Jurnal Informatika: Jurnal pengembangan IT

Vol. 11, No. 2, 2026
ISSN : 2477-5126, DOI:10.30591/jpit.v11i2.10170

Analisis Efektivitas *Fine-Tuning* dan *Prompt Engineering* Berbasis Llama 3.1 pada Deteksi Depresi di Media Sosial

Muhammad Ikhsan Asagaf [1], Junta Zeniarja [2]
[1,2] Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Indonesia
[1] ikhsan.asagaf2012@gmail.com,[2] junta@dsn.dinus.ac.id

Info Artikel

Riwayat Artikel:

Received 2026-01-13
Revised 2026-04-10
Accepted 2026-04-21

Corresponding Author:

Muhammad Ikhsan Asagaf

Email:

ikhsan.asagaf2012@gmail.com

Abstract *– Detecting depression has become an important concern in addressing mental health issues. According to WHO, more than 300 million people suffer from depression. Large Language Models offer great potential to address this issue, however the full fine-tuning process is often hampered by heavy computational requirements, and LLMs that are not specifically configured for a particular context can result in biased and inaccurate outcomes. This study aims to analyze the effectiveness of Prompt Engineering and Fine-Tuning using QLoRA in improving the accuracy of depression detection. Utilizing the Llama-3.1-8B-Instruct model on social media datasets, this research compares model performance in two scenarios consist of the application of direct prompting strategies on the base model and the application of QLoRA fine-tuning. Evaluation results demonstrate that the Chain-of-Thought strategy improved baseline accuracy from 81.4% to 84.4%, but still exhibited significant bias towards the 'Severe' class. In contrast, the QLoRA Fine-Tuning approach proved superior, achieving 92.4% accuracy with balanced F1-Scores across classes, effectively eliminating detection bias in the 'Minimum' class. These findings confirm that while prompting techniques can enhance baseline performance, QLoRA provides a more accurate, stable, and objective solution for depression detection tasks.*

Keywords: *Depression; Fine-Tuning; LLM; Prompt Engineering; QLoRA.*

Abstrak *– Deteksi depresi menjadi perhatian penting dalam menangani masalah kesehatan mental. Menurut WHO, terdapat lebih dari 300 juta orang mengidap depresi. Large Language Model menawarkan potensi besar untuk pemecahan masalah ini, namun proses full fine-tuning seringkali terhambat oleh kebutuhan komputasi yang berat dan LLM yang tidak dikonfigurasi secara khusus untuk suatu konteks dapat mengakibatkan hasil yang bias dan tidak akurat. Penelitian ini bertujuan untuk menganalisis efektivitas dari penerapan ilmu Prompt Engineering dan Fine-Tuning menggunakan QLoRA dalam meningkatkan akurasi deteksi depresi. Menggunakan model Llama-3.1-8B-Instruct pada dataset media sosial, penelitian ini membandingkan kinerja model dalam dua skenario, yakni penerapan strategi prompting langsung pada base-model, dan penerapan fine-tuning QLoRA. Hasil evaluasi menunjukkan bahwa strategi Chain-of-Thought mampu meningkatkan akurasi baseline dari 81,4% menjadi 84,4%, namun masih menunjukkan bias signifikan terhadap kelas Severe. Sebaliknya, metode Fine-Tuning QLoRA terbukti jauh lebih unggul, mencapai akurasi 92,4% dengan F1-Score yang seimbang antar kelas, serta berhasil mengeliminasi bias deteksi pada kelas Minimum. Temuan ini mengonfirmasi bahwa meskipun teknik prompting dapat meningkatkan kinerja dasar, pendekatan QLoRA memberikan solusi yang lebih akurat, stabil, dan objektif untuk tugas deteksi depresi.*

Kata Kunci*: Depresi, Fine-Tuning, LLM, Prompt Engineering, QLoRA.*

I. PENDAHULUAN

Gangguan penyakit depresi merupakan masalah kesehatan mental yang paling sering dijumpai dan berdampak di seluruh dunia. Menurut data yang diberikan oleh *World Health Organization* (*WHO*) pada tahun 2023, terdapat lebih dari 300 juta orang mengidap depresi, terutama pada remaja berumur 15 tahun ke atas yang dimana kasus tersebut selalu meningkat [1]. Penelitian lain juga menyimpulkan bahwa depresi yang tidak terdiagnosis dengan optimal terbilang sangat banyak, yang dimana hal tersebut akan mengarah pada akibat jangka panjang, seperti kualitas hidup yang menurun hingga resiko bunuh diri [2]. Diperkirakan sekitar 3,8% dari populasi dunia mengalami depresi, yang dimana 5% nya orang dewasa, dan 5,7% di dapati orang dewasa yang berusia lebih dari 60 tahun [3].

Media sosial telah berubah menjadi instrumen utama dalam membangun hubungan dan berbagi informasi selama satu dekade terakhir [4]. Hingga saat ini, media sosial seringkali digunakan untuk menunjukkan eksistensi diri, yang meleburkan batas antara realitas dan dunia maya. Seiring kemajuan teknologi, akses media kini menjadi kebutuhan primer bagi masyarakat di seluruh dunia untuk memperoleh informasi, edukasi, dan hiburan secara global dengan mudah [5]. Menurut penelitian [6], orang yang mengalami stres dan depresi sering kali menggunakan platform media sosial untuk berbagi pikiran, emosi, perasaan melalui unggahan atau komentar dengan pengguna lain. Konten yang dibagikan pengguna dapat mengandung indikator-indikator psikologis yang dapat mencerminkan status kesehatan mental mereka. Karena informasi media sosial sangat membantu untuk mengidentifikasi orang-orang yang

berisiko mengalami depresi atau gangguan mental lainnya [7], berbagai arsitektur model *machine learning* dan sistem otomatis berhasil dikembangkan, dimana hal tersebut mampu mengenali karakteristik dan gejala penyakit mental.

Sebagai contoh, pada penelitian [8] yang mengintegrasikan metode *Support Vector Machine* (*SVM*) dengan teknik *SHAP* untuk mendeteksi stres dan depresi pada unggahan Twitter, menghasilkan akurasi yang tinggi sebesar 96,44% serta memberikan transparansi mengenai fitur kata yang paling berpengaruh. Sementara itu, pendekatan *Deep Learning* menggunakan metode *Bidirectional LSTM* terbukti efektif dalam menangkap konteks kalimat dua arah untuk mendeteksi depresi dan kecemasan dengan capaian akurasi sebesar 94,12%, yang lebih unggul dibandingkan model *LSTM* standar maupun algoritma tradisional [9]. Selain itu, studi komparatif [10] menemukan bahwa arsitektur *Convolutional Neural Network* memiliki performa terbaik dengan akurasi 94%, sedikit mengungguli *LSTM* dan secara signifikan melampaui *Naive Bayes*.

