conf. Comput. Sci. Inf. Technol. (ICOSNIKOM)*, 2022, hal. 1-8, doi: 10.1109/ICOSNIKOM56551.2022.10034886.

[5] I. Nadzir, S. Seftiani, dan Y.S. Permana, "Hoax and misinformation in Indonesia: Insights from a nationwide survey," *ISEAS-Yusof Ishak Inst.*, vol. 2019, hal. 1-12, Nov. 2019.

[6] A. Yuliani (2017) "Ada 800.000 situs penyebar hoax di Indonesia," [Online], https://www.kominfo.go.id/content/detail/12008/ada-800000-situs-penyebar-hoax-di-indonesia/0/sorotan_media, access date: 10-Jan-2023.

[7] R. Sultana dan T. Nishino, "Fake news detection system: An implementation of BERT and boosting algorithm," *Proc. 38th Int. Conf. Comput. Their Appl.*, 2023, hal. 124-137, doi: 10.29007/d931.

[8] L.H. Suadaa, I. Santoso, dan A.T.B. Panjaitan, "Transfer learning of pre-trained transformers for COVID-19 hoax detection in Indonesian language," *Indones. J. Comput. Cybern. Syst. (IJCCS)*, vol. 15, no. 3, hal. 317-326, Jul. 2021, doi: 10.22146/ijccs.66205.

[9] J. Devlin, M.-W. Chang, K. Lee, dan K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Proc. 2019 Conf. N. Am. Chapter Assoc. Comput. Linguist., Hum. Lang. Technol.*, 2019, hal. 4171-4186, doi: 10.18653/v1/N19-1423.

[10] J. Fawaid, A. Awalina, R.Y. Krisnabayu, dan N. Yudistira, "Indonesia's fake news detection using transformer network," *Proc. 6th Int. Conf. Sustain. Inf. Eng. Technol.*, 2021, hal. 247-251, doi: 10.1145/3479645.3479666.

[11] M. Guderlei dan M. Aßenmacher, "Evaluating unsupervised representation learning for detecting stances of fake news," *Proc. 28th Int. Conf. Comput. Linguist.*, 2020, hal. 6339-6349, doi: 10.18653/v1/2020.coling-main.558.

[12] R.R. Rajalaxmi dkk., "Optimizing hyperparameters and performance analysis of LSTM model in detecting fake news on social media," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, to be published, doi: 10.1145/3511897.

[13] N. Kanagavalli, S.B. Priya, dan D. Jeyakumar, "Design of hyperparameter tuned deep learning based automated fake news detection in social networking data," *2022 6th Int. Conf. Comput. Methodol. Commun. (ICCMC)*, 2022, hal. 958-963, doi: 10.1109/ICCMC53470.2022.9753739.

[14] C.W. Kencana, E.B. Setiawan, dan I. Kurniawan, "Hoax detection system on Twitter using feed-forward and back-propagation neural networks classification method," *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 4, no. 4, hal. 655-663, Agu. 2020, doi: 10.29207/resti.v4i4.2038.

[15] M.E. Peters dkk., "Deep contextualized word representations," *Proc. 2018 Conf. N. Am. Chapter Assoc. Comput. Linguist., Hum. Lang. Technol.*, 2018, hal. 2227-2237, doi: 10.18653/v1/N18-1202.

[16] R.K. Kaliyar, A. Goswami, dan P. Narang, "FakeBERT: Fake news detection in social media with a BERT-based deep learning approach," *Multimed. Tools Appl.*, vol. 80, no. 8, hal. 11765-11788, Mar. 2021, doi: 10.1007/s11042-020-10183-2.

[17] F. Koto, A. Rahimi, J.H. Lau, dan T. Baldwin, "IndoLEM and IndoBERT: A benchmark dataset and pre-trained language model for Indonesian NLP," *Proc. 28th Int. Conf. Comput. Linguist.*, 2020, hal. 757--770, doi: 10.18653/v1/2020.coling-main.66.

[18] S.M. Isa, G. Nico, dan M. Permana, "IndoBERT for Indonesian fake news detection," *ICIC Express Lett.*, vol. 16, no. 3, hal. 289-297, Mar. 2022, doi: 10.24507/icicel.16.03.289.

[19] B. Bischl dkk., "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," *WIREs Data Min. Knowl. Discov.*, vol. 13, no. 2, hal. 1-43, Mar./Apr. 2023, doi: 10.1002/widm.1484.

[20] T. Akiba dkk., "Optuna: A next-generation hyperparameter optimization framework," *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2019, hal. 2623-2631, doi: 10.1145/3292500.3330701.

[21] S.A. Alasadi dan W.S. Bhaya, "Review of data preprocessing techniques in data mining," *J. Eng. Appl. Sci.*, vol. 12, no. 16, hal. 4102-4107, Sep. 2017, doi: 10.3923/jeasci.2017.4102.4107.