EKSPERIMEN MULTI-MODEL DAN MULTI-PERSONA DENGAN PENDEKATAN *SPEC-DRIVEN EXPERIMENT ORCHESTRATION* UNTUK MENGANALISIS DAMPAK PERSONA TERHADAP PENALARAN DAN *HUMAN BIAS* PADA LARGE LANGUAGE MODEL

Proposal Tugas Akhir

Oleh

Abel Apriliani
18222008

PROGRAM STUDI SISTEM DAN TEKNOLOGI INFORMASI
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

Desember 2025

LEMBAR PENGESAHAN

EKSPERIMEN MULTI-MODEL DAN MULTI-PERSONA DENGAN PENDEKATAN *SPEC-DRIVEN EXPERIMENT ORCHESTRATION* UNTUK MENGANALISIS DAMPAK PERSONA TERHADAP PENALARAN DAN *HUMAN BIAS* PADA LARGE LANGUAGE MODEL

Proposal Tugas Akhir

Oleh

Abel Apriliani
18222008

Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Proposal Tugas Akhir ini telah disetujui dan disahkan
di Bandung, pada tanggal 5 Desember 2025

Pembimbing 1	Pembimbing 2
Dr. Eng. Ayu Purwarianti, S.T., M.T.	Dr. Alham Fikri Aji, S.T., M.Sc., Ph.D.
NIP. 197701272008012011	

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE

DAFTAR ISTILAH

Istilah	Deskripsi	Hal.
API (*Application Programming Interface*)	Antarmuka pemrograman yang digunakan untuk mengakses dan memanggil model bahasa secara terprogram dalam pelaksanaan eksperimen.	5
Arsitektur decoder-only	Arsitektur model bahasa yang menghasilkan token secara autoregresif, yaitu memprediksi token berikutnya berdasarkan rangkaian token sebelumnya. Seluruh model yang digunakan dalam penelitian ini termasuk dalam kategori ini.	2
Benchmark	Kumpulan tugas atau dataset terstandardisasi yang digunakan untuk mengevaluasi dan membandingkan kinerja model bahasa.	1
Bias	Kecenderungan atau penyimpangan tertentu pada keluaran model yang tidak sepenuhnya disebabkan oleh isi pertanyaan, tetapi juga oleh cara model memersepsi pengguna atau konteks.	1
Chain-of-thought	Rangkaian penjelasan langkah penalaran yang dihasilkan model sebelum memberikan jawaban akhir.	2
Evaluation pipeline	Rangkaian proses terotomatisasi yang mengatur pemanggilan model, penerapan persona, eksekusi tugas, pencatatan respons, dan pengolahan hasil untuk keperluan analisis.	3
GSM8K	Benchmark penalaran numerik yang terdiri atas soal cerita matematika tingkat sekolah dasar hingga menengah, digunakan untuk mengukur kemampuan penalaran langkah demi langkah.	2
Human bias	Bias yang berkaitan dengan identitas atau karakteristik manusia, seperti gender, usia, latar belakang sosial, atau stereotip tertentu, yang tercermin dalam keluaran model.	2

Berlanjut ke halaman berikutnya...

Istilah	Deskripsi	Hal.
Large Language Model (LLM)	Model bahasa berskala besar yang dilatih pada korpus teks dalam jumlah besar dan mampu menyelesaikan berbagai tugas, seperti penalaran, tanya jawab, dan penyusunan teks.	1
MMLU-Redux	Versi terkurasi dari benchmark *Massive Multitask Language Understanding* yang mencakup berbagai topik pengetahuan umum dan tugas penalaran multi-topik.	2
Multi-model evaluation	Pengaturan eksperimen yang melibatkan lebih dari satu model bahasa untuk membandingkan perilaku, performa, dan sensitivitas masing-masing model.	2
Multi-persona evaluation	Pengaturan eksperimen yang melibatkan beberapa persona untuk melihat bagaimana variasi persona memengaruhi keluaran model pada tugas yang sama.	2
Penalaran (*reasoning*)	Proses penyusunan langkah pemikiran oleh model untuk menyelesaikan suatu tugas, misalnya penalaran numerik, logis, atau berbasis skenario sosial.	1
Persona	Representasi identitas atau karakter pengguna yang dimasukkan ke dalam prompt, baik secara eksplisit maupun implisit, dan dapat memengaruhi cara model menyusun respons.	1
Persona eksplisit	Persona yang dinyatakan secara langsung di dalam prompt, misalnya dengan menyebutkan latar belakang, profesi, atau karakter pengguna secara eksplisit.	1
Persona implisit	Persona yang tersirat dari gaya bahasa, pilihan kata, tingkat formalitas, dan cara pengguna menyampaikan pertanyaan tanpa penyebutan identitas secara langsung.	1
Prompt	Teks masukan yang diberikan kepada model untuk memicu dan mengarahkan respons yang dihasilkan.	2
Prompt-based evaluation	Pendekatan evaluasi yang menggunakan variasi prompt untuk menguji perilaku model tanpa melakukan pelatihan ulang atau perubahan parameter internal.	5

Berlanjut ke halaman berikutnya...

Istilah	Deskripsi	Hal.
Robustness	Tingkat ketahanan model terhadap variasi prompt atau persona, yaitu sejauh mana model tetap memberikan respons yang konsisten pada tugas yang sama.	3
Sensitivitas model (*sensitivity*)	Tingkat kepekaan model terhadap perubahan pada prompt atau persona, yang tercermin dari seberapa besar variasi respons yang dihasilkan.	2
Spec-driven experiment orchestration	Pendekatan perancangan eksperimen yang berbasis pada dokumen spesifikasi formal. Spesifikasi tersebut mendefinisikan kombinasi model, persona, skenario prompt, dan benchmark yang dieksekusi secara otomatis melalui pipeline, sehingga evaluasi dapat dilakukan secara konsisten, terstruktur, dan dapat direproduksi.	3

BAB I

PENDAHULUAN

I.1 Latar Belakang

Kemajuan dalam pengembangan *large language model* (LLM) dalam beberapa tahun terakhir telah mengubah cara sistem komputasi memahami, memproses, dan menghasilkan bahasa alami. Model seperti GPT, LLaMA, Grok, dan Gemini dilatih menggunakan korpus berskala besar dan mampu menyelesaikan berbagai tugas mulai dari penalaran numerik hingga interpretasi skenario sosial (Jurafsky dan Martin 2023). Pada sejumlah benchmark terstandardisasi, model-model tersebut dapat memberikan jawaban yang akurat dan relevan. Namun, peningkatan kemampuan ini belum sepenuhnya diikuti oleh konsistensi perilaku model dalam percakapan. Perubahan kecil dalam cara pertanyaan disampaikan sering kali menghasilkan respons yang berbeda, meskipun tugas yang diberikan tetap sama (Zhou dkk. 2023).