Namun seiring perkembangan teknologi, *Large Language Model* (*LLM*) lahir sebagai salah satu algoritma *Artificial Intelligence* yang dirancang untuk mengidentifikasi, menerjemahkan, memprediksi, dan menghasilkan konten baru [11]. Model ini dilatih menggunakan korpus teks yang sangat besar dan dirancang untuk melakukan berbagai macam tugas secara luas. Beberapa contoh model terkenal yang mengimplementasikan *LLM* adalah *GPT* oleh *OpenAi*, *Gemini* oleh *Google* dan *Llama* oleh *META* [12]. *Llama* atau *Large Language Model Meta AI* adalah model yang dirancang untuk menjadi *open-source*, efisien, serta dapat diadaptasi atau dikembangkan untuk berbagai aplikasi, termasuk di bidang medis, sains, dan teknologi. *Llama 3.1* adalah versi lanjutan dari seri *Llama 3* yang dirilis pada Juli 2024 dengan dukungan *context window* hingga 128.000 *token*, memungkinkan model untuk memproses dokumen panjang atau percakapan tanpa kehilangan konteks.

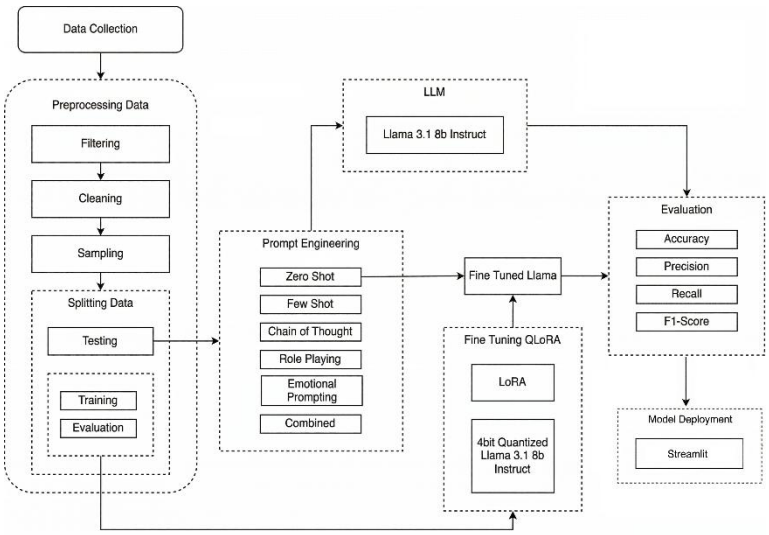
Penelitian terkait penggunaan *LLM* untuk klasifikasi penyakit mental sudah pernah dilakukan [13], yang menyoroti bahwa integrasi *LLM* pada klasifikasi status kesehatan mental masih cukup efektif dan efisien, namun terdapat kelemahan bahwa *LLM* tanpa penyesuaian khusus memiliki akurasi yang rendah. Serta kebutuhan sumber daya komputasi yang besar menjadi tantangan dalam menggunakan *LLM*, sehingga dibutuhkan sebuah cara untuk meningkatkan kemampuan dari *LLM* dalam klasifikasi. *Fine-tuning* adalah pendekatan yang efektif untuk meningkatkan kinerja dari *LLM*, terutama saat dihadapkan pada dataset baru maupun untuk memperbaiki perilaku atau pengetahuan *LLM* serta menghilangkan respons yang tidak diinginkan [14]. Meskipun demikian, kelemahan utama dari *full fine-tuning* saat ini adalah kebutuhan memori dan biaya komputasi yang sangat tinggi [15]. Untuk mengatasi kendala tersebut, berbagai pendekatan telah dikembangkan untuk mengurangi ukuran model *pre-trained* saat proses *fine-tuning*, contohnya seperti *Low-Rank Adaptation* (*LoRA*) dan *Quantized Low-Rank Adaptation* (*QLoRA*) [16]. *QLoRA* sendiri merupakan sebuah metode *fine-tuning* yang menggabungkan proses kuantisasi dan *LoRA*. Tujuannya adalah untuk mengurangi penggunaan memori secara drastis selama proses *fine-tuning* tanpa mengorbankan akurasi atau kinerja [17]. Hal ini secara signifikan meningkatkan aksesibilitas untuk melatih model-model besar bagi peneliti atau tim yang tidak memiliki sumber daya komputasi skala besar [18].

Prompt engineering merupakan sebuah ilmu dalam merancang serta menyusun input atau instruksi secara efektif agar *LLM* dapat menghasilkan output yang diinginkan dengan meliputi berbagai metode, mulai dari *prompt* sederhana hingga kompleks [19]. Penelitian [20] menunjukkan bahwa *prompt engineering* dapat secara signifikan meningkatkan kemampuan *LLM* dalam deteksi stres di media sosial, di mana teknik ini mampu meningkatkan akurasi model hingga 17% dan mengurangi *false positive* sebesar 80% tanpa perlu *fine-tuning*. Kebaruan dari penelitian ini dibandingkan dengan arsitektur dan metode-metode sebelumnya terletak pada penggunaan *LLM* untuk klasifikasi depresi, yang berbeda dengan model tradisional yang memerlukan desain arsitektur saraf yang spesifik. Dan penelitian ini mengeksplorasi kemampuan model tersebut melalui mekanisme *prompt engineering* dan *fine-tuning* menggunakan *QLoRA*, yang memungkinkan performa tinggi pada sumber daya komputasi terbatas.

Berdasarkan pendahuluan serta uraian latar belakang di atas, penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam pengembangan sistem deteksi indikasi depresi berbasis *LLM* yang akurat dan efisien dengan menyajikan analisis komparatif antara teknik *prompt engineering* dan metode *fine-tuning QLoRA* pada model *Llama 3.1-8B* dan menghasilkan evaluasi mendalam terhadap metode yang digunakan untuk menemukan instruksi yang efektif dalam mendeteksi depresi pengguna media sosial.

II. METODE

Penelitian ini dilakukan secara sistematis untuk menganalisis efektivitas strategi *Prompt Engineering* dan pengaruh teknik *Fine-Tuning* pada penggunaan *LLM* dalam mendeteksi depresi pada data berbasis teks dari media sosial. Tahapan penelitian dimulai dari *data collection*, *preprocessing data*, implementasi *Prompt Engineering*, Inferensi pada *LLM* dan *Fine-Tuning QloRA* menggunakan model *Llama-3.1-8b-Instruct*, *evaluation* dan *Model Deployment*. Seluruh tahapan tesebut di kerjakan pada platform *Kaggle* dengan konfigurasi *accelerator GPU T4* 15GB x2. Alur kerja penelitian digambarkan dalam Gambar 1 berikut:



Gambar 1. *Research Workflow* untuk Analisis *Fine-tuning* dan *Prompt Engineering* pada Deteksi Depresi

A. Data Collection

Data yang digunakan dalam penelitian ini merupakan dataset publik yang digunakan pada penelitian terdahulu, yakni *Depression Severity Levels Dataset* [21]. Korpus ini dibangun dari data media sosial melalui mekanisme *web crawling* pada platform media sosial *Twitter* yang berjumlah 41.873 *records*. Variabel utama dalam dataset ini mencakup kolom *text,* yang berisi narasi atau curahan hati pengguna terkait gejala depresi, keputusasaan, dan kondisi mental yang dialaminya. Rincian informasi mengenai atribut pada dataset disajikan pada Tabel 1.

TABEL 1 INFORMASI ATRIBUT DATASET

Atribut	Deskripsi
text	*Raw text* dari unggahan media sosial.
label	Label yang terdiri dari empat kategori: *Minimum, Mild, Moderate,* dan *Severe*

Dataset ini memuat informasi berupa label tingkat keparahan sebagai variabel target. Kolom *label* mengklasifikasikan setiap teks ke dalam empat tingkatan depresi, yaitu *Minimum, Mild, Moderate,* dan *Severe*. Proses kategorisasi tingkat keparahan depresi dilakukan dengan menggunakan *BDI-3* dan *Depression Severity Annotation Schema* (*DSAS*) sebagai acuan. Berdasarkan standar tersebut, setiap entri teks diklasifikasikan ke dalam salah satu dari empat kelas keparahan. Setiap kelas menandakan rentang skor tertentu yang mengindikasikan keparahan gejala depresi yang dialami. Namun, dalam konteks eksperimen penelitian ini, fokus analisis akan dipusatkan pada klasifikasi biner dengan menyaring data hanya pada kelas *Minimum* dan *Severe*. Pemilihan dua kelas ini didasarkan pada strategi untuk meminimalkan ambiguitas fitur pada kategori menengah yakni *Mild* dan *Moderate*, sehingga efektivitas teknik *Prompt Engineering* dan *Fine-tuning* dapat diukur secara lebih objektif. Pendekatan ini memungkinkan penelitian untuk mengekstraksi fitur bahasa yang membedakan antara kondisi sehat dan depresi berat. Meskipun demikian, penyederhanaan ini menimbulkan potensi bias di mana model mungkin menjadi terlalu sensitif terhadap indikator depresi berat dan memiliki keterbatasan generalisasi dalam mengidentifikasi depresi pada tingkat menengah dalam skenario dunia nyata.

B. Preprocessing Data

Pada tahap ini, dilakukan beberapa proses seperti *Filtering*, *Cleaning*, *Sampling*, dan *Splitting Data* agar menghasilkan data yang bersih dan siap untuk diimplementasi teknik-teknik yang akan diuji.

1) Filtering: Data filtering adalah proses seleksi variabel yang bertujuan menghapus variabel tidak relevan untuk menyederhanakan model, mengurangi waktu pelatihan, dan mengatasi masalah dimensi tinggi pada data [22]. Pada penelitian ini, *filtering* dilakukan untuk mengurangi bobot klasifikasi, yang tadinya dataset merupakan *multi-class* akan dirubah menjadi *binary*. Dataset asli memuat empat label tingkat keparahan*: Minimum, Mild, Moderate*, dan *Severe*. Data dengan label *'Mild'* dan *'Moderate'* akan di hapus dari dataset. Langkah ini diambil untuk mempertegas keputusan yang dihasilkan model dan meningkatkan sensitivitas pada pendeteksian kondisi antara individu dengan indikasi depresi yang minim atau tidak ada dan individu dengan tingkat depresi berat.

TABEL 1
DISTRIBUSI DATA SETELAH FILTERING

Label	Jumlah Data	Total
minimum	10.556	21.732
severe	11.176	

2) Cleaning: Selanjutnya dilakukan proses pembersihan data untuk meningkatkan kualitas data sebelum di proses [23]. Proses ini melibatkan penghapusan data yang memiliki atribut teks kosong atau *missing value*. Langkah ini bertujuan untuk mencegah kegagalan teknis pada saat proses tokenisasi.

3) Sampling: Mengingat besarnya ukuran dataset asli dan keterbatasan sumber daya komputasi, penelitian ini menerapkan teknik *sampling*. *Sampling* merupakan proses memilih sebagian kecil data dari seluruh data untuk dianalisis dengan tujuan agar proses dapat dijalankan lebih cepat dan hemat memori tanpa mengorbankan kualitas hasil secara signifikan [24]. Dataset diacak dan kemudian diambil subset sebanyak 5.000 data pertama.

TABEL 2
DISTRIBUSI DATA SETELAH SAMPLING

Label	Jumlah Data	Total
minimum	2.430	5.000
severe	2.570	

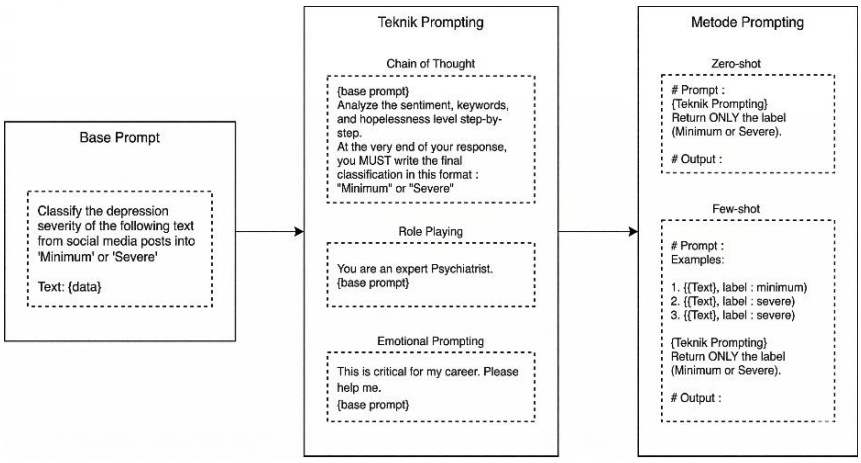
4) Splitting Data: Dataset yang sudah melalui tahap *cleaning* dan sudah disampel kemudian akan dilakukan proses *splitting data* dengan membagi dataset menjadi 3 himpunan bagian, yaitu data latih, data validasi, dan data uji. Tahap pertama pada *splitting data* adalah memisahkan 10% dari total data untuk *test set*, kemudian sisanya akan dibagi lagi dengan proporsi 10% untuk *eval set* dan 90% untuk *train set*. Sehingga diperoleh tiga himpunan data yang siap digunakan pada proses selanjutnya dengan masing-masing jumlah data tiap subset dijelaskan di Tabel 4 di bawah.

TABEL 3
DISTRIBUSI DATA SETELAH SPLITTING

Subset	Jumlah Data	Total
Train set	4.050	
Eval set	450	5.000
Test set	500	

C. Implementasi Prompt Engineering

Dalam skenario pertama, penelitian ini menerapkan teknik *Prompt Engineering* untuk mengevaluasi kemampuan model *Llama-3.1-8B-Instruct* dalam mengklasifikasikan tingkat keparahan depresi. Sebelum dilakukan proses inferensi oleh model, data teks mentah dikonversi menjadi struktur *prompt* sesuai dengan strategi yang dirancang. Kerangka penerapan *prompt engineering* dapat dilihat pada Gambar 2 dibawah:

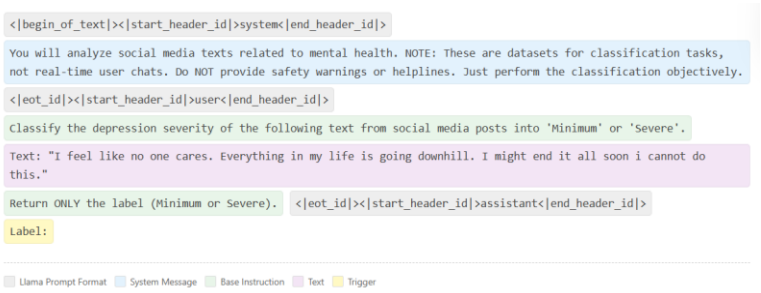


Gambar 2. Kerangka Implementasi *Prompt Engineering*

Format *prompting* untuk *Llama 3.1* menggunakan struktur percakapan berbasis teks yang mengandalkan token spesial untuk membedakan peran dan batasan giliran [25]. Setiap urutan input diawali dengan token <|*begin_of_text*|>, diikuti oleh blok pesan yang masing-masing memiliki *header* spesifik, *system* untuk instruksi dasar, *user* untuk input pengguna, dan *assistant* untuk respons model. Setiap *header* wajib dibungkus dengan <|*start_header_id*|> dan <|*end_header_id*|>, yang di setiap giliran ditutup dengan token <|*eot_id*|>. Kemudian diakhiri dengan *header*

assistant, yang berfungsi sebagai *trigger* bagi model untuk menghasilkan teks *output*. Pada penelitian ini menguji 6 teknik *prompt engineering* antara lain sebagai berikut:

1) Zero-shot: Merupakan teknik *prompting* di mana *LLM* diberikan instruksi untuk melakukan tugas yang diinginkan tanpa diberikan contoh input maupun output yang spesifik. Pendekatan ini menguji kemampuan model dalam memahami instruksi klasifikasi secara langsung hanya berdasarkan pengetahuannya [26]. Gambar 3 berikut adalah contoh hasil dari implementasi *prompting* yang sudah disesuaikan dengan format *Llama 3.1* menggunakan metode *Zero-shot*.



Gambar 3. Format *Prompt Zero-shot*

2) Few-shot: Strategi *Few-shot* dijalankan dengan membuat konteks *prompting* lebih spesifik dengan menyertakan tiga contoh teks beserta labelnya sebagai referensi. Studi [26] menunjukkan bahwa *Few-shot* secara konsisten meningkatkan akurasi dibandingkan *Zero-shot*, terutama pada tugas-tugas yang kompleks atau ambigu. Pemberian contoh atau referensi ini bertujuan untuk membuat model memahami mekanisme *in-context learning*, sehingga model dapat mengenali pola distribusi label dan gaya bahasa yang diharapkan dengan lebih akurat. Detail *prompt* menggunakan *Few-shot* dijabarkan pada Gambar 4.

Gambar 4. Format *Prompt Few-shot*

3) Role Playing: Merupakan teknik *prompting* di mana model diarahkan untuk mengambil peran atau persona tertentu yang relevan dengan tugas yang diberikan [27]. Langkah ini dilakukan memadukan *base instruction Zero-shot* kemudian memberikan persona kepada model sebagai seorang ahli psikiater yang berspesialisasi dalam analisis depresi. Teknik ini akan mengarahkan model agar memiliki sudut pandang profesional dan menggunakan dasar pengetahuan medis yang relevan saat menganalisis indikator depresi pada teks.

4) Emotional Prompting: Metode ini juga memadukan *base instruction Zero-shot* dengan menambah ekspresi emosional seperti ke dalam instruksi sistem. Penambahan emosi ini didasarkan pada penelitian [28], bahwa kondisi emosional dapat meningkatkan kinerja model untuk memberikan prediksi yang lebih presisi dan berhati-hati.

5) Chain of Thought: Metode ini dijalankan dengan *base instruction Zero-shot* yang menginstruksikan model untuk tidak langsung menjawab, melainkan melakukan *step-by-step reasoning* dengan menganalisis sentimen, kata kunci, dan tingkat keputusasaan terlebih dahulu. Dengan memecah proses inferensi menjadi langkah-langkah logis, teknik ini bertujuan untuk meminimalisir kesalahan klasifikasi akibat pemahaman konteks yang minim dan memaksa

model untuk berpikir sebelum menyimpulkan [29]. Karena model menghasilkan teks penalaran yang panjang sebelum memberikan jawaban, label klasifikasi akhir didapatkan melalui mekanisme *parsing* yang memindai pola format khusus di bagian akhir respons model.

6) Combined Strategy: Pendekatan ini menggabungkan metode *Few-shot* dengan 3 teknik *prompting* sebelumnya ke dalam satu struktur *prompt*. Kombinasi ini dirancang untuk mengevaluasi apakah penerapan seluruh elemen *prompting* sebelumnya dapat menciptakan metode yang membuat kinerja model lebih baik dan efisien. Serupa dengan strategi *CoT*, output final dari strategi ini diperoleh dengan mekanisme *parsing*.

D. Inferensi LLM

Penelitian ini menggunakan model yang dikembangkan oleh *META* yaitu *Llama-3.1-8B-Instruct* sebagai *base model*. Varian *Instruct* secara spesifik digunakan karena model telah dioptimasi atau dilatih untuk mengikuti instruksi kompleks. Lalu varian *8b* dipilih untuk mengoptimalkan penggunaan sumber daya komputasi yang terbatas pada platform *Kaggle Notebooks*. Model dipanggil dengan memanfaatkan *library Transformers* tanpa dikuantisasi, lalu proses prediksi pada data yang sudah diberi *prompt* dilakukan pada *pipeline* dan output nantinya akan dievaluasi.

E. Fine-Tuning QLoRA

Pada skenario kedua, penelitian ini menerapkan teknik *Quantized Low-Rank Adaption* (*QLoRA*). *QLoRA* dipilih sebagai metode *fine-tuning* karena efisiensinya dalam mengurangi kebutuhan memori namun tetap mempertahankan kualitas model yang setara dengan *full fine-tuning* [17]. Kemudian setelah model yang telah berhasil melalui proses *fine-tuning* akan diinferensikan pada *Test Set* yang sama pada skenario sebelumnya dan dievaluasi hasilnya. Pada tahap ini akan dilakukan beberapa konfigurasi antara lain:

1) Konfigurasi Kuantisasi Model: Untuk mengatasi keterbatasan sumber daya komputasi, *base model* dimuat menggunakan konfigurasi kuantisasi 4-bit melalui *library BitsAndBytes*. Proses ini memampatkan bobot model dari presisi standar 16-bit menjadi format *4-bit NormalFloat* (*NF4*). Konfigurasi spesifik yang diterapkan dijelaskan pada Tabel 5 dibawah:

TABEL 4
KONFIGURASI KUANTISASI MODEL MENGGUNAKAN BITSANDBYTES

Konfigurasi *BitsAndBytes*	
Load in 4-bit	*True*
Quantization Type	*nf4*
Compute Dtype	*float16*

2) Konfigurasi LoRA: Penelitian ini menyisipkan modul *adaptor Low-Rank* dengan menggunakan *library peft* dari *HuggingFace*. Konfigurasi *LoraConfig* yang digunakan dijelaskan pada Tabel 6 sebagai berikut:

TABEL 5
KONFIGURASI PARAMETER LORA

Konfigurasi *LoraConfig*	
Rank (r)	*64*
Alpha (α)	*16*
lora_dropout	*0*
Target Modules	*[q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj]*
Bias	*"none"*

3) Training Hyperparameter: Proses pelatihan dilakukan menggunakan *SFTTrainer* dari *library trl*. Parameter pelatihan diatur menggunakan *SFFTConfig* untuk menyeimbangkan kecepatan dan keseimbangan model yang di-*fine-tuning*. Penjelasan mengenai konfigurasi *hyperparameter* dijelaskan pada Tabel 7 dibawah.

TABEL 6
KONFIGURASI TRAINING HYPERPARAMETER

Konfigurasi *SFFTConfig*	
Train Epochs	*1*
Optimizer	*paged_adamw_32bit*
Learning Rate	*2e-4*
Batch Size	*1*
Gradient Accumulation Steps	*8*
Max Sequence Length	*512 token*
Warmup Ratio	*0.03*
Weight Decay	*0.001*

F. Evaluasi

Kinerja dari setiap model akan dievaluasi menggunakan *confusion matrix*. Evaluasi performa model merupakan langkah penting yang bertujuan untuk mengukur keberhasilan prediksi model terhadap data uji dan menganalisis kinerjanya secara menyeluruh [8]. Beberapa matriks evaluasi tersebut antara lain:

1) Accuracy: Nilai akurasi merupakan perbandingan antara jumlah prediksi yang benar dengan total keseluruhan data. Metrik ini memberikan gambaran umum tentang seberapa sering model membuat prediksi yang benar secara keseluruhan. Nilai akurasi dapat diperoleh dengan persamaan (1) berikut:

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

2) Precision: Menggambarkan tingkat ketepatan model dari seluruh data yang diklasifikasikan sebagai positif. Metrik ini menjawab pertanyaan "Dari semua prediksi yang menyatakan positif, berapa persen yang sebenarnya benar-benar positif?" [13]. Presisi menjadi sangat penting dalam kasus ini di mana *False Positive* harus diminimalkan. Presisi dapat diperoleh dengan persamaan (2) berikut:

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

3) Recall: Metrik ini mengukur kemampuan model untuk menemukan kembali semua data yang seharusnya positif. Dan *recall* memastikan berapa persen hasil yang teridentifikasi positif sebenarnya oleh model [30]. *Recall* dihitung dengan persamaan (3) berikut:

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

4) F1-Score: Merupakan nilai rata-rata dari Presisi dan *Recall*, yang memberikan pandangan yang seimbang antara kedua metrik tersebut [13]. *F1-Score* dihitung dengan persamaan (4) berikut:

$$\boldsymbol{F1-Score} = \frac{\boldsymbol{2TP}}{\boldsymbol{2TP+FP+FN}} \quad (4)$$

G. Model Deployment

Tahap akhir penelitian ini difokuskan pada skenario *model deployment* untuk mendemonstrasikan model *LLM* pada penelitian ini dalam sebuah aplikasi terapan. Implementasi ini menggunakan *framework Streamlit* untuk membangun antarmuka web yang interaktif, sementara seluruh proses komputasi dan inferensi model tetap dijalankan pada infrastruktur *platform Kaggle* guna mengoptimalkan penggunaan akselerasi *GPU T4*. Agar sistem dapat diakses secara publik melalui jaringan internet, protokol *tunneling* melalui layanan *Ngrok* diterapkan untuk menjembatani koneksi antara lingkungan *server Kaggle* dengan perangkat lokal pengguna. Melalui skenario arsitektur ini, model mampu menerima input teks media sosial secara dinamis dan memberikan prediksi klasifikasi tingkat keparahan depresi secara *real-time*.

III. HASIL DAN PEMBAHASAN

Berdasarkan hasil pengujian yang telah dilakukan terhadap model *Llama-3.1-8B-Instruct* menggunakan berbagai strategi *Prompt Engineering* dan metode *Fine-Tuning* menggunakan *QLoRA*, didapatkan beberapa kesimpulan terkait performa model dalam mendeteksi depresi. Pada bagian ini, hasil yang diperoleh akan dibahas lebih lanjut dengan mengacu pada hasil evaluasi strategi *prompting* dan hasil evaluasi model setelah dilakukan *fine-tuning*.

A. Hasil Evaluasi Prompt Engineering

Pada skenario pertama, evaluasi dilakukan untuk mengukur efektivitas *base model Llama 3.1-8b-Instruct* dengan menerapkan metode *prompt engineering* pada *Test set* yang berjumlah 500 data. Nilai *precision*, *recall*, dan *f1-score* pada hasil evaluasi diambil dari nilai *average* kedua kelas. Tabel 8 di bawah merangkum kinerja dari enam strategi yang diuji.

TABEL 7
HASIL EVALUASI PROMPT ENGINEERING MENGGUNAKAN LLAMA 3.1-8B-INSTRUCT

Metode	*Accuracy*	*Precision*	*Recall*	*F1-Score*
Zero-Shot	0.81	0.83	0.81	0.81
Few-Shot	0.83	0.84	0.84	0.83
RP	0.83	0.85	0.82	0.82
EP	0.82	0.84	0.82	0.82
CoT	**0.84**	**0.85**	**0.84**	**0.84**
Combined	0.83	0.85	0.83	0.83

Berdasarkan Tabel 8, terlihat adanya perbedaan performa antar metode meskipun tidak signifikan. Namun hal ini tetap menegaskan bahwa *prompt engineering* memiliki pengaruh dalam efektivitas deteksi. Model mendapatkan akurasi 0.81 yang dimana sudah cukup baik meskipun hanya dengan metode *Zero-shot*. Performa terbaik pada skenario ini diraih oleh metode *Chain of Thought* dengan akurasi 0.84 dan *F1-Score* rata-rata 0,84. Namun hasil tersebut perlu dianalisis kembali dengan melihat kemampuan model pada masing-masing kelas. Perlu diketahui jumlah data setiap kelas yang di uji pada metode ini berjumlah 243 untuk kelas *Minimum* dan 257 untuk kelas *Severe* yang menandakan data uji ini bersifat *balanced*. Performa masing-masing metode per-kelas dapat dilihat di Tabel 9.

TABEL 8
PERFORMA METODE PER-KELAS

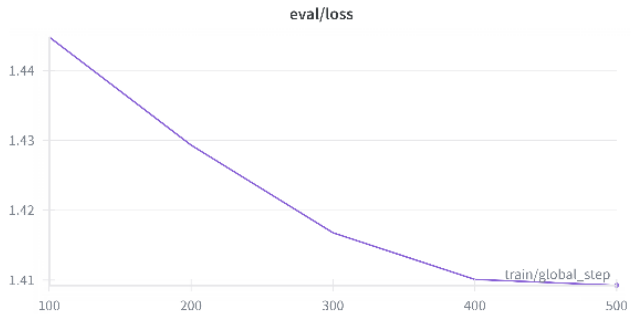
Metode	Label	*Precision*	*Recall*	*F1-Score*
Zero-Shot	*Minimum*	0.91	0.68	0.78
	Severe	0.76	0.94	0.84
Few-Shot	*Minimum*	0.80	0.88	0.84
	Severe	0.88	0.79	0.83
RP	*Minimum*	0.92	0.70	0.80
	Severe	0.77	0.95	0.85
EP	*Minimum*	0.93	0.68	0.79
	Severe	0.76	0.95	0.84
CoT	*Minimum*	0.89	0.77	0.83
	Severe	0.81	0.91	0.86
Combined	*Minimum*	0.78	0.93	0.85
	Severe	0.92	0.73	0.81

Analisis terhadap metrik performa per kelas memperoleh hasil yang signifikan antara sensitivitas model terhadap kelas *Minimum* dan *Severe* yang sangat dipengaruhi oleh teknik *prompting*. Metode *Zero-Shot*, *RP*, dan *EP*, menunjukkan pola bias ke arah kelas *Severe* dengan nilai *Recall* 0.94 dan 0.95. Yang dimana sangat dominan pada kelas *Severe*, namun mengalami penurunan drastis pada *Recall* kelas *Minimum* yang hanya berkisar di angka 0,68 dan 0,70. Menandakan bahwa penambahan elemen emosional atau peran spesifik membuat model menjadi sensitif dalam mendeteksi gejala berat, namun kurang dalam menangkap depresi ringan. Sebaliknya, metode yang menyertakan contoh atau referensi seperti *Few-Shot* dan strategi *Combined* (*FS+RP+EP+CoT*) berhasil mengurangi bias tersebut dengan meningkatkan kemampuan deteksi kelas *Minimum* secara signifikan, namun *Recall* kelas *Severe* turun hingga 0,73 pada metode *Combined*. Hal ini menunjukkan adanya perubahan pemikiran model yang lebih condong ke kelas *Minimum* akibat *prompt* yang lebih panjang dan kompleks. Di sisi lain, metode *CoT* mampu mempertahankan *Recall* kelas *Severe* tetap tinggi di angka 0,91 sekaligus memperbaiki deteksi kelas *Minimum* menjadi 0,77 yang menjadikannya strategi paling efektif pada skenario ini.

B. Fine-Tuning QLoRA

Pada skenario kedua, penelitian ini menerapkan teknik *fine-tuning* menggunakan metode *QLoRA* pada model *Llama-3.1-8B-Instruct*. Tujuannya adalah untuk mengadaptasi bobot model agar memiliki pengetahuan yang spesifik dalam mendeteksi teks depresi dari media sosial. Proses ini dilakukan dengan melatih model dengan *Train Set* (4.050 data) data dan *Eval Set* (450 data). Untuk memastikan bahwa model benar-benar belajar dan menghindari *overfitting*, dilakukan pemantauan terhadap metrik *Training Loss* dan *Validation Loss* secara berkala menggunakan *Wandb*.

Gambar 5. Grafik *Training Loss*



Gambar 6. Grafik *Validation Loss*

Selama proses pelatihan, terlihat kurva dari Gambar 5 dan 6 memperlihatkan pola yang positif, meskipun *Training Loss* mengalami fluktuasi tajam akibat karakteristik metode *QLoRA* yang memperbarui parameter secara parsial dengan *batch size* kecil, grafik *Validation Loss* justru menunjukkan tren penurunan yang konsisten dan stabil dari 1.4448 menjadi 1.4093. Hal ini secara tegas mengonfirmasi bahwa model berhasil mencapai konvergensi yang optimal dalam mempelajari fitur depresi yang baik tanpa mengalami *overfitting*.

C. Hasil Evaluasi Model Fine-Tuned

Setelah dilakukan proses *fine-tuning* menggunakan metode *QLoRA* pada model *Llama-3.1-8B-Instruct*, dilakukan evaluasi kembali menggunakan data uji yang sama. Model *fine-tuned* akan diuji hanya dengan metode *Zero-shot* untuk mengukur kualitas dari pengetahuan model setelah di *training*. Hasil pengukuran performa disajikan dalam Tabel 10 di bawah ini.

TABEL 9
HASIL EVALUASI FINE TUNED LLAMA 3.1-8B-INSTRUCT

Metrik	*Minimum*	*Severe*	*Weighted Avg*
Precision	0,91	0.94	0.92
Recall	0.93	0.92	0.92
F1-Score	0.92	0.93	0.92
Accuracy		0.92	

Hasil pada Tabel 10 menunjukkan lonjakan kinerja yang signifikan dengan akurasi mencapai 0.92, jauh melampaui metode prompting terbaik (*CoT*) yang hanya mencapai 0.84. Aspek yang lebih krusial dari peningkatan ini adalah keseimbangan performa antar kelas, di mana metode *Fine-Tuning* berhasil mengurangi bias yang sebelumnya sering muncul pada metode *prompting*. Hal ini terlihat dari kemampuan model mendeteksi gejala depresi ringan pada kelas *Minimum* dengan sangat baik dengan *Recall* 0.93, serta capaian *Precision* 0,94 pada kelas *Severe* yang menandakan minimnya *False Positive* saat mendeteksi depresi.

D. Analisis Perbandingan

Untuk menemukan metode yang paling efektif dari kedua strategi yang sudah berhasil dijalankan, akan dilakukan analisis perbandingan hasil terbaik dari skenario *Prompt Engineering* dengan hasil dari skenario *Fine-Tuning*. Hasil dari masing-masing evaluasi metode terbaik tiap skenario akan dijelaskan pada Tabel 11 dibawah.

TABEL 10
PERBANDINGAN PERFORMA PROMPT ENGINEERING DAN FINE-TUNING

Metode	*Accuracy*	*Precision*	*Recall*	*F1-Score*
Prompt Engineering (CoT)	0.84	0.85	0.84	0.84
Fine-Tuning (QLoRA)	**0.92**	**0.92**	**0.92**	**0.92**

Merujuk pada Tabel 11, metode *Fine-Tuning* menunjukkan peningkatan yang signifikan hingga 8% di seluruh metrik evaluasi. Meskipun metode terbaik *Prompt Engineering* seperti *Chain of Thought* terbukti mampu mendorong kemampuan *base model* melalui instruksi saja, metode ini masih memiliki keterbatasan fundamental. *Prompting* hanya mengubah cara model memproses informasi yang ada di satu percakapan tanpa mengubah parameter atau pengetahuan model. Sehingga pemahaman model terhadap teks depresi masih bergantung pada kualitas instruksi. Sebaliknya, lonjakan akurasi model hingga 0.92 pada *Fine-Tuning* mengindikasikan bahwa model telah berhasil mempelajari pola linguistik dataset secara mendalam. Proses adaptasi bobot melalui *QLoRA* memungkinkan model untuk menangkap unggahan pada sosial media yang sering kali terlewatkan oleh model.

Untuk mengevaluasi posisi dan performa model *Llama 3.1-8B* yang telah di-*fine-tune*, dilakukan perbandingan dengan beberapa penelitian terdahulu yang menggunakan arsitektur *Machine Learning* dan *Deep Learning* konvensional pada domain deteksi depresi. Ringkasan perbandingan hasil dapat dilihat pada Tabel 12.

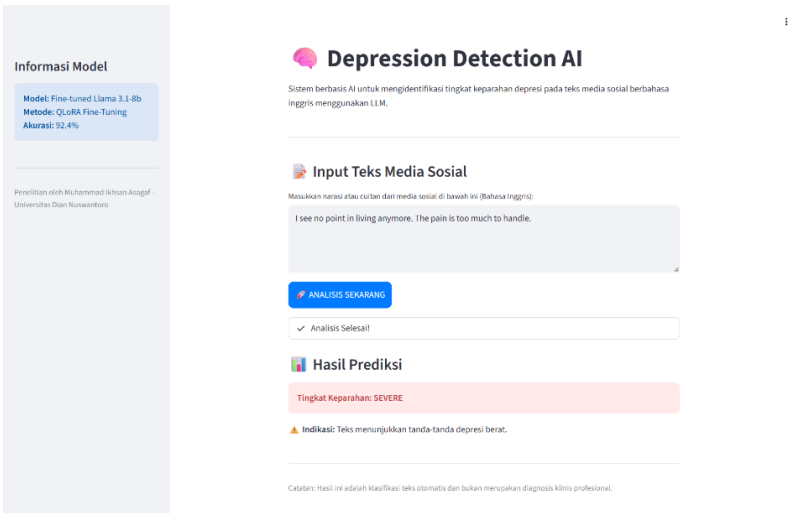
TABEL 11
PERBANDINGAN PERFORMA DENGAN PENELITIAN TERDAHULU

Metode	Akurasi
SVM + SHAP [8]	96.44
Bi-LSTM [9]	94.12
CNN [10]	94.00
Fine-Tuned Llama 3.1-8b (QLoRA)	92.40

Meskipun capaian akurasi *Llama 3.1-8b* sebesar 92,4% sedikit di bawah performa *SVM*, *Bi-LSTM*, dan *CNN*, penggunaan *LLM* menawarkan keunggulan fundamental dalam pemahaman kontekstual yang lebih kompleks. Perlu dipertimbangkan bahwa variasi performa ini juga dipengaruhi oleh perbedaan karakteristik dan volume dataset yang digunakan pada masing-masing penelitian, sehingga hasil akurasi tidak dapat diperbandingkan secara absolut. Namun, implementasi teknik *QLoRA* tetap membuktikan efisiensi tinggi pada sumber daya terbatas. Dengan demikian, hasil ini tetap sangat kompetitif karena mengombinasikan presisi yang kuat dengan *generative AI* yang jauh lebih kompleks dan fleksibel dibandingkan algoritma tradisional.

E. Model Deployment menggunakan Streamlit

Implementasi model ke dalam aplikasi berbasis *web* dilakukan untuk menguji validitas sistem dalam skenario penggunaan riil serta membuktikan kemampuan terapan dari model *fine-tuned* yang telah dikembangkan. Pengujian ini menggunakan *framework Streamlit* seperti yang terlihat pada Gambar 7 dibawah.



Gambar 7. Tampilan *UI* Hasil *Deployment* menggunakan *Streamlit*

Proses inferensi pada aplikasi ini berjalan secara *real-time* dengan memanfaatkan *LoRA adapter* yang telah dilatih sebelumnya. Saat pengguna menekan tombol analisis, sistem akan memproses teks melalui rangkaian *prompt* yang telah dioptimasi untuk menghasilkan klasifikasi tingkat keparahan depresi. Hasil prediksi ditampilkan melalui elemen visual berbasis warna merah untuk kategori *Severe* dan hijau untuk kategori *Minimum*.

Dari hasil evaluasi dan implementasi diatas, penelitian ini membuktikan tingkat keberhasilan sistem dalam memenuhi tujuan utama penelitian untuk menyediakan instrumen deteksi dini depresi yang presisi dan reliabel. Faktor kunci keberhasilan ini didorong oleh efektivitas teknik *QLoRA* dalam mengoptimasi parameter model secara spesifik, meskipun efisiensinya masih terbatasi oleh ketergantungan pada kapasitas memori *GPU* saat proses inferensi. Oleh karena itu, implementasi sistem ini memiliki batasan teknis pada *max sequence length* sebesar 512 *token* guna menjaga stabilitas penggunaan *VRAM* serta meminimalkan *latency*. Keterbatasan sumber daya komputasi dan volume data tetap menjadi hambatan utama dalam penelitian ini. Meskipun demikian, kontribusi riil dari penggunaan model hasil *fine-tuning* dengan teknik *QLoRA* ini adalah terbukanya peluang integrasi *LLM* skala besar ke dalam ekosistem *telemedicine* dengan biaya infrastruktur yang lebih efisien. Model *generative AI* ini dapat berfungsi sebagai fitur cerdas untuk melakukan *screening* awal secara *real-time* dan masif, sehingga memungkinkan penyedia layanan kesehatan untuk memberikan intervensi medis yang lebih cepat dan tepat sasaran berdasarkan analisis linguistik yang objektif.

IV. SIMPULAN

Setelah dilakukan analisis pada kedua metode ini, dapat disimpulkan bahwa meskipun teknik *Prompt Engineering* khususnya strategi *Chain of Thought* mampu meningkatkan akurasi dasar model hingga 84,4%, metode *Fine-Tuning* menggunakan *QLoRA* terbukti jauh lebih unggul dengan capaian akurasi signifikan sebesar 92,4%. Berdasarkan temuan tersebut, implementasi *Fine-Tuning* dan *Prompt Engineering* sangat direkomendasikan sebagai solusi utama untuk pengembangan sistem deteksi dini depresi yang menuntut presisi dan reliabilitas tinggi. Namun penelitian ini masih terhalang beberapa batasan seperti kurangnya sumber daya komputasi. Sehingga pada pengembangan selanjutnya, disarankan memperluas cakupan data yang digunakan, seperti mengembangkan penelitian dengan dataset bahasa lokal serta meningkatkan sumber daya komputasi sehingga dapat menguji varian model lain yang memiliki parameter dan *knowledge* yang lebih terkini.

DAFTAR PUSTAKA

[1] N. Aisyaroh, I. Hudaya, and R. Supradewi, "Trend Penelitian Kesehatan Mental Remaja di Indonesia dan Faktor yang Mempengaruhi: Literature Review," *Scientific Proceedings of Islamic and Complementary Medicine*, vol. 1, no. 1, pp. 41–51, Aug. 2022, doi: 10.55116/spicm.v1i1.6.

[2] D. Ridha Dwiki Putri, M. Reza Fahlevi, M. Sadikin, R. Utami, and Rizki Fajar Utomo, "Prediksi Tingkat Depresi Remaja Menggunakan Metode Naïve Bayes Classifier: Analisis Faktor Psikologis Dan Lingkungan," *KESATRIA: Jurnal Penerapan Sistem Informasi (Komputer & Manajemen)*, vol. 5, no. 4, pp. 2034–2043, Oct. 2024.

[3] World Health Organization (WHO), "Depressive disorder (depression)," www.who.int. Accessed: Dec. 12, 2025. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/depression#

[4] S. N. Salsabila and T. Ardi Ardani, "Analisis Dampak dan Konsekuensi Penggunaan Media Sosial Terhadap Tingkat Bunuh Diri pada Usia Dewasa Awal: A Systematic Literature Review," *Jurnal Ilmu Psikologi dan Kesehatan (SIKONTAN)*, vol. 3, no. 2, pp. 45–52, Oct. 2024, doi: 10.47353/sikontan.v3i2.1921.

[5] Peldi, Syahruddin, and Asmurti, "Penggunaan Media Sosial Sebagai Representase Gaya Hidup Mahasiswa," *Jurnal Ilmiah Ilmu Sosial dan Pendidikan*, vol. 2, no. 2, pp. 78–83, May 2024.

[6] L. Ilias, S. Mouzakitis, and D. Askounis, "Calibration of Transformer-Based Models for Identifying Stress and Depression in Social Media," *IEEE Trans. Comput. Soc. Syst.*, vol. 11, no. 2, pp. 1979–1990, Apr. 2024, doi: 10.1109/TCSS.2023.3283009.

[7] R. Salas-Zárate, G. Alor-Hernández, M. D. P. Salas-Zárate, M. A. Paredes-Valverde, M. Bustos-López, and J. L. Sánchez-Cervantes, "Detecting Depression Signs on Social Media: A Systematic Literature Review," *Healthcare (Switzerland)*, vol. 10, no. 2, Feb. 2022, doi: 10.3390/healthcare10020291.

[8] Y. Tolla and Kusrini, "Deteksi Stres dan Depresi Unggahan Media Sosial dengan Machine Learning," *Jurnal Fasilkom*, vol. 15, no. 1, pp. 84–92, Apr. 2025, doi: 10.37859/jf.v15i1.9067.

[9] K. Setyo Nugroho, I. Akbar, A. Nizar Suksmawati, and Istiadi, "Deteksi Depresi dan Kecemasan Pengguna Twitter menggunakan Bidirectional Lstm," in *The 4th Conference on Innovation and Application of Science and Technology (CIASTECH 2021)*, 2021, pp. 287–296. doi: 10.31328/ciastech.v0i0.3321.

[10] A. A. Pangestu and P. Akhmad Rezki, "Perbandingan Performa Arsitektur Machine Learning untuk Deteksi Dini Depresi Berbasis Natural Language Processing dalam Bahasa Indonesia," *Journal of Informatics, Information System, and Artificial Intelligence)*, vol. 3, no. 2, pp. 93–104, 2025, doi: 10.24815/j-sign.v3i2.49873.

[11] H. R. Lawrence, R. A. Schneider, S. B. Rubin, M. J. Matarić, D. J. McDuff, and M. Jones Bell, "The Opportunities and Risks of Large Language Models in Mental Health.," *JMIR Ment. Health*, vol. 11, p. e59479, Jul. 2024, doi: 10.2196/59479.

[12] M. A. K. Raiaan *et al.*, "A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges," *IEEE Access*, vol. 12, pp. 26839–26874, 2024, doi: 10.1109/ACCESS.2024.3365742.

[13] T. Kallstenius, A. J. Capusan, G. Andersson, and A. Williamson, "Comparing traditional natural language processing and large language models for mental health status classification: a multi-model evaluation," *Sci. Rep.*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-08031-0.

[14] S. S. Alahmari, L. O. Hall, P. R. Mouton, and D. B. Goldgof, "Repeatability of Fine-Tuning Large Language Models Illustrated Using QLoRA," *IEEE Access*, vol. 12, pp. 153221–153231, 2024, doi: 10.1109/ACCESS.2024.3470850.

[15] A. Mahendra and Styawati, "Implementasi Lowk-Rank Adaptation of Large Languange Model (LORA) Untuk Effisiensi Large Language Model," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, vol. 9, no. 4, pp. 1881–1890, Nov. 2024, doi: 10.29100/jipi.v9i4.5519.

[16] G. Il Kim, S. Hwang, and B. Jang, "Efficient Compressing and Tuning Methods for Large Language Models: A Systematic Literature Review," *ACM Comput. Surv.*, vol. 57, no. 10, pp. 1–39, Oct. 2025, doi: 10.1145/3728636.

[17] A. Shen *et al.*, "Accurate and Efficient Fine-Tuning of Quantized Large Language Models Through Optimal Balance in Adaptation," *Trans. Assoc. Comput. Linguist.*, vol. 13, pp. 861–877, Jul. 2025, doi: 10.1162/TACL.a.23.

[18] Z. Tan, X. Xiong, and D. Xu, "Efficient Differentially Private Fine-Tuning with QLoRA and Prefix Tuning for Large Language Models," *Journal of Computer Science and Artificial Intelligence*, vol. 2, no. 3, pp. 50–54, Mar. 2025, doi: 10.54097/we271q84.

[19] G. Phillips-Wren and A. Håkansson, "Towards Using Prompt Engineering in Large Language Models to Assist Decision Making," *Procedia Comput. Sci.*, vol. 270, pp. 5225–5238, 2025, doi: 10.1016/j.procs.2025.09.650.

[20] N. Esmi, A. Shahbahrami, Y. Nabati, B. Rezaei, G. Gaydadjiev, and P. de Jonge, "Stress detection through prompt engineering with a general-purpose LLM," *Acta Psychol. (Amst).*, vol. 260, p. 105462, Oct. 2025, doi: 10.1016/j.actpsy.2025.105462.

[21] Y. H. P. P. Priyadarshana, Z. Liang, and I. Piumarta, "HelaDepDet: A Novel Multi-class Classification Model for Detecting the Severity of Human Depression," in *Collaboration Technologies and Social Computing: 29th International Conference, CollabTech 2023, Osaka, Japan, August 29–September 1, 2023, Proceedings*, Berlin, Heidelberg: Springer-Verlag, 2023, pp. 3–18. doi: 10.1007/978-3-031-42141-9_1.

[22] M. Cavus and P. Biecek, "Investigating the impact of balancing, filtering, and complexity on predictive multiplicity: A data-centric perspective," *Information Fusion*, vol. 123, p. 103243, Nov. 2025, doi: 10.1016/j.inffus.2025.103243.

[23] A. M. Sharifnia, D. E. Kpormegbey, D. K. Thapa, and M. Cleary, "A Primer of Data Cleaning in Quantitative Research: Handling Missing Values and Outliers," *J. Adv. Nurs.*, Mar. 2025, doi: 10.1111/jan.16908.

[24] S. Wu, X. Zhu, and H. Wang, "Subsampling and Jackknifing: A Practically Convenient Solution for Large Data Analysis With Limited Computational Resources," *Stat. Sin.*, 2023, doi: 10.5705/ss.202021.0257.

[25] Meta, "Llama 3.1 Model Cards and Prompt Formats," llama.com. Accessed: Dec. 12, 2025. [Online]. Available: https://www.llama.com/docs/model-cards-and-prompt-formats/llama3_1/

[26] S. Sivarajkumar, M. Kelley, A. Samolyk-Mazzanti, S. Visweswaran, and Y. Wang, "An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing: Algorithm Development and Validation Study," *JMIR Med. Inform.*, vol. 12, p. e55318, Apr. 2024, doi: 10.2196/55318.

[27] A. Kong *et al.*, "Better Zero-Shot Reasoning with Role-Play Prompting," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2024, pp. 4099–4113. doi: 10.18653/v1/2024.naacl-long.228.

[28] R. Vinay, G. Spitale, N. Biller-Andorno, and F. Germani, "Emotional prompting amplifies disinformation generation in AI large language models," *Front. Artif. Intell.*, vol. 8, May 2025, doi: 10.3389/frai.2025.1543603.

[29] J. Li, G. Li, Y. Li, and Z. Jin, "Structured Chain-of-Thought Prompting for Code Generation," *ACM Transactions on Software Engineering and Methodology*, vol. 34, no. 2, pp. 1–23, Feb. 2025, doi: 10.1145/3690635.

[30] G. Hermawan and E. Rainarli, "Evaluasi Gemini Flash pada Ekstraksi Jadwal Skripsi Terstruktur dan Tidak Terstruktur," *Jurnal Informatika: Jurnal Pengembangan IT*, vol. 10, no. 4, pp. 1080–1091, Nov. 2025, doi: 10.30591/jpit.v10i4.9047.