Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.2, April 2025, pp. 432~441
e-ISSN: 2548-6861

SciBERT Optimisation for Named Entity Recognition on NCBI Disease Corpus with Hyperparameter Tuning

Syaiful Rizal Sidiq[1]*, Abu Salam[2]*
* Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro
111202113849@mhs.dinus.ac.id [1], abu.salam@dsn.dinus.ac.id [2]

Article Info

Article history:

Received 2025-03-14
Revised 2025-03-17
Accepted 2025-03-19

Keyword:

Named Entity Recognation (NER),
SciBERT,
Hyperparameter Tuning,
Biomedical Text Processing,
Natural Language Processing (NLP).

ABSTRACT

Named Entity Recognition (NER) in the biomedical domain faces challenges due to the variety of medical terms and context ambiguity. Transformer-based models like SciBERT have proven effective for natural language processing (NLP) tasks in scientific domains, but their performance depends on proper hyperparameter selection. This study analyses the impact of hyperparameter tuning on SciBERT performance using the NCBI Disease Corpus dataset. Experiments were conducted by training a SciBERT baseline and then optimizing hyperparameters using Grid Search, Random Search, and Bayesian Optimization. Model evaluation was based on precision, recall, and F1-score.vResults show that Grid Search and Random Search produced the best performance with learning rate 3e-5, batch size 16, dropout 0.2, weight decay 0.01, and AdamW optimizer, achieving precision, recall, and F1-score of 0.82, an improvement from the baseline (precision 0.72, recall 0.72, F1-score 0.68). The McNemar test confirmed this improvement as statistically significant (p-value = 0.0000).vThis study confirms that hyperparameter tuning enhances SciBERT's accuracy in medical entity extraction. The findings contribute to optimization methods in biomedical text processing, particularly in improving Transformer-based models for NER.

I. PENDAHULUAN

Pada saat ini, perkembangan teknologi yang cepat tidak dapat disangkal. Kemajuan dalam teknologi komputasi telah mendorong berbagai inovasi dalam berbagai bidang, di antaranya dalam Natural Language Processing (NLP). NLP merupakan cabang kecerdasan buatan yang memungkinkan komputer untuk memahami, menganalisis, serta memproses bahasa manusia[1]. Teknologi ini digunakan dalam berbagai aplikasi, seperti terjemahan mesin, sistem respons (QA), pengumpulan informasi, chatbot, dan sistem untuk rekomendasi[1][2]. Dalam bidang medis, NLP berperan penting dalam mengekstraksi informasi dari teks yang tidak terstruktur. Salah satu teknik NLP yang banyak digunakan adalah Named Entity Recognition (NER)[3], yang bertujuan untuk mengidentifikasi dan mengklasifikasikan entitas medis seperti nama penyakit, obat- obatan, dan prosedur medis dalam teks klinis atau literatur ilmiah[4][5]. Namun, penerapan NER di bidang medis memiliki tantangan tersendiri, mengingat banyaknya istilah medis yang kompleks dan beragam yang harus dipahami oleh model NLP[6][7].

Untuk meningkatkan akurasi dalam tugas NER pada domain medis, model berbasis *deep learning* telah banyak dikembangkan dan digunakan, salah satunya adalah SciBERT. SciBERT merupakan model berbasis Bidirectional Encoder Representations from Transformers (BERT) yang dirancang khusus untuk menangani teks ilmiah, termasuk dalam domain medis, sehingga lebih unggul dalam memahami terminologi spesifik dibandingkan model NLP umum[6][8]. Dalam penelitian ini, digunakan dataset NCBI Disease Corpus, yaitu kumpulan teks medis beranotasi yang berfokus pada identifikasi nama penyakit. Dataset ini sangat relevan dalam tugas NER medis karena menyediakan data yang kaya akan informasi penyakit yang telah anotasi secara manual oleh ahli[9].

Namun, meskipun SciBERT telah menunjukkan peforma yang baik dalam tugas NER, optimalisasi model tetap diperlukan untuk memastikan hasil yang lebih efisien dan

akurat. Salah satu tantangan utama dalam penerapan model *deep learning* adalah pemilihan hyperparameter yang optimal, yang dapat berdampak langsung pada konvergensi, generalisasi, dan kinerja model[10]. Hyperparameter seperti *learning rate, batch size, dropout, optimizer*, dan *weight decay* memainkan peran penting dalam menentukan seberapa baik model dapat belajar dari data[11]. Pemilihan hyperparameter yang tidak tepat dapat menyebabkan *overfitting* atau *underfitting*[11]. Oleh karena itu, perlu ada pendekatan sistematis untuk mengeksplorasi dan menetapkan hyperparameter yang optimal untuk SciBERT dalam konteks NER medis. Dalam hal ini, untuk mencapai performa terbaik dari model SciBERT dalam NER, tuning hyperparameter menjadi langkah penting.

Menurut Ilemobayo et al.[10], tuning yang tepat dapat meningkatkan kinerja model secara signifikan. Pemilihan hyperparameter seperti *learning rate, batch size*, dan jumlah *epoch* mempengaruhi sejauh mana model dapat mengonvergensi dan generalisasi pada data yang tidak terlihat. Teknik seperti grid search, random search, dan bayesian optimization dapat diterapkan untuk menentukan kombinasi hyperparameter yang optimal. Selain itu, penggunaan teknik regularisasi dan cross-validation dalam pelatihan model dapat mengurangi risiko *overfitting* dan memastikan bahwa model dapat beradaptasi dengan baik pada data baru.

Selain itu, penelitian yang dilakukan oleh Belgaty et al.[12] memperkenalkan SciBERT sebagai model yang mampu memahami teks ilmiah dengan lebih baik dibandingkan BERT umum, tetapi penelitian mereka hanya menyajikan hasil baseline tanpa eksplorasi tuning lebih lanjut.

Penelitian yang dilakukan oleh Lee at al.[13], membahas upaya meningkatkan kinerja model BioBERT dalam NER pada teks medis, penting untuk menyoroti peran krusial tuning hyperparameter. Pemilihan hyperparameter yang tepat, termasuk *batch size, learning rate,* dan langkah pelatihan, secara signifikan mempengaruhi efektivitas model yang diadaptasi dari *pre-trained* language models (PLMs).

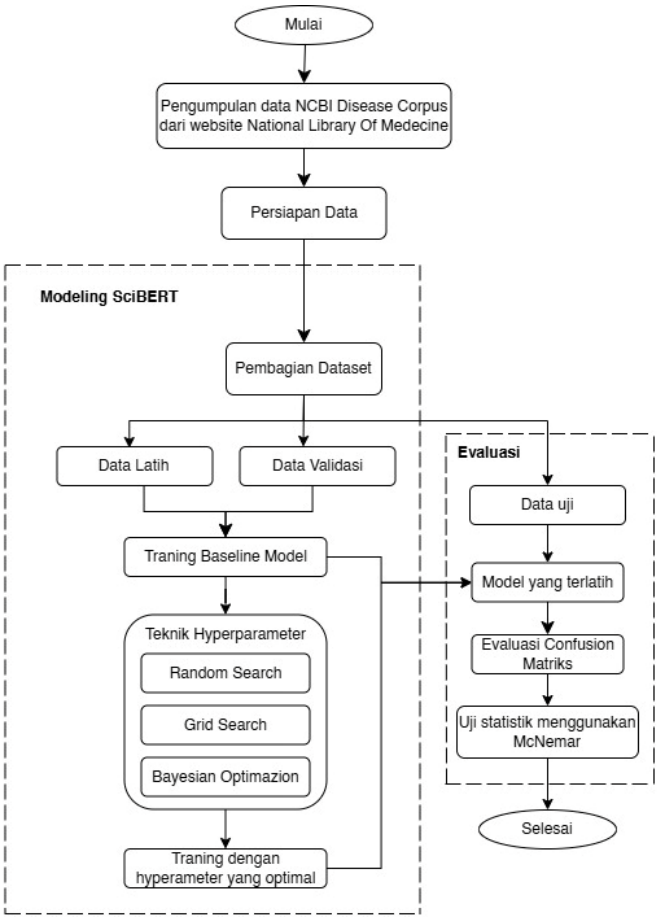
Berdasarkan penelitian yang dilakukan oleh Abbas et al. [14], SciBERT menunjukkan pencapaian F1-score tertinggi, yaitu 95.31% dalam klasifikasi konsep klinis, dibandingkan dengan BERTBase dan DistilBERT yang masing-masing mencapai 92.67% dan 90.45% . Kinerja yang lebih baik ini menunjukkan bahwa SciBERT lebih mampu menangkap konteks dan kompleksitas bahasa medis.

Dalam penelitian ini, dilakukan optimasi SciBERT menggunakan random search, grid search dan bayesian optimization pada dataset NCBI Disease Corpus. Tujuan utama dari penelitian ini adalah menganalisis dampak optimasi hyperparameter terhadap performa SciBERT dalam tugas NER, dengan fokus pada *precision, recall*, dan *f1-score* sebagai matrix evaluasi utama. Penelitian ini juga bertujuan untuk membandingkan keefektifan ketiga teknik optimasi guna menentukan pendekatan yang lebih sesuai dalam meningkatkan akurasi deteksi entitas penyakit.

Dari latar belakang tersebut, penelitian ini diharapakan dapat memberikan kontribusi dalam pengembangan model SciBERT dalam biomedis yang lebih akurat dan efisien. Hasil penelitian ini dapat menjadi referensi dan wawasan bagi para peneliti dan praktisi di bidang NLP medis dalam memilih metode optimasi hyperparameter yang tepat untuk meningkatkan performa model berbasis transformer seperti SciBERT.

II. METODE

Proses penelitian dirancang sesuai pada gambar 1, yang menggambarkan tahap-tahapan penelitian. Penjelasan mengenai langkah penelitian yang dilakukan untuk mencapai tujuan sebagai berikut.



Gambar 1. Tahapan penelitian

A. Pengumpulan Data

Penelitian ini menggunakan dataset NCBI Disease corpus, yang diperoleh dari Website National Library of Medicine. Dataset ini terdiri dari 769 abstrak dari PubMed teks ilmiah medis. Terdapat total 6.892 penyebutan penyakit dalam dataset mencakup berbagai istilah penyakit dan variasinya. Dataset ini terbagi menjadi 3 yaitu data training sebanyak 5.145 data, 787 data development dan 960 data testing [9].

B. Persiapan Data

1. Prepocessing

Prepocessing adalah tahapan penting dalam pengolahan dataset teks yang bertujuan untuk membersihkan dan meningkatkan kualitas data sebelum diproses lebih lanjut. Langkah ini membantu memastikan informasi yang diekstraksi lebih akurat dan berkontribusi pada peningkatan performa model[3][15]. Sebelum digunakan dalam pelatihan model, dataset mengalami beberapa tahap preprocessing untuk meningkatkan kualitas data dan memastikan model dapat memahami informasi dengan lebih baik. Langkah pertama adalah menghapus tag pada kalimat, yang umumnya ditemukan dalam dataset medis seperti NCBI Disease Corpus. Tag ini tidak memiliki makna semantik yang berguna bagi model dan dapat mengganggu tokenisasi teks. Selanjutnya, konversi ke huruf kecil dilakukan untuk memastikan model tidak membedakan antara kata yang sama dengan kapitalisasi berbeda, sehingga mengurangi ukuran kosakata dan meningkatkan efisiensi pelatihan.

Langkah berikutnya adalah menghapus tanda baca, yang bertujuan untuk menghilangkan karakter non-alfabet yang tidak berkontribusi terhadap pemahaman teks dalam konteks Named Entity Recognition (NER). Selain itu, penghapusan stopword juga diterapkan untuk mengurangi kata-kata umum yang tidak memiliki makna spesifik dalam tugas NER. Stopword seperti the, and, is tidak membawa informasi penting terkait identifikasi entitas medis, sehingga menghilangkannya dapat membantu model fokus pada kata-kata yang lebih bermakna. Dengan menerapkan preprocessing ini, dataset menjadi lebih bersih dan efisien untuk digunakan dalam pelatihan model SciBERT, memungkinkan model untuk lebih fokus pada pola yang relevan dalam teks.

2. Tokenisasi

Setelah dilakukan tahap *preprocessing,* selanjutnya adalah tokenisasi menggunakan SciBERT Tokenizer dari Hugging Face. Proses ini untuk memecah teks menjadi token-token yang dapat dipahami oleh model dan mengonversinya menjadi ID numerik yang sesuai. Token-token ini kemudian digunakan sebagai input untuk langkah selanjutnya.

3. Konversi Format IOBES

Setelah tokenisasi, token-token diberi label menggunakan format IOBES (Inside, Outside, Beginning, End, Single). Label "O" digunakan secara default untuk token yang bukan bagian dari entitas. Untuk token yang termasuk entitas, label B-Disease, I-Disease, E-Disease, dan S-Disease diterapkan sesuai posisinya dalam entitas.

4. Padding dan Attention Mask

Pada tahap ini, token yang telah diberi label dipadatkan menjadi panjang maksimum 256 token menggunakan token [PAD]. Token yang lebih pendek dari 256 akan ditambah [PAD], sedangkan yang lebih panjang akan dipotong. Bersamaan dengan itu, dibuat juga attention mask, yaitu array yang berisi 1 untuk token yang valid dan 0 untuk token [PAD]. Attention mask ini membantu model untuk mengabaikan token [PAD] selama proses pelatihan. Tahapan ini penting untuk dilakukan guna menghindari noise dalam data serta untuk memastikan data dalam format yang optimal sebelum digunakan untuk pelatihan model.

C. Modelling

1. Split Data

Split dataset untuk melatih model SciBERT digunakan data latih yang diambil dari dataset sebanyak 5.145 data, yang kemudian dibagi (80% train dan 20% validasi). Sedangkan data *testing* yang digunakan sebanyak 960 data yang diambil dari dataset. Data *testing* ini 100% nantinya digunakan untuk meguji hasil dari model yang telah dilatih sebelumnya.

2. Built Model

Setelah proses split dataset, langkah selanjutnya adalah membangun model klasifikasi menggunakan SciBERT, varian BERT yang dilatih khusus pada literatur ilmiah. Model ini mengubah token hasil tokenisasi menjadi vektor menggunakan *word embeddings, position embeddings,* dan *token type embeddings*. lalu memprosesnya melalui 12 lapisan *encoder* dengan mekanisme *self-attention* untuk memahami konteks antar token. Di bagian output, terdapat lapisan *fully connected (classifier)* yang memprediksi label IOBES (O, B-Disease, I-Disease, E-Disease, S-Disease) untuk setiap token. *Pretrained weights* dari SciBERT digunakan sebagai baseline untuk mengevaluasi performa sebelum dilakukan optimasi lebih lanjut.

3. Training Baseline Model

Selanjutnya *training* model SciBERT menggunakan *baseline* hyperparameter yang digunakan serupa pada penelitian yang dilakukan oleh Beltagy et al.[12], dengan konfigurasi sebagai berikut:

- **learning rate :** 2e-5 (default yang direkomendasikan untuk fine-tuning model Transformer dari peneltian tersebut)
- **optimizer :** Adam (digunakan karena kemampuannya dalam menyesuaikan learning rate secara adaptif dan mencegah vanishing gradient)
- **dropout :** 0.1 (diterapkan untuk mencegah *overfitting*, sebagaimana digunakan dalam penelitian SciBERT)
- **batch size :** 32 (dipilih untuk menyeimbangkan stabilitas pelatihan dan efisiensi komputasi)
- **epoch :** 3, 5, dan 10 (untuk mengeksplorasi efek jumlah epoch terhadap konvergensi model)

Kemudian, model dievaluasi menggunakan validation set yang diperoleh dari pembagian dataset sebesar 80% data *train* dan 20% data validasi. *Validation set* digunakan untuk mengukur performa model selama pelatihan dan menentukan kapan model mulai mengalami *overfitting*. Tujuan dari pelatihan *baseline* ini adalah untuk memperoleh performa awal model sebelum dilakukan optimasi lebih lanjut. *Baseline*

model berfungsi sebagai titik acuan untuk mengukur seberapa besar peningkatan performa yang diperoleh setelah hyperparameter tuning.

4. Hyperparamter Tuning

Setelah baseline model dilatih, langkah berikutnya adalah melakukan hyperparameter tuning dengan tiga metode berbeda untuk mencari kombinasi parameter terbaik yang dapat meningkatkan performa model. Selama proses tuning, ketiga metode tuning yang dilatih menggunakan model SciBERT ini dievaluasi menggunakan validation set yang diperoleh dari pembagian dataset sebesar 80% untuk train dan 20% untuk validasi. *Validation set* digunakan untuk mengukur performa model selama pelatihan dan menentukan kapan model mulai mengalami *overfitting*. Teknik-teknik yang digunakan berdasarkan referensi penelitian yang dilakukan oleh Ilemobayo et al.[10] adalah sebagai berikut.

- Random Search

Random search merupakan metode tuning yang mengambil sampel secara acak dari ruang hyperparameter[16]. Metode ini lebih efisien daripada grid search karena dapat menjelajahi kombinasi lebih luas dan menemukan pengaturan optimal lebih cepat. Dalam melatih model dengan metode ini digunakan konfigurasi hyperparameter dengan ruang pencarian atau *search space.*

- Learning rate : {1e-5, 2e-5, 3e-5}
- Batch size : 16
- Dropout : {0.1, 0.2}
- Weight decay : (0, 0.01)
- Optimizer : AdamW
- Epoch : (3, 5, 10)

Stopping criteria ditetapkan sebanyak 10 percobaan dengan kombinasi yang dipilih secara acak untuk menyeimbangkan eksplorasi dan efisiensi komputasi, mengingat setiap kombinasi memerlukan waktu pelatihan yang cukup lama. Strategi ini memungkinkan model mengeksplorasi berbagai konfigurasi tanpa membebani sistem secara berlebihan, sekaligus meningkatkan peluang menemukan pengaturan hyperparameter yang optimal.

- Grid Search

Grid search merupakan metode tuning yang melakukan pencarian secara sistematis pada semua kombinasi hyperparameter yang telah ditentukan sebelumnya[16][17]. Meskipun mudah diterapkan, grid search dapat sangat memakan waktu dan sumber daya. Selanjutnya model dilatih dengan metode ini menggunakan konfigurasi hyperparameter dengan ruang pencarian atau *search space* sebagai berikut.

- Learning rate : {2e-5, 3e-5}
- Batch size : {16, 32}
- Dropout : {0.1, 0.2}
- Weight Decay : {0, 0.01}
- Optimizer : AdamW
- Epoch : {3, 5, 10}

Dari kombinasi hyperparameter tersebut, total percobaan yang dilakukan sebanyak 48 kali, yang juga sekaligus sebagai *stopping criteria* dalam eksperimen menggunakan metode ini.

- Bayesian Optimization

Bayesian optimization adalah metode tuning yang menggunakan model probabilistik untuk memperkirakan fungsi objektif[18]. Metode ini memilih kombinasi hyperparameter yang menjanjikan secara iteratif, sehingga mengoptimalkan performa dengan lebih sedikit evaluasi. Dalam percobaan ini, model dilatih menggunakan bayesian optimization dengan ruang pencarian atau hyperparameter yang mencakup :

- Learning rate : {2e-5, 3e-5}
- Batch size : {16, 32}
- Dropout : {0.1, 0.2}
- Weight decay : {0, 0.01}
- Optimizer : AdamW
- Epoch : {3, 5, 10}

Dalam eksperimen ini, *stopping criteria* ditetapkan sebanyak 10 percobaan untuk mengoptimalkan efisiensi komputasi, mengingat bayesian optimization memerlukan evaluasi probabilistik tambahan di setiap iterasi. Pembatasan ini juga diterapkan karena setiap kombinasi hyperparameter membutuhkan waktu pelatihan yang signifikan, dan jumlah iterasi yang lebih banyak akan meningkatkan beban komputasi secara drastis. Meskipun bayesian optimization dapat mencapai hasil optimal dengan iterasi yang lebih sedikit dibandingkan grid search, keterbatasan sumber daya komputasi menjadi pertimbangan utama dalam menetapkan batas percobaan. Dengan strategi ini, model tetap dapat mengeksplorasi kombinasi hyperparameter yang menjanjikan tanpa mengorbankan efisiensi komputasi.

5. Training Model dengan Hyperparameter Terbaik

Setelah mendapatkan hasil kombinasi hyperparameter terbaik dari tiga metode tuning, dilakukan pelatihan ulang model SciBERT menggunakan konfigurasi terbaik tersebut. Tujuan dari pelatihan ulang ini adalah untuk memastikan bahwa model dapat memanfaatkan pengaturan parameter yang optimal untuk mencapai performa lebih baik dibandingkan *baseline*.

D. Evaluasi

Tahap evaluasi ini terbagi menjadi 2 tahapan, tahap pengujian model yaitu melatih model yang sudah di *training* sebelumnya baik untuk *baseline* model maupun model hasil tuning, kedua model diuji menggunakan data uji. Pengujian ini bertujuan untuk menilai seberapa jauh model dapat menggeneralisasi data baru yang belum pernah ditemui sebelumnya. Proses pengujian ini menghasilkan prediksi entitas yang akan digunakan dalam evaluasi kinerja model.

Selanjutnya, tahap terakhir dari penelitian ini yaitu melakukan evaluasi terhadap performa model menggunakan *confusion matrix*. *confusion matrix*, merupakan instrumen

yang digunakan untuk mengevaluasi kinerja model NER dengan mengukur seberapa baik model tersebut mengelompokkan entitas penyakit pada dataset uji. Evaluasi ini melibatkan empat komponen utama, True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN)[15][19]. Berdasarkan hasil dari *confusion matrix*, penelitian ini menghitung empat matrix evaluasi utama, yaitu *f1-score, precision, recall*, dan *accuracy*, untuk menilai seberapa akurat dan efektif model dalam mendeteksi entitas secara tepat[20]. Hasil dari nilai yang diperoleh *confusion matrix*, bisa didapatkan informasi tambahan untuk mengevaluasi kinerja model yang digunakan. Informasi ini mencakup :

1. Accuracy

Accuracy dijelaskan sebegai rasio prediksi nilai benar yang diperoleh dari sejauh mana kesamaan antara nilai prediksi dan nilai aktual. *Accuracy* dihitung menggunakan rumus sebagai berikut (1) :

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

2. Precision

Precision megukur proporsi prediksi positif yang banar terhadap suatu prediksi posotif dalam dataset. *Precision* dapat dihitung menggunakan rumus berikut (2) :

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

3. Recall

Recall dapat didenifisikan untuk mengukur data positif yang berhasil dipredikasi dengan benar terhadap semua data positif sebenarnya. *Recall* dapat dihitung menggunakan rumus berikut (3) :

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

4. F1-Score

F1-Score merupakan nilai rerata harmonik yang didapat dari hasil *precision* dan *recall* yang dapat membantu untuk menghitung ketidakseimbangan kelas. Perhitungan *f1-score* dapat menggunakan rumus berikut (4) :

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

Selain metrix evaluasi di atas, penelitian ini juga menggunakan McNemar's Test sebagai metode untuk menguji signifikansi perbedaan antara model *baseline* dan model hasil tuning. McNemar’s Test adalah uji statistik non-parametrik yang digunakan untuk mengevaluasi perbedaan antara dua kelompok yang berpasangan pada variabel dikotomis. Uji ini umum digunakan dalam analisis pretest-post-test atau perbandingan dua model klasifikasi pada dataset yang sama, terutama ketika hasil prediksi bersifat biner (benar atau salah)[21]. Penggunaan McNemar's Test dalam penelitian ini bertujuan untuk menentukan apakah ada perbedaan signifikan antara hasil prediksi dari model yang dibandingkan.

McNemar’s Test menggunakan contingency table 2×2, yang dihitung berdasarkan jumlah instance yang diklasifikasikan secara berbeda oleh dua model yang diuji. Rumus dari McNemar’s Test adalah sebagai berikut (5):

$$X^2 = \frac{(b-c)^2}{b+c} \quad (5)$$

Penjelasan rumus :

- b : Jumlah instance yang salah diklasifikasikan oleh model pertama tetapi benar diklasifikasikan oleh model kedua
- c : Jumlah instance yang benar diklasifikasikan oleh model pertama tetapi salah diklasifikasikan oleh model kedua

Hipotesis nol (H_0) dalam McNemar’s Test menyatakan bahwa kedua model memiliki performa yang sama, sedangkan hipotesis alternatif (H_A) menyatakan bahwa terdapat perbedaan signifikan antara kedua model. Jika p-value < 0.05, maka perbedaan antara model dianggap signifikan.

Dengan menggunakan McNemar's Test, penelitian ini dapat mengidentifikasi apakah model hasil tuning memiliki perbedaan signifikan dibandingkan dengan baseline model dalam mendeteksi entitas penyakit. Ini membantu memastikan bahwa peningkatan performa model tidak hanya disebabkan oleh faktor kebetulan, tetapi benar-benar memberikan manfaat yang nyata dalam proses Named Entity Recognition (NER).

III. HASIL DAN PEMBAHASAN

Bagian ini mengambarkan hasil yang didapat dari setiap tahapan yang sebelumnya dijelaskan pada bagian metode penelitian. Berikut penjelasan hasil yang didapat.

A. Persiapan data

Pada awal penelitian ini, dilakukan tahap persiapan dataset NCBI Disease Corpus yang digunakan untuk melatih dan menguji model SciBERT dalam tugas NER. Dataset ini terdiri dari kumpulan teks medis yang telah diberi anotasi dengan entitas penyakit. Sebelum digunakan dalam proses pelatihan, dataset ini dilakukan tahap *preprocessing* yang meliputi menghapus tag, konversi ke huruf kecil, menghapus tanda baca, menghilangkan *stop word*. Berikut hasil contoh dari tahap *preprocessing* yang dapat dilihat pada Tabel 1.

TABEL 1
HASIL PREPOCESSING

Preprocessing Data	Hasil Preprocessing
Teks asli	Identification of APC2, a homologue of the <category="Modifier">adenomatous polyposis coli tumour</category> suppressor.
Menghapus Tag	Identification of APC2, a homologue of the adenomatous polyposis coli tumour suppressor.
Konversi ke huruf kecil	identification of apc2, a homologue of the adenomatous polyposis coli tumour suppressor.
Menghapus tanda baca	identification of apc2 a homologue of the adenomatous polyposis coli tumour suppressor.
Menghilangkan *Stopword*	identification apc2 homologue adenomatous polyposis coli tumour suppressor.

Setelah data dibersihkan, langkah selanjutnya adalah melakukan tokenisasi menggunakan SciBERT Tokenizer. Proses tokenisasi ini bertujuan untuk memecah teks menjadi token-token yang dapat dipahami oleh model serta mengonversinya menjadi ID numerik yang sesuai. SciBERT Tokenizer menggunakan metode subword tokenization untuk menangani istilah medis yang jarang ditemui, sehingga dapat mempertahankan makna kata secara lebih baik. Sebagai contoh, frasa "adenomatous polyposis coli" dipecah menjadi beberapa *subword* yang masing-masing memiliki ID unik. Hasil tokenisasi ini kemudian digunakan untuk proses pelabelan IOBES. Format ini digunakan untuk menandai posisi setiap token dalam entitas penyakit pada teks medis. Misalnya, token pertama dari entitas diberi label B-Disease (Beginning), token di tengah sebagai I-Disease (Inside), token terakhir sebagai E-Disease (End), dan jika hanya terdiri dari satu token, maka dilabeli S-Disease (Single). Token yang bukan bagian dari entitas diberi label O (Outside).

Kemudian selanjutnya, dilakukan padding untuk memastikan semua token memiliki panjang yang sama, yaitu 256 token. Padding ini bertujuan untuk menghindari informasi hilang pada token yang lebih panjang serta mempermudah pengolahan dalam *batch* selama proses pelatihan. Bersamaan dengan itu, dibuat jug attention mask untuk membedakan token yang valid dan token [PAD], sehingga model hanya memproses informasi yang relevan.

B. Built Models

Semua eksperimen yang dilakukan dijalankan di Jupyter Notebook menggunakan NVIDIA GeForce RTX 3050 GPU dan CUDA. Pada arsitektur model, SciBERT digunakan sebagai backbone untuk menghasilkan representasi kontekstual token. Keluaran SciBERT berupa *last hidden state* yang diteruskan ke lapisan *dropout* untuk mencegah *overfitting*, lalu diproses oleh *linear classifier* dengan jumlah neuron sesuai jumlah label (5 kelas). hasil dari *linear classifier* berupa logits, yang akan diproses lebih lanjut pada tahap evaluasi menggunakan fungsi argmax untuk menentukan kelas dengan skor tertinggi.

Setelah data selesai dipersiapkan, model baseline dilatih menggunakan SciBERT, dengan parameter *learning rate* 2e-5*, batch size* 32, *optimizer* Adam, *dropout* 0.1, dan *epoch* 3,5 dan 10 seperti yang sudah dijelaskan pada bab metode. Dataset dibagi menjadi *train set* dan *validation set* untuk menghindari *overfitting* dan mengukur performa model melalui *train loss* dan *validation loss.*

Selanjutnya merupakan proses training menggunakan hyperparameter tuning untuk meningkatkan kinerja model SciBERT. Oleh karena itu, dilakukan pencarian hyperparameter terbaik dengan menggunakan tiga metode, yaitu grid search, random search, dan bayesian optimization. Masing-masing metode ini memiliki pendekatan yang berbeda dalam mencari kombinasi hyperparameter optimal. Grid search mengevaluasi semua kombinasi yang mungkin dari rentang parameter yang ditentukan. Random search mencari kombinasi secara acak untuk menghemat waktu komputasi. Sedangkan bayesian optimization menggunakan pendekatan probabilistik untuk menemukan parameter terbaik dengan jumlah eksperimen yang lebih sedikit.

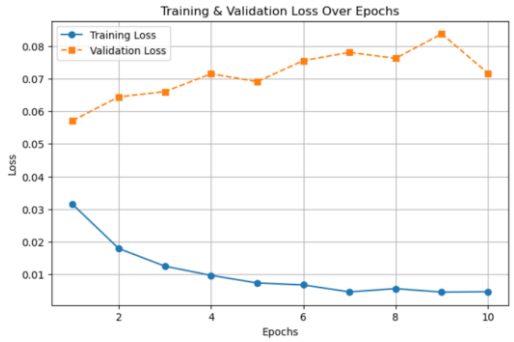
Tahap ini, digunakan beberapa hyperparameter tuning meliputi *learning rate* (1e-5, 3e-5, 5e-5), *batch size* (16, 32), *dropout* (0.1, 0.3, 0.5), *weight decay* (0.01, 0.001), serta jenis *optimizer* AdamW. Setelah di training pada ketiga metode hasil dari tuning menunjukkan bahwa grid search dan random search menghasilkan kombinasi hyperparameter yang sama baiknya, sementara bayesian optimization memberikan hasil yang kurang optimal. Dari hasil training tiga metode tuning didapatkan hasil hyperparamter terbaik yang dapat dilihat pada Tabel 2.

TABEL 2
HASIL HYPERPARAMETER TERBAIK DARI TIGA METODE

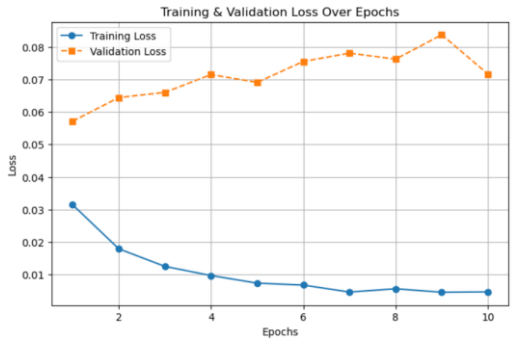
Teknik Optimasi	Optimizer	Weight Decay	Epoch	Learning rate	Dropout	Batch Size
Random Search	AdamW	0.01	10	3e-5	0.2	16
Grid Search	AdamW	0.01	10	3e-5	0.2	16
Bayesian optimization	AdamW	0.066	10	1e-5	0.243	16

Selain itu, dapat dilihat hasil grafik training curve model *baseline* dan model dengan menggunakan tiga metode tuning. Berikut dapat dilihat pada Gambar 2 sampai 5.

Gambar 2. Training curve baseline



Gambar 3. Training curve grid search



Gambar 4. Training curve random search

Gambar 5. Training curve bayesian optimization

Analisis grafik *training loss* dan *validation loss* menunjukkan bagaimana model belajar dan menggeneralisasi data. Pada baseline (Gambar 2), training loss dan validation loss menurun stabil selama 10 epoch tanpa lonjakan signifikan, menandakan model dapat menggeneralisasi dengan baik.

Sementara itu, hasil hyperparameter tuning dengan random search dan grid search (Gambar 3 dan 4) menunjukkan training loss yang terus menurun, tetapi validation loss mengalami fluktuasi atau spike pada beberapa epoch. Lonjakan ini dapat disebabkan oleh karakteristik dataset atau sensitivitas model terhadap perubahan parameter, yang menunjukkan bahwa model terkadang kesulitan beradaptasi dengan data validasi. Meskipun demikian, fluktuasi ini masih dalam batas yang terkendali. Sedangkan, bayesian optimization (Gambar 5) menunjukkan pola penurunan yang lebih stabil tanpa spike mencolok, menandakan proses pembelajaran yang lebih konsisten. Namun, validation loss yang lebih rendah tidak selalu berarti performa terbaik dalam metrik lain seperti *precision, recall*, dan *f1-score.*

Secara keseluruhan, perbedaan pola ini mencerminkan pengaruh tuning hyperparameter terhadap stabilitas model. Setelah pelatihan selesai, model yang telah dilatih akan dievaluasi lebih lanjut pada data testing menggunakan matrix evaluasi untuk mengukur kemampuan generalisasi model.

C. Testing dan Evaluasi Model

Model dengan menggunakan hyperparameter terbaik yang telah diperoleh dari hasil tuning diuji menggunakan dataset uji sebanyak 960 data. Pengujian ini bertujuan untuk mengukur seberapa baik model dapat mengenali entitas penyakit pada teks medis yang belum pernah dilihat sebelumnya. Selain melihat *matrix* evaluasi secara keseluruhan, dilakukan juga analisis lebih dalam terhadap beberapa contoh prediksi model. Hasil pengujian menunjukkan bahwa model SciBERT dengan hyperparameter tuning memiliki kemampuan lebih baik dalam mengenali entitas penyakit dibandingkan model baseline, terutama dalam menangani variasi bahasa yang kompleks dalam teks medis.

Kemudian untuk menganalisis kesalahan prediksi model, digunakan *confusion matrix* yang menggambarkan performa model dalam mengklasifikasikan entitas penyakit dengan lebih rinci. Matrix utama yang dievaluasi adalah *precision, recall, f1-score*, dan *accuracy*, yang diperoleh dari hasil eksperimen dengan berbagai teknik optimasi. Dibawah ini hasil perbandingan antara model tanpa tuning dengan model yang menggunakan tuning hyperparameter dapat dilihat pada Tabel 3.

TABEL 3
HASIL MATRIX PERBANDINGAN PENGUJIAN

Teknik Optimasi	Precision	Recall	F1-Score	Accuracy
Baseline	0.72	0.72	0.68	0.9719
Randon Search	0.82	0.82	0.82	0.9829

Grid Search	0.82	0.82	0.82	0.9829
Bayesian Optimization	0.79	0.71	0.71	0.98

Hasil menunjukkan bahwa model SciBERT baseline memiliki *precision* dan *recall* sebesar 0.72, dengan *f1-score* 0.68 dan *accuracy* 97.19%. Meskipun akurasi baseline cukup tinggi, nilai F1-score yang hanya 0.68 menunjukkan bahwa model masih mengalami kesulitan dalam mendeteksi entitas dengan keseimbangan precision dan recall yang optimal. Setelah dilakukan hyperparameter tuning menggunakan random search dan grid search, terjadi peningkatan yang signifikan pada performa model. Kedua metode ini menghasilkan *precision*, *recall*, dan *f1-score* masing-masing sebesar 0.82, menunjukkan peningkatan sebesar 14% dibandingkan baseline. Hal ini dapat menunjukkan bahwa pemilihan hyperparameter yang optimal dapat meningkatkan performa model dalam tugas Named Entity Recognition (NER).

Sebaliknya, model dengan bayesian optimization menunjukkan performa yang lebih rendah dibandingkan grid search dan random search, dengan *precision 0.79*, *Recall* 0.71 dan *f1-score* 0.71. Penurunan performa ini kemungkinan disebabkan oleh jumlah iterasi bayesian optimization yang dibatasi hanya 10 iterasi, yang belum cukup untuk membangun model probabilistik yang optimal dalam memilih kombinasi hyperparameter terbaik. Selain itu, bayesian optimization bekerja dengan strategi eksplorasi bertahap berdasarkan perkiraan probabilistik, sehingga dengan ruang pencarian yang tidak terlalu luas, metode ini bisa gagal menemukan konfigurasi optimal dibandingkan dengan grid search yang menjelajahi semua kombinasi atau random search yang memiliki kemungkinan menemukan kombinasi baik secara acak.

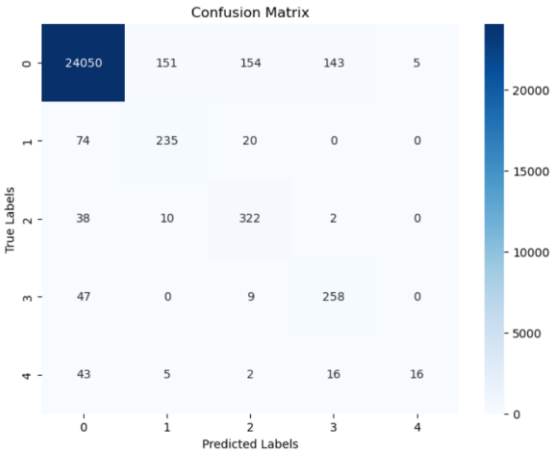
Selain itu, untuk memastikan bahwa peningkatan hasil yang diperoleh bukan hanya kebetulan statistik, dilakukan uji McNemar guna membandingkan hasil model sebelum dan sesudah tuning hyperparameter. Hasil uji menunjukkan bahwa dibandingkan dengan baseline, baik random search maupun grid search memberikan peningkatan yang signifikan, dengan Chi-square value: 204.3532 dan P-value: 0.0000. Demikian pula, Bayesian Optimization juga menunjukkan perbedaan signifikan dibandingkan baseline, dengan Chi-square value: 273.7264 dan P-value: 0.0000. Nilai P-value yang sangat kecil (0.0000) pada semua perbandingan menunjukkan bahwa peningkatan performa setelah tuning hyperparameter bukan sekadar fluktuasi acak, melainkan memiliki dampak nyata terhadap kemampuan model dalam mengenali entitas penyakit secara lebih akurat.

Hasil peningkatan F1-score dari 0.68 ke 0.82 dalam penelitian ini sejalan dengan studi sebelumnya yang menunjukkan bahwa tuning hyperparameter dapat meningkatkan performa model berbasis Transformer dalam tugas NER. Misalnya, penelitian yang dilakukan Yinchun et al.[22] menunjukkan bahwa optimasi hyperparameter dapat meningkatkan efisiensi dan kinerja model tanpa mengorbankan akurasi. Dalam eksperimen pada SQuAD, AutoTinyBERT-KD-S1 mencapai F1-score 87.6%, lebih tinggi dibandingkan beberapa model distilasi lainnya. Temuan ini mendukung bahwa tuning hyperparameter yang tepat berperan penting dalam meningkatkan performa model berbasis Transformer, sebagaimana yang terlihat dalam hasil penelitian ini.

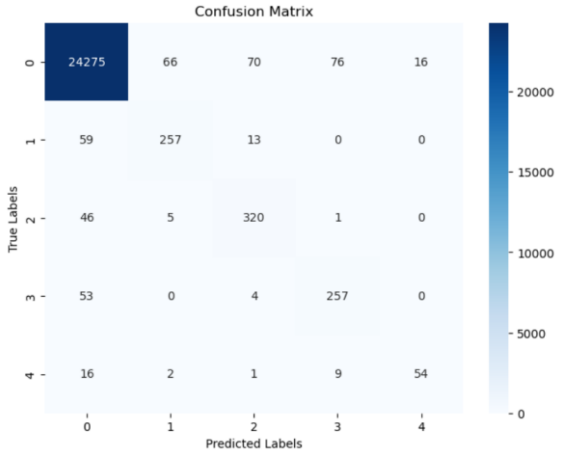
Dapat dilihat juga Tabel 4 dan Gambar 6 sampai 9 berkaitan dengan hasil dari *confusion matrix* baseline dan 3 metode tuning dibawah ini.

TABEL 4
HASIL CONFUSION MATRIX

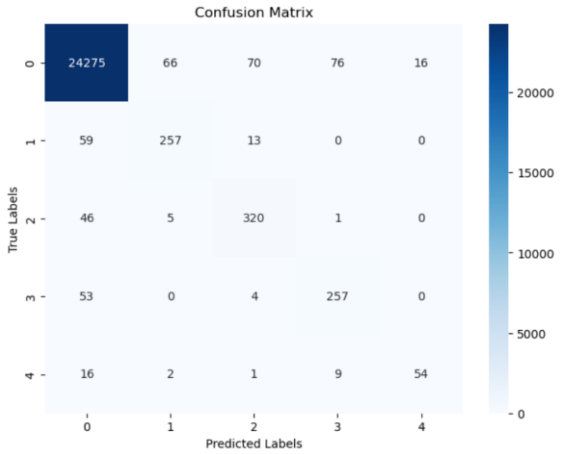
Teknik Optimasi	TP	TN	FP	FN
Baseline	24881	101226	719	719
Randon Search	25163	101966	437	437
Grid Search	25163	101966	437	437
Bayesian Optimization	25030	101830	570	570



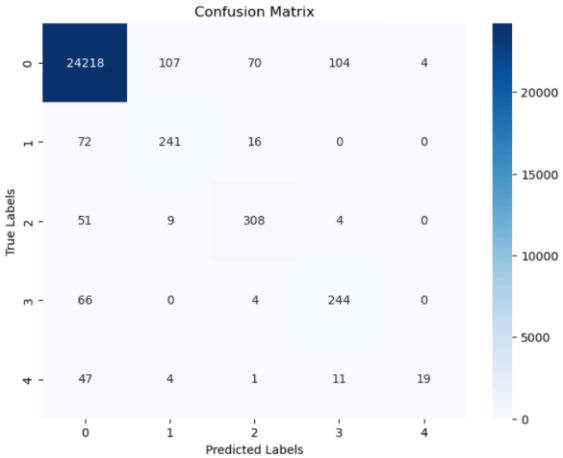
Gambar 6. Hasil confusion matrix baseline



Gambar 7. Hasil confusion matrix grid search



Gambar 8. Hasil confusion matrix random search



Gambar 9. Hasil confusion matrix bayesian optimization

Dari hasil *confusion matrix*, tuning hyperparameter dengan grid search dan random search menunjukkan peningkatan akurasi serta keseimbangan antara *precision, recall,* dan *f1-score* dibandingkan baseline. Kedua metode ini menghasilkan True Positive (TP) tertinggi (25.163) serta False Positive (FP) dan False Negative (FN) terendah (437), menandakan deteksi entitas yang lebih akurat dengan sedikit kesalahan.

Sementara itu, bayesian optimization menunjukkan kinerja yang sedikit lebih rendah dalam mendeteksi entitas penyakit, dengan FP dan FN lebih tinggi (570). Hal ini menyebabkan sedikit penurunan recall dan precision, yang menunjukkan bahwa model mengalami lebih banyak kesalahan dalam mendeteksi entitas dibandingkan metode random search dan grid search. Salah satu kemungkinan penyebabnya adalah jumlah iterasi yang dibatasi hanya 10 iterasi, sehingga bayesian optimization tidak memiliki cukup variasi untuk menemukan kombinasi hyperparameter terbaik. Selain itu, ruang pencarian yang lebih sempit dapat menghambat metode ini dalam mengeksplorasi solusi yang lebih optimal.

Meskipun bayesian optimization menggunakan pendekatan tuning yang lebih sistematis, hasilnya tetap menunjukkan bahwa grid search dan random search lebih unggul. F1-score untuk bayesian optimization hanya mencapai 0.71, sedangkan grid search dan random search mencapai 0.82. Hal ini menunjukkan bahwa eksplorasi ruang hyperparameter yang lebih luas dalam kedua metode tersebut lebih efektif dalam menemukan konfigurasi optimal untuk tugas Named Entity Recognition (NER) pada teks medis. Dengan demikian, Grid Search dan Random Search terbukti sebagai pendekatan yang lebih baik dalam meningkatkan performa model SciBERT.

IV. KESIMPULAN

Penelitian ini mengevaluasi pengaruh optimasi hyperparameter terhadap performa model SciBERT dalam tugas Named Entity Recognition (NER) pada domain biomedis menggunakan NCBI Disease Corpus.Hasil menunjukkan bahwa model baseline memiliki keterbatasan dalam mengenali entitas penyakit secara akurat, dengan F1-score 0.68 dan akurasi 97.19%.

Setelah dilakukan hyperparameter tuning, grid search dan random search berhasil meningkatkan *f1-score* menjadi 0.82 serta akurasi 98.29%, menunjukkan peningkatan signifikan dalam deteksi entitas penyakit. Hasil uji McNemar's juga menunjukkan bahwa perbedaan performa sebelum dan sesudah tuning signifikan secara statistik (p-value = 0.0000), sehingga peningkatan ini bukan hanya kebetulan.

Meskipun demikian, penelitian ini memiliki keterbatasan, terutama dalam cakupan dataset dan jumlah iterasi tuning yang masih terbatas. Hasil yang diperoleh dalam penelitian ini hanya diuji pada dataset NCBI Disease Corpus, sehingga belum tentu memberikan performa yang sama jika diterapkan pada dataset lain. Untuk penelitian selanjutnya, disarankan untuk menggunakan dataset yang lebih luas, seperti BC5CDR atau MedMentions, serta menerapkan metode tuning yang lebih adaptif dan kapasitas komputasi yang lebih tinggi agar eksplorasi hyperparameter lebih optimal.

DAFTAR PUSTAKA

[1] M. Zhou, N. Duan, S. Liu, and H. Shum, "Progress in Neural NLP : Modeling , Learning , and Reasoning," *Engineering*, vol. 6, no. 3, pp. 275–290, 2020, doi: 10.1016/j.eng.2019.12.014.

[2] S. Naseer, M. Mudasar, and S. Khalid, "Named Entity Recognition (NER) in NLP Techniques , Tools Accuracy and Performance," vol. 2, no. 2, pp. 293–308, 2021.

[3] H. N. M. Yusof, S. A. N. Mohd, N. Khamis, and N. Hamzah, "Named Entity Recognition of an Oversampled and Preprocessed Manufacturing Data Corpus," vol. 1, no. 1, pp. 203–216, 2024.

[4] L. Weber *et al.*, "Data and text mining HunFlair : an easy-to-use tool for state-of-the-art biomedical named entity recognition," vol. 37, no. January, pp. 2792–2794, 2021, doi: 10.1093/bioinformatics/btab042.

[5] I. P. Tummala, "Text Summarization based Named Entity Recognition for Certain Application using BERT," *2024 Second Int. Conf. Intell. Cyber Phys. Syst. Internet Things*, no. ICoICI, pp. 1136–1141, 2024, doi: 10.1109/ICoICI62503.2024.10696673.

[6] M. S. Usha, A. M. Smrity, and S. Das, "Named Entity Recognition Using Transfer Learning with the Fusion of Pre-trained SciBERT Language Model and Bi-directional Long Short Term Memory," no. December 2022, 2024, doi: 10.1109/ICCIT57492.2022.10055784.

[7] C. D. Nafanda and A. Salam, "Optimalisasi Model BioBERT untuk Pengenalan Entitas pada Teks Medis dengan Conditional Random Fields (CRF)," vol. 6, no. 4, 2025, doi: 10.47065/bits.v6i4.7042.

[8] A. K. Ambalavanan and M. V Devarakonda, "Using the contextual language model BERT for multi-criteria classification of scientific articles," *J. Biomed. Inform.*, vol. 112, no. September, p. 103578, 2020, doi: 10.1016/j.jbi.2020.103578.

[9] Z. L. Doğan1 Rezarta Islamaj, Robert Leaman1, 2, "NCBI Disease Corpus: A Resource for Disease Name Recognition and Concept Normalization," pp. 1–10, 2014, doi: 10.1016/j.jbi.2013.12.006.NCBI.

[10] J. A. Ilemobayo, O. I. Durodola, T. O. Adewumi, and O. Falana, "Hyperparameter Tuning in Machine Learning : A Comprehensive Review," no. June, 2024, doi: 10.9734/jerr/2024/v26i61188.

[11] J. P. Cuevas, J. A. Reyes-ortiz, A. D. Cuevas-rasgado, and R. A. Mora-gutiérrez, "applied sciences MédicoBERT : A Medical Language Model for Spanish Natural Language Processing Tasks with a Question-Answering Application Using Hyperparameter Optimization," 2024.

[12] I. Belgaty, K. Lo, and A. Cohan, "SCIBERT : A Pretrained Language Model for Scientific Text," pp. 3615–3620, 2019.

[13] J. Lee *et al.*, "Data and text mining BioBERT : a pre-trained biomedical language representation model for biomedical text mining," vol. 36, no. September 2019, pp. 1234–1240, 2020, doi: 10.1093/bioinformatics/btz682.

[14] A. Abbas, M. Lee, N. Shanavas, and V. Kovatchev, "Clinical concept annotation with contextual word embedding in active transfer learning environment," 2024, doi: 10.1177/20552076241308987.

[15] A. Sander, P. Braja, A. Kodar, F. I. Komputer, U. Mercu, and B. Jakarta, "Implementasi Fine-Tuning BERT untuk Analisis Sentimen terhadap Review Aplikasi PUBG Mobile di Google Play Store," vol. 7, no. 3, pp. 120–128, 2023.

[16] B. Bischl *et al.*, "Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges," no. Ml, 2021.

[17] N. Hikmah, A. Suradika, and R. A. Ahmad Gunadi, "Metode Agile Untuk Meningkatkan Kreativitas Guru Melalui Berbagi Pengetahuan (Knowledge Sharing) (Studi Kasus: Sdn Cipulir 03 Kebayoran Lama, Jakarta," *Instruksional*, vol. 3, no. 1, p. 30, 2021, doi: 10.24853/instruksional.3.1.30-39.

[18] A. Candelieri, "A Gentle Introduction to Bayesian Optimization," *IEEE*, pp. 1–16, 2021, doi: 10.1109/WSC52266.2021.9715413.

[19] A. C. Kurniawan and A. Salam, "Seleksi Fitur Information Gain untuk Optimasi Klasifikasi Penyakit Tuberkulosis," vol. 8, pp. 70–82, 2024, doi: 10.30865/mib.v8i1.7122.

[20] F. Fajri, B. Tutuko, and S. Sukemi, "Membandingkan Nilai Akurasi BERT dan DistilBERT pada Dataset Twitter Tahapan Penelitian," vol. 8, no. 2, 2022.

[21] A. Asefa, A. Morgan, M. A. Bohren, and M. Kermode, "Lessons learned through respectful maternity care training and its implementation in Ethiopia : an interventional mixed methods study," pp. 1–12, 2020.

[22] Y. Yin, C. Chen, L. Shang, X. Jiang, X. Chen, and Q. Liu, "AutoTinyBERT: Automatic Hyper-parameter Optimization for Efficient Pre-trained Language Models," 2021, doi: https://doi.org/10.48550/arXiv.2107.13686.