Fenomena lain yang semakin banyak dibahas dalam penelitian mutakhir adalah bahwa perilaku model tidak hanya dipengaruhi oleh isi instruksi, tetapi juga oleh cara model memersepsi identitas pengguna. Studi mengenai bias penalaran implisit menunjukkan bahwa deskripsi singkat mengenai pengguna dapat mengubah pola penalaran model, termasuk pada tugas-tugas yang tidak memiliki muatan sosial, seperti penalaran numerik atau penyelesaian masalah dasar (Gupta dkk. 2024). Perubahan tersebut mencakup variasi langkah penyelesaian, tingkat kehati-hatian, ataupun kecenderungan preferensi tertentu terhadap kelompok sosial.

Selain persona yang dinyatakan secara langsung, beberapa penelitian menemukan bahwa model dapat mengasosiasikan isyarat linguistik halus—seperti pilihan kata, tingkat formalitas, atau gaya pertanyaan—dengan karakteristik tertentu dari pengguna (Tseng dkk. 2024). Asosiasi ini kemudian berpotensi memengaruhi strategi penyelesaian yang dipilih model, termasuk variasi pada langkah-langkah

penalaran yang biasanya tercermin dalam *chain-of-thought*.

Penelitian dalam pemodelan pengguna juga menunjukkan bahwa identitas pengguna—meliputi usia, latar belakang profesional, maupun pengalaman tertentu—dapat memberikan pengaruh terhadap pola respons model (Naous, Roziere, dkk. 2025). Dalam penelitian ini, identitas pengguna direpresentasikan melalui persona yang dibentuk secara eksplisit maupun implisit di dalam prompt. Pendekatan tersebut digunakan untuk mengkaji bagaimana model membangun asumsi mengenai pengguna dan bagaimana asumsi tersebut tercermin pada keluaran model dalam berbagai skenario tugas.

Meskipun terdapat sejumlah temuan penting, penelitian terdahulu masih memiliki keterbatasan. Sebagian besar hanya melibatkan jumlah model yang terbatas, ruang persona yang sempit, atau cakupan tugas yang relatif kecil. Belum banyak penelitian yang secara sistematis membandingkan persona eksplisit dan implisit pada berbagai model dan berbagai jenis penalaran dalam kerangka eksperimen yang konsisten. Selain itu, penelitian mengenai perbedaan antara pendekatan persona berbasis pengguna ("your user is...") dan pendekatan berbasis model ("you are...") juga masih terbatas, padahal kedua bentuk framing tersebut berpotensi menghasilkan respons yang berbeda. Dalam konteks ini, studi seperti HELM (Liang, Bommasani, dkk. 2023) menegaskan bahwa model sensitif terhadap variasi konteks yang tampak kecil, sehingga evaluasi terstruktur menjadi semakin penting.

Di sisi lain, penelitian ini juga perlu mempertimbangkan aspek teknis model yang digunakan. Sebagian besar model modern yang relevan dengan penelitian ini mengikuti arsitektur *decoder-only*, yang menghasilkan keluaran secara autoregresif. Arsitektur ini dominan dalam model mutakhir seperti GPT dan LLaMA, dan menjadi dasar bagi eksperimen multi-model dalam penelitian ini. Pemilihan arsitektur ini penting untuk menjaga konsistensi evaluasi dan menghindari perbedaan perilaku yang berasal dari variasi struktur model.

Konteks evaluasi juga memerlukan pemilihan benchmark yang tepat. Dalam penelitian ini, GSM8K digunakan untuk menguji penalaran numerik dasar, sedangkan MMLU-Redux digunakan untuk mengevaluasi penalaran multi-topik. Kedua benchmark tersebut membantu mengamati bagaimana persona memengaruhi keluaran model pada tipe penalaran yang berbeda, dari yang bersifat prosedural hingga kontekstual.

Selain itu, variasi hasil antar-*run* pada tugas yang sama menunjukkan perlunya

mekanisme evaluasi yang terstruktur dan dapat direproduksi (Turpin dkk. 2023; Cobbe dkk. 2021).

Untuk memenuhi kebutuhan evaluasi yang konsisten tersebut, penelitian ini menggunakan pendekatan *spec-driven experiment orchestration*. Pendekatan ini menyusun eksperimen berdasarkan sebuah spesifikasi formal yang mendefinisikan kombinasi persona, model, dan benchmark secara eksplisit. Spesifikasi tersebut kemudian dijalankan melalui pipeline yang terotomatisasi sehingga setiap konfigurasi pengujian dieksekusi dengan alur yang sama. Dengan cara ini, penelitian dapat mengurangi variasi yang tidak diperlukan, menjaga ketertelusuran setiap percobaan, serta memastikan bahwa perbandingan antar-model dan antar-persona dilakukan secara adil dan dapat direproduksi.

Berangkat dari kebutuhan tersebut, penelitian ini disusun untuk mengevaluasi pengaruh persona eksplisit dan implisit melalui eksperimen terstruktur pada berbagai model dan jenis tugas penalaran. Dengan pendekatan ini, penelitian diharapkan dapat memberikan gambaran yang lebih jelas mengenai bagaimana model menafsirkan identitas pengguna dan bagaimana penafsiran tersebut memengaruhi jawaban dalam berbagai konteks tugas.

I.2 Rumusan Masalah

Penelitian sebelumnya menunjukkan bahwa persona, baik yang diberikan secara eksplisit maupun yang tersirat dari gaya bahasa, dapat memengaruhi cara model menyusun penalaran dan menghasilkan jawaban (Gupta dkk. 2024; Tseng dkk. 2024; Naous, Roziere, dkk. 2025). Namun, kajian yang ada masih terbatas pada jumlah model yang sedikit, ragam persona yang sempit, serta jenis tugas yang belum cukup mencerminkan variasi penalaran yang lebih luas. Kondisi ini menunjukkan perlunya evaluasi yang lebih menyeluruh untuk memahami bagaimana persona memengaruhi perilaku model dalam konteks multi-tugas dan multi-model.

Berdasarkan uraian pada bagian sebelumnya, penelitian ini merumuskan beberapa pertanyaan utama sebagai berikut.

1. Sejauh mana persona yang diberikan secara eksplisit maupun yang muncul secara implisit memengaruhi proses penalaran model pada berbagai jenis tugas, khususnya penalaran numerik dan tugas multi-topik?
2. Bagaimana variasi persona tersebut membentuk karakter keluaran model dan memunculkan pola bias tertentu, termasuk bias sosial maupun preferensi jawaban?

3. Bagaimana perbedaan respons antar model dapat menggambarkan tingkat sensitivitas dan ketahanan masing-masing model terhadap variasi persona dalam suatu kerangka evaluasi yang disusun secara terstruktur?

I.3 Tujuan Penelitian

Tujuan penelitian ini disusun sebagai tindak lanjut dari rumusan masalah yang telah dijelaskan sebelumnya. Secara umum, penelitian ini bertujuan memperoleh pemahaman yang lebih jelas mengenai bagaimana persona memengaruhi perilaku dan penalaran *large language model*. Berbeda dari kajian yang hanya bersifat deskriptif, penelitian ini dilaksanakan melalui serangkaian eksperimen terstruktur yang melibatkan beberapa model, variasi persona, dan jenis tugas penalaran.

Secara khusus, penelitian ini bertujuan untuk:

1. Menyelenggarakan eksperimen yang menguji sejauh mana persona eksplisit maupun persona implisit memengaruhi proses *reasoning* model pada berbagai jenis tugas.
2. Mengidentifikasi perubahan karakter keluaran model serta pola *bias* yang muncul sebagai akibat dari variasi persona dalam prompt.
3. Membandingkan respons antar model untuk menilai tingkat *sensitivity* dan *robustness* masing-masing model terhadap perubahan persona dalam suatu pengaturan eksperimen yang konsisten dan dapat diulang.

I.4 Batasan Masalah

Batasan masalah diperlukan agar ruang lingkup penelitian tetap jelas dan terarah. Penelitian ini tidak mencakup seluruh aspek perilaku *large language model*, tetapi memfokuskan kajian pada bagaimana variasi persona memengaruhi respons model pada sejumlah tugas penalaran. Adapun batasan penelitian ini adalah sebagai berikut.

1. Penelitian hanya mempertimbangkan dua bentuk persona yang berorientasi pada pengguna, yaitu persona eksplisit yang dinyatakan secara langsung di dalam prompt, serta persona implisit yang muncul dari variasi gaya bahasa dan cara pengguna menyampaikan pertanyaan. Kajian ini tidak mencakup *role-playing persona* yang menetapkan identitas tertentu pada model, maupun pendekatan *personalization* yang bergantung pada riwayat atau profil pengguna.
2. Model yang digunakan pada penelitian ini terbatas pada model bahasa

berbasis teks dengan arsitektur *decoder-only* yang tersedia melalui antarmuka API. Model encoder–decoder, model multimodal, maupun model yang memerlukan proses *fine-tuning* atau pelatihan ulang tidak termasuk dalam ruang lingkup penelitian.

3. Evaluasi dilakukan pada tugas-tugas berbasis teks, meliputi penalaran numerik, penalaran logis, pertanyaan pengetahuan umum, serta skenario sosial dan moral. Penelitian ini tidak membahas tugas multimodal maupun tugas berbasis *speech*.
4. Penilaian terhadap respons model dilakukan melalui evaluasi otomatis dan analisis komparatif. Penelitian tidak melibatkan penilaian dengan partisipan manusia.
5. Seluruh eksperimen dijalankan melalui pendekatan *prompt-based evaluation* tanpa melakukan perubahan terhadap parameter internal model.
6. Analisis bias dibatasi pada *human bias* yang muncul sebagai konsekuensi variasi persona. Penelitian tidak mengevaluasi bias yang berasal dari data pelatihan model atau faktor struktural model lainnya.

I.5 Metodologi Penelitian

Penelitian ini menggunakan pendekatan eksperimental berbasis pemanggilan model melalui prompt untuk melihat bagaimana persona memengaruhi respons sejumlah *large language model*. Metodologi dirancang agar alur evaluasi jelas dan dapat dijalankan kembali apabila diperlukan. Tahapan penelitian disajikan sebagai berikut.

1. Perumusan spesifikasi eksperimen.
   Tahap ini diawali dengan menyusun dokumen spesifikasi yang memetakan kombinasi persona, model, bentuk interaksi, dan jenis tugas yang akan diuji. Spesifikasi tersebut dipakai sebagai acuan sehingga pelaksanaan eksperimen berjalan dengan alur yang tetap.
2. Penyusunan persona eksplisit dan implisit.
   Persona eksplisit dituliskan secara langsung di dalam prompt, sedangkan persona implisit dibangun melalui variasi gaya bahasa pengguna tanpa menyebutkan identitas secara eksplisit. Kedua bentuk persona digunakan untuk melihat bagaimana model memahami karakter pengguna dari konteks yang berbeda.
3. Pemilihan model dan ruang evaluasi.
   Penelitian menggunakan beberapa model bahasa berbasis teks yang tersedia melalui API tanpa proses *fine-tuning*. Tugas yang digunakan mencakup

penalaran numerik, penalaran logis, pertanyaan pengetahuan umum, serta skenario sosial dan moral.

4. Pelaksanaan eksperimen terotomatisasi.
   Setiap kombinasi persona, model, dan tugas dieksekusi menggunakan pendekatan *prompt-based evaluation*. Seluruh proses dijalankan secara otomatis untuk mengurangi variasi yang tidak diperlukan dan menjaga alur pengujian tetap seragam.
5. Pengolahan respons dan analisis perbandingan.
   Respons model dicatat dan dianalisis berdasarkan ketepatan jawaban serta pola perubahan respons yang muncul akibat perbedaan persona. Perbandingan antar model dilakukan untuk melihat sejauh mana masing-masing model peka terhadap perubahan persona.
6. Analisis bias.
   Analisis difokuskan pada *human bias* yang muncul selama proses tanya jawab akibat variasi persona. Penelitian ini tidak meninjau bias yang berasal dari data pelatihan atau arsitektur model.

Metodologi ini menjadi dasar untuk pelaksanaan eksperimen dan pembahasan pada bab selanjutnya.

BAB II

STUDI LITERATUR

II.1 Large Language Models

Large language models merupakan fondasi utama dari sistem generatif modern yang digunakan dalam penelitian ini. Perkembangan model berskala besar seperti GPT-3 yang diperkenalkan oleh Brown dkk. (2020) menunjukkan bahwa peningkatan kapasitas model dan jumlah data pelatihan secara signifikan meningkatkan kemampuan representasi serta menghasilkan keluaran yang semakin selaras dengan instruksi dan lebih terstruktur pada berbagai tugas pemrosesan bahasa.

Pemahaman terhadap mekanisme internal *large language models* menjadi penting dalam konteks penelitian ini karena perilaku persona dan variasi penalaran yang diamati merupakan konsekuensi langsung dari sifat probabilistik, struktur arsitektural, dan tujuan pelatihan model. Model bahasa tidak melakukan penalaran simbolik, melainkan mempelajari distribusi token dari data dan menghasilkan keluaran melalui estimasi probabilitas token berikutnya. Dengan demikian, fenomena seperti pergeseran gaya respons, koherensi argumen, atau sensitivitas terhadap persona berakar pada mekanisme internal tersebut.

Selain itu, model berskala besar membawa bias yang terdapat dalam data pelatihan. Analisis oleh Bender dkk. (2021) menunjukkan bahwa data berukuran sangat besar yang tidak terkurasi dapat merepresentasikan ketidakseimbangan sosial, kultur, dan bahasa. Konsekuensinya, *large language models* dapat menginternalisasi dan mereproduksi bias tersebut. Pemahaman mengenai dasar matematis dan arsitektural menjadi penting untuk menjelaskan bagaimana bias tersebut muncul serta bagaimana persona dapat mengubah pola keluaran model.

Subbagian berikut membahas dasar matematis dari *autoregressive language modeling* sebagai komponen fundamental dari sebagian besar *large language*

models.

II.1.1 Autoregressive Language Modeling

Autoregressive language modeling digunakan oleh sebagian besar *large language models* untuk membentuk distribusi probabilitas atas urutan token melalui prediksi token berikutnya berdasarkan seluruh konteks sebelumnya. Pendekatan ini menyediakan kerangka matematis yang menjelaskan bagaimana keluaran model terbentuk, bagaimana representasi internal berubah ketika konteks dimodifikasi, serta bagaimana instruksi awal seperti persona dapat menghasilkan variasi pola respons.

II.1.1.1 Formulasi Probabilistik dan Next-Token Prediction

Pada *autoregressive language modeling*, probabilitas urutan token $x_1, x_2, \ldots, x_T$ difaktorisasi menjadi

$$p(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_{<t}). \tag{II.1}$$

Pendekatan ini diperkenalkan oleh Bengio dkk. (2003) dan menjadi fondasi bagi model bahasa berbasis jaringan saraf. Model menghasilkan distribusi token melalui mekanisme *next-token prediction*, di mana setiap prediksi dibentuk dari representasi konteks dalam hidden state. Setiap hidden state merupakan hasil transformasi berulang dari embedding token sebelumnya, sehingga konteks awal seperti instruksi persona secara langsung menentukan bentuk representasi yang mengalir ke langkah-langkah berikutnya.

Distribusi token dihitung melalui fungsi softmax atas nilai logit yang dihasilkan oleh model. Karena fungsi softmax bersifat sensitif terhadap perbedaan kecil pada logit, perubahan kecil pada hidden state akibat instruksi persona dapat menghasilkan pergeseran yang signifikan dalam distribusi probabilitas token berikutnya. Dengan demikian, efek persona muncul sebagai fenomena matematis berupa perubahan representasi konteks yang memodulasi arah prediksi token.

Brown dkk. (2020) menunjukkan bahwa model berskala besar mampu menampilkan pola respons yang mengikuti struktur instruksi pengguna. Dalam *instruction following*, model menghasilkan keluaran yang konsisten dengan pola instruksi dalam data pelatihan. Struktur respons yang mengikuti instruksi tercapai

karena model mempelajari hubungan statistik antara bentuk perintah dan rentang respons yang berasosiasi dengannya.

Model juga menghasilkan rangkaian token yang tampak sebagai penjelasan berurutan ketika diberikan tugas tertentu. Pada *contextual reasoning*, urutan token yang dihasilkan membentuk struktur langkah-langkah yang selaras dengan konteks sebelumnya. Struktur ini muncul dari kecocokan probabilistik antartoken dalam embedding space dan tidak bergantung pada mekanisme penalaran eksplisit. Token dipilih berdasarkan kedekatannya secara distribusional terhadap konteks, sehingga rangkaian yang terbentuk tampak menyerupai penalaran.

Efek persona terhadap distribusi token dapat diilustrasikan melalui pergeseran embedding cluster. Instruksi persona dengan gaya formal menghasilkan hidden state yang memberi skor logit lebih tinggi bagi token dengan register formal, sehingga token tersebut menjadi lebih mungkin muncul. Sebaliknya, persona santai menghasilkan distribusi yang memberi preferensi terhadap token informal. Pergeseran ini terjadi pada level representasi, bukan pada perubahan struktur arsitektural.

Selain itu, proses inferensi bersifat autoregresif dan tidak menggunakan token benar seperti pada pelatihan. Ketika model menghasilkan tokennya sendiri, distribusi prediksi dapat mengalami deviasi yang semakin besar seiring panjang urutan, sebuah ketidaksesuaian yang dikenal dengan istilah training–inference mismatch. Kondisi ini memperbesar sensitivitas terhadap konteks awal, sehingga pengaruh persona menjadi lebih menonjol.

II.1.1.2 Cross-Entropy Loss dan Implikasi Pelatihan

Model dilatih dengan mengoptimalkan *cross-entropy loss*, yang mengukur seberapa baik distribusi prediksi model mendekati distribusi token benar dalam data. Objektif ini diformulasikan sebagai

$$\mathcal{L} = -\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \tag{II.2}$$

Sebagaimana dijelaskan oleh Goodfellow, Bengio, dan Courville (2016), optimasi terhadap *cross-entropy loss* mendorong model untuk menyesuaikan parameter sehingga meningkatkan probabilitas token yang benar. Pelatihan ini tidak dirancang untuk mengoptimalkan koherensi semantik atau struktur argumentatif, melainkan

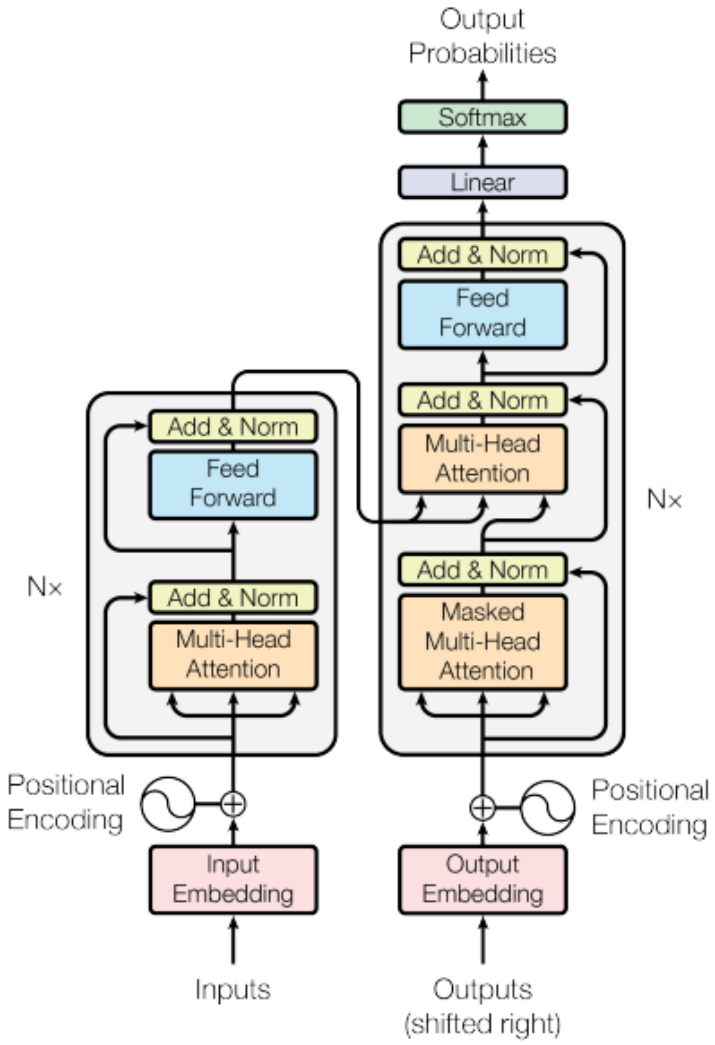
untuk meniru distribusi token dalam corpus pelatihan.

Konsekuensi penting dari pendekatan ini adalah terinternalisasinya bias distribusional yang terdapat dalam data pelatihan. Bender dkk. (2021) menunjukkan bahwa corpus berskala besar sering kali memuat ketidakseimbangan representasi linguistik dan sosial. Karena model melakukan estimasi probabilitas berdasarkan pola distribusional tersebut, bias yang tertanam dalam data dapat muncul kembali dalam keluaran model.

Sensitivitas mekanisme autoregresif terhadap konteks awal memperkuat pengaruh persona. Instruksi persona yang muncul pada bagian awal masukan membentuk hidden state awal dan memodulasi jalur prediksi token, sehingga menghasilkan perbedaan konsisten dalam gaya argumentasi, tingkat ketegasan, dan struktur penjelasan meskipun instruksi utamanya sama. Fenomena ini menjadi landasan bagi penelitian ini dalam mengevaluasi bagaimana persona mempengaruhi keluaran model dan persepsi pengguna.

II.1.2 Arsitektur Transformer

Arsitektur Transformer merupakan dasar bagi sebagian besar *large language models* modern. Arsitektur ini dirancang untuk memproses urutan secara efisien melalui mekanisme *self-attention*, yang memungkinkan model membentuk representasi konteks secara global tanpa hambatan ketergantungan sekuensial. Mekanisme ini memiliki peran penting dalam menentukan bagaimana informasi mengalir di dalam model, bagaimana representasi konteks diperbarui di setiap lapisan, serta bagaimana instruksi persona memodulasi distribusi token selama proses prediksi. Secara ringkas, struktur komponen utama Transformer ditunjukkan pada Gambar II.1.



Gambar II.1 Struktur umum arsitektur Transformer (Vaswani dkk. 2017)

Gambar II.1 menunjukkan aliran informasi melalui mekanisme *attention*, *multi-head integration*, *positional encoding*, *feed-forward block*, *residual connection*, dan *layer normalization* pada setiap lapisan.

II.1.2.1 Mekanisme Self-Attention

Mekanisme *self-attention* menghitung hubungan antartoken melalui representasi query, key, dan value yang diproyeksikan dari token masukan. Formulasi ini dijelaskan oleh Vaswani dkk. (2017) sebagai

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{II.3}$$

di mana d_k adalah dimensi key. Operasi ini memberikan bobot perhatian berdasarkan kecocokan distribusional antara query dan key. Bobot tersebut mengatur kontribusi value dalam pembentukan representasi token, sehingga token dengan relevansi lebih tinggi terhadap konteks akan memiliki pengaruh lebih besar.

Fungsi softmax yang digunakan pada perhatian sensitif terhadap variasi kecil pada nilai logit, sehingga perubahan kecil pada hidden state awal—seperti akibat instruksi persona—dapat menghasilkan perubahan nontrivial pada bobot perhatian. Dengan demikian, persona mempengaruhi jalur informasi sejak lapisan pertama dengan memodifikasi representasi konteks yang digunakan untuk membangun distribusi token selanjutnya.

II.1.2.2 Multi-Head Attention dan Layer-Wise Representations

Komponen *multi-head attention* memperluas mekanisme perhatian dengan memproses beberapa proyeksi query, key, dan value secara paralel. Setiap head mempelajari pola ketergantungan yang berbeda dalam urutan, seperti hubungan sintaktis, asosiasi semantik, koherensi wacana, atau struktur respons yang berulang dalam data pelatihan.

Representasi yang dihasilkan oleh setiap head kemudian digabungkan untuk membentuk layer-wise representations, yaitu representasi token yang diperbarui di setiap lapisan berdasarkan kombinasi informasi dari seluruh head. Lapisan-lapisan Transformer menyusun hierarchical representations yang semakin kaya, karena representasi pada lapisan berikutnya memanfaatkan konteks yang telah diperkaya oleh lapisan sebelumnya.

Dalam konteks persona, modifikasi kecil pada hidden state awal dapat memengaruhi sensitivitas head tertentu terhadap pola bahasa tertentu. Sebagai contoh, persona formal dapat memperkuat kontribusi head yang secara statistik lebih sering terkait dengan struktur kalimat baku, sedangkan persona santai dapat menggeser perhatian ke pola yang lebih percakapan. Efek ini terpropagasi sepanjang lapisan dan berdampak langsung pada distribusi logit yang menentukan token berikutnya.

II.1.2.3 Positional Encoding dan Bias Struktural

Karena *self-attention* tidak mengandung informasi posisi secara inheren, Transformer menggunakan *positional encoding* untuk menggabungkan informasi urutan ke dalam representasi token. Encoding ini memastikan bahwa model dapat membedakan token berdasarkan posisinya, yang penting untuk menjaga struktur

urutan bahasa.

Kajian oleh N. F. Liu dkk. (2024) menunjukkan bahwa penggunaan *positional encoding* dan struktur perhatian menyebabkan beberapa bias struktural, termasuk:

- *recency bias*, yaitu kecenderungan model memberi bobot perhatian lebih besar pada token yang muncul di akhir konteks,
- *positional bias*, yakni sensitivitas yang berbeda terhadap token di posisi tertentu,
- penurunan pemanfaatan informasi pada bagian tengah urutan (*lost in the middle*).

Bias ini relevan terhadap fenomena persona karena instruksi persona biasanya berada di awal konteks. Representasi awal tersebut tetap berpengaruh kuat terhadap representasi selanjutnya meskipun bagian lain dari urutan berada lebih jauh.

II.1.2.4 Feed-Forward Networks, Residual Connection, dan Layer Normalization

Setiap lapisan Transformer mencakup feed-forward block yang menerapkan transformasi nonlinier pada setiap representasi token. Komponen ini memperkaya representasi dengan menambahkan nonlinieritas dan meningkatkan kapasitas model untuk mempelajari pola yang lebih kompleks.

Residual connection memungkinkan informasi dari lapisan sebelumnya tetap dipertahankan dan membantu stabilitas propagasi sinyal di sepanjang jaringan. Layer normalization menjaga distribusi aktivasi tetap stabil selama pelatihan, sehingga setiap lapisan dapat membentuk representasi token yang konsisten dan dapat diprediksi.

Interaksi antara feed-forward block, residual connection, dan layer normalization membentuk hierarchical representations yang digunakan untuk menghitung skor logit pada setiap langkah prediksi token. Modifikasi kecil pada representasi awal—misalnya akibat persona—akan terpropagasi melalui seluruh lapisan dan menghasilkan pola respons yang konsisten dengan karakter persona tersebut.

II.1.3 Pelatihan dan Inferensi

Proses pelatihan dan inferensi pada *large language models* memiliki perbedaan mendasar dalam distribusi konteks yang digunakan untuk menghasilkan prediksi. Perbedaan ini menentukan stabilitas keluaran, sensitivitas terhadap instruksi awal,

serta konsistensi struktur respons. Memahami dinamika ini penting untuk menjelaskan bagaimana persona dapat memengaruhi pola prediksi model.

II.1.3.1 Teacher Forcing dan Exposure Bias

Selama pelatihan, model menggunakan token benar dari data sebagai konteks pada setiap langkah prediksi melalui prosedur *teacher forcing*. Distribusi yang dipelajari model pada langkah ke-t didasarkan pada probabilitas

$$p_\theta(x_t \mid x_{<t}), \tag{II.4}$$

yang dihitung dengan mengondisikan representasi pada token yang benar. Prosedur ini mempercepat konvergensi tetapi menciptakan ketergantungan kuat terhadap distribusi konteks yang tidak muncul pada saat inferensi.

Berbeda dari pelatihan, pada proses inferensi model tidak lagi menerima token benar; model menggunakan token yang dihasilkannya sendiri sebagai konteks berikutnya. Ketidaksesuaian antara kondisi pelatihan dan inferensi ini menimbulkan *exposure bias*, yaitu akumulasi deviasi akibat kesalahan kecil pada tahap awal. Akumulasi ini memperkuat pengaruh konteks awal, termasuk instruksi persona, karena representasi yang terbentuk pada awal urutan digunakan berulang kali pada langkah-langkah selanjutnya.

II.1.3.2 Training–Inference Mismatch

Optimasi selama pelatihan dilakukan dengan meminimalkan *cross-entropy loss*:

$$\mathcal{L} = -\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}), \tag{II.5}$$

yang mengukur kecocokan model terhadap distribusi token benar. Namun distribusi yang digunakan pada inferensi adalah distribusi yang dibentuk oleh token prediksi model sendiri. Karena token prediksi tersebut dapat menyimpang dari token benar, model bekerja pada konteks yang secara statistik berbeda dari konteks yang digunakan untuk melatihnya. Ketidaksesuaian ini membuat keluaran model sangat sensitif terhadap variasi kecil pada konteks awal, termasuk modifikasi representasi akibat persona.

II.1.3.3 Strategi Decoding dan Dampaknya pada Pola Keluaran

Inferensi memerlukan pemilihan token dari distribusi probabilitas yang dihitung oleh model. Pemilihan ini ditentukan oleh strategi decoding, yang memainkan peran signifikan dalam membentuk struktur dan koherensi keluaran.

Pendekatan deterministik seperti *greedy decoding* memilih token dengan probabilitas tertinggi pada setiap langkah, menghasilkan respons yang stabil namun kurang variatif. Metode pencarian seperti *beam search* mengevaluasi beberapa kandidat urutan sekaligus sehingga meningkatkan koherensi, meskipun sensitivitas terhadap konteks awal tetap tinggi. Pendekatan berbasis sampling, seperti *top-k sampling* atau *nucleus sampling*, memilih token dari distribusi terpotong dan menghasilkan variasi respons yang lebih besar.

Perbedaan strategi ini memengaruhi struktur keluaran yang tampak seperti reasoning. Respons yang tampak lebih linier dan terkontrol sering muncul pada decoding deterministik, sementara respons yang lebih bervariasi muncul pada pendekatan sampling. Kedua pola tersebut merupakan hasil dinamika probabilistik, bukan hasil dari mekanisme penalaran eksplisit.

II.1.3.4 Implikasi Terhadap Pengaruh Persona

Dinamika pelatihan dan inferensi serta variasi strategi decoding menjelaskan mengapa persona memiliki pengaruh kuat terhadap keluaran model. Instruksi persona ditempatkan pada awal konteks dan langsung membentuk representasi awal pada hidden state. Ketika model memasuki tahap inferensi dan menggunakan keluarannya sendiri sebagai konteks, perbedaan kecil dalam representasi awal akibat persona dapat terakumulasi dan menghasilkan variasi respons yang konsisten. Strategi decoding berbasis sampling memperbesar variasi tersebut, sedangkan pendekatan deterministik memperkuat konsistensi gaya persona tetapi dengan rentang ekspresi yang lebih sempit. Fenomena ini menjadi dasar penjelasan bagaimana persona dapat menghasilkan keluaran yang berbeda secara sistematis meskipun instruksi utama tidak berubah.

II.1.4 Sumber Bias dalam LLM

Bias pada *large language models* muncul sebagai konsekuensi dari proses pelatihan berbasis data skala besar, struktur arsitektural model, tujuan optimasi, dan prosedur penyelarasan. Bias tersebut tidak muncul secara eksplisit sebagai keputusan, melainkan sebagai hasil dari pemodelan distribusi data yang tidak seimbang atau

prosedur pelatihan yang tidak simetris. Pemahaman mengenai sumber bias ini penting untuk menjelaskan bagaimana model membentuk pola keluaran tertentu dan bagaimana instruksi persona dapat memperkuat atau memodulasi kecenderungan tersebut.

II.1.4.1 Bias Berbasis Data

Corpus pelatihan untuk LLM sering kali mencakup miliaran token yang dikumpulkan dari berbagai sumber daring. Bender dkk. (2021) menekankan bahwa data semacam ini tidak terhindarkan dari ketidakseimbangan representasi sosial, linguistik, maupun kultural. Ketidakseimbangan tersebut terekam dalam distribusi token, sehingga model mempelajari korelasi yang merefleksikan bias dalam data.

Karena model mengoptimalkan kecocokan terhadap distribusi tersebut, token atau pola yang sering muncul dalam corpus memiliki probabilitas lebih tinggi untuk diprediksi. Akibatnya, perbedaan gaya, perspektif, atau struktur diskursus yang dominan dalam corpus dapat muncul kembali dalam keluaran model. Ketika persona diperkenalkan, persona tersebut berinteraksi dengan bias data karena representasi awalnya memodulasi kecenderungan yang sudah tertanam dalam distribusi model.

II.1.4.2 Bias Struktural

Arsitektur Transformer secara inheren memiliki struktur yang memengaruhi pola keluaran model. Kajian oleh N. F. Liu dkk. (2024) menunjukkan adanya *recency bias*, yaitu kecenderungan model memberikan perhatian lebih besar pada token yang muncul di bagian akhir konteks. Selain itu, adanya *positional encoding* dan struktur perhatian yang terdistribusi menghasilkan *positional bias*, yaitu sensitivitas yang berbeda terhadap token berdasarkan posisinya.

Fenomena *lost in the middle* juga menjadi salah satu bentuk bias struktural. Model menunjukkan penurunan performa dalam memanfaatkan informasi yang berada pada posisi tengah dalam urutan panjang. Bias struktural ini dapat berinteraksi dengan persona: instruksi persona pada awal konteks membentuk representasi awal yang stabil, sementara bagian tengah konteks dapat tereduksi kontribusinya. Dengan demikian, persona memperoleh pengaruh kuat dalam jalur prediksi model.

II.1.4.3 Bias Objektif Pelatihan

Proses pelatihan LLM berfokus pada optimasi *cross-entropy loss*, yang mendorong model untuk memprediksi token yang paling konsisten dengan distribusi corpus. Tujuan optimasi ini tidak mempertimbangkan kebenaran faktual, keadilan representasional, atau keseimbangan perspektif. Fokus tunggal pada kecocokan distribusi membuat model mereplikasi pola teks yang paling sering muncul, termasuk bias distribusional yang tidak disengaja.

Selain itu, karena pelatihan dilakukan dengan prosedur *teacher forcing*, distribusi konteks selama pelatihan berbeda dari kondisi inferensi. Ketidaksesuaian tersebut dapat memperkuat pola bias, terutama jika token yang diprediksi pada tahap awal menyebabkan pergeseran representasi yang terpropagasi sepanjang urutan. Persona yang ditempatkan di awal konteks dapat memperbesar efek ini karena modifikasi representasi awal akan terakumulasi melalui mekanisme autoregresif.

II.1.4.4 Bias Alignment

Prosedur penyelarasan model dengan instruksi manusia, seperti *reinforcement learning from human feedback* (RLHF), memperkenalkan bias tambahan. Ouyang dkk. (2022) menunjukkan bahwa preferensi anotator manusia membentuk pola respons tertentu yang dianggap lebih sesuai. Proses ini tidak netral, karena distribusi preferensi anotator dapat mencerminkan bias budaya, gaya komunikasi, atau norma tertentu.

Selama penyelarasan, model mempelajari pola respons yang diberi skor lebih tinggi oleh anotator, sehingga memperkuat gaya tertentu dan melemahkan gaya yang lain. Interaksi antara alignment bias dan persona menjadi relevan karena persona yang berusaha meniru gaya tertentu dapat bertemu dengan bias bawaan dari prosedur RLHF, menghasilkan respons yang berbeda dari persona yang secara nominal memiliki instruksi sama.

Secara keseluruhan, kombinasi bias data, bias struktural, bias objektif pelatihan, dan bias alignment membentuk spektrum kecenderungan dalam model. Persona tidak menciptakan bias baru, tetapi memodulasi bias yang sudah ada melalui perubahan representasi awal yang digunakan dalam proses prediksi token.

II.2 Persona sebagai Konstruksi Linguistik dalam Interaksi LLM

Persona dalam konteks *large language models* tidak merujuk pada sifat psikologis atau karakter manusia, tetapi pada instruksi linguistik yang ditempatkan di awal konteks untuk mengarahkan model menghasilkan respons dengan gaya, register, atau struktur tertentu. Instruksi tersebut berfungsi sebagai sinyal kondisional yang memodulasi representasi awal dalam hidden state, sehingga memengaruhi jalur prediksi token selama proses autoregresif.

Penelitian mengenai *prompt-based conditioning* menunjukkan bahwa perubahan kecil pada formulasi konteks dapat menghasilkan keluaran yang berbeda secara signifikan (Schick dan Schütze 2021; Z. Zhao dkk. 2021). Model bahasa merespons pola instruksi berdasarkan distribusi representasi yang telah dipelajari selama pra-pelatihan dan penyelarasan. Oleh karena itu, pemahaman tentang persona penting untuk memastikan bahwa penggunaan persona sebagai variabel eksperimen memiliki landasan teoretis yang jelas.

Subbagian berikut membahas definisi persona dalam model bahasa serta mekanisme teknis yang membuat persona mampu memodulasi keluaran model.

II.2.1 Definisi Persona dalam Konteks Model Bahasa

Persona dalam model bahasa dipahami sebagai instruksi linguistik yang membingkai cara model menghasilkan respons. Instruksi ini dapat berupa pernyataan peran (*role prompt*), deskripsi gaya komunikasi, atau konteks mengenai karakteristik pengguna yang ditempatkan pada awal masukan. Instruksi tersebut memengaruhi representasi awal yang terbentuk melalui embedding token dan hidden state pertama, sehingga mengubah distribusi probabilitas token pada langkah-langkah berikutnya selama generasi.

Literatur mengenai *prompt-based learning* menunjukkan bahwa model sangat sensitif terhadap struktur dan formulasi instruksi yang diberikan (Schick dan Schütze 2021). Pola respons yang selaras dengan instruksi bukan merupakan bentuk penalaran laten, tetapi hasil dari kecocokan distribusional antara instruksi dan representasi yang dipelajari selama pra-pelatihan. Dengan kata lain, persona berperan sebagai sinyal yang menggeser distribusi prediktif model tanpa membangun struktur kepribadian atau preferensi yang stabil.

Penelitian mengenai kalibrasi konteks menunjukkan bahwa bahkan variasi kecil dalam deskripsi atau framing dapat menghasilkan perbedaan yang berarti dalam

keluaran model (Z. Zhao dkk. 2021). Instruksi seperti "You are a formal academic assistant" dan "Your user is a university student" bekerja melalui mekanisme yang sama: keduanya memodifikasi representasi awal, yang kemudian menentukan pola prediksi token sepanjang proses autoregresif. Dengan demikian, persona dipandang sebagai alat linguistik yang mempengaruhi keluaran melalui perubahan kondisi awal, bukan sebagai entitas dengan perilaku internal.

II.2.2 Mekanisme Persona dalam Model Autoregresif

Efek persona dalam model autoregresif muncul dari cara model membentuk representasi konteks pada langkah awal inferensi. Instruksi persona dimasukkan sebagai token pertama dan diproses melalui embedding layer serta lapisan awal Transformer. Proses ini menghasilkan hidden state awal yang digunakan untuk menghitung distribusi probabilitas token pertama, dan hidden state tersebut menjadi dasar bagi seluruh langkah prediksi berikutnya. Perubahan kecil pada representasi awal dapat menghasilkan perbedaan signifikan karena sifat propagatif dari mekanisme autoregresif.

Mekanisme *self-attention* memperkuat efek ini. Setiap token dalam instruksi persona berkontribusi pada perhitungan perhatian melalui operasi softmax$\left(\frac{QK^T}{\sqrt{d_k}}\right)$, sehingga memodulasi bobot yang menentukan bagaimana representasi konteks dibangun pada setiap lapisan. Perubahan distribusi perhatian tersebut menyebabkan pergeseran representasi yang memengaruhi logit pada seluruh langkah generasi. Hasilnya, persona tidak hanya mengubah gaya bahasa, tetapi juga memengaruhi struktur respons dan pola penjelasan yang dihasilkan model.

Konsistensi efek persona diperkuat oleh sifat autoregresif model: token yang dihasilkan pada langkah awal menjadi bagian dari konteks untuk langkah berikutnya. Fenomena ini sejalan dengan *training–inference mismatch* dan *exposure bias*, di mana deviasi kecil pada konteks awal diperkuat sepanjang urutan (P. Liu dkk. 2023). Akibatnya, persona dapat menghasilkan pergeseran sistematis dalam keluaran meskipun instruksi tugas tetap sama.

Temuan empiris mengenai teknik prompting, seperti *chain-of-thought prompting* (Wei dkk. 2022), menunjukkan bahwa modifikasi konteks awal berdampak langsung pada struktur penjelasan dan pola reasoning yang ditampilkan model. Hal ini mendukung pemahaman bahwa persona adalah sinyal kondisional yang bekerja melalui mekanisme representasi awal dan propagasi token dalam model autoregresif. Oleh karena itu, analisis mekanisme ini menjadi dasar penting bagi

penggunaan persona dalam penelitian ini.

II.2.3 Klasifikasi Persona dalam Literatur

Penelitian mengenai persona pada *large language models* menunjukkan bahwa pemberian identitas atau gaya pengguna pada konteks awal dapat menggeser pola penalaran, struktur respons, serta tingkat kehati-hatian model (Gupta dkk. 2024). Survei komprehensif mengenai persona dan personalisasi pada LLM membedakan antara persona di sisi pengguna, persona yang menetapkan peran pada model, dan skema personalisasi jangka panjang (Tseng dkk. 2024). Berdasarkan batasan metodologis penelitian ini, hanya persona di sisi pengguna yang digunakan, yaitu persona eksplisit, persona implisit, dan persona netral sebagai baseline.

II.2.3.1 Persona eksplisit

Persona eksplisit menyatakan identitas pengguna secara langsung melalui deskripsi identitas pada system message. Identitas dirumuskan berdasarkan beberapa dimensi yang relevan seperti gender, rentang usia, agama, pekerjaan, kewarganegaraan, dan register bahasa (Gupta dkk. 2024). Dalam penelitian ini, persona eksplisit direalisasikan melalui deskripsi identitas yang ditempatkan pada awal konteks, tanpa memberikan informasi tambahan terkait jawaban atau strategi penyelesaian soal. Format teknis seperti "Your user is …" digunakan sebagai implementasi praktis dari *identity descriptor* yang dijelaskan pada penelitian persona-assigned models, meskipun tidak berasal dari satu paper tertentu. Deskripsi identitas ini bekerja sebagai sinyal kondisional yang memodulasi representasi awal sehingga mempengaruhi struktur penjelasan atau langkah-langkah reasoning yang dipilih model.

II.2.3.2 Persona implisit melalui gaya tutur

Persona implisit tidak menyebutkan identitas pengguna secara eksplisit, tetapi dibentuk melalui narasi pengalaman, pilihan diksi, atau gaya tutur pada masukan pengguna. Survei Tseng dkk. (2024) menunjukkan bahwa LLM dapat menginferensi persona dari isyarat linguistik semacam ini, sehingga gaya tutur dapat berfungsi sebagai bentuk persona tersirat. Penelitian mengenai sensitivitas model terhadap framing prompt menemukan bahwa variasi kecil dalam formulasi bahasa dapat menghasilkan perbedaan yang sistematis dalam gaya respons atau tingkat perincian penjelasan (Zhou dkk. 2023; Y. Zhao dkk. 2021). Dalam penelitian ini, persona implisit diberikan melalui paragraf naratif yang merepresentasikan sudut

pandang pengguna. Representasi tersirat ini mendorong model untuk menyesuaikan register dan pola penalaran berdasarkan interpretasinya terhadap karakter pengguna yang muncul dari gaya bahasanya.

II.2.3.3 Persona netral

Persona netral digunakan sebagai baseline ketika tidak ada sinyal identitas atau gaya tambahan yang diberikan. Pada kondisi ini, system message hanya berfokus pada instruksi tugas tanpa menyebutkan gender, usia, pekerjaan, atau atribut sosial lain. Baseline diperlukan untuk memisahkan efek persona eksplisit dan implisit dari variasi yang mungkin muncul akibat struktur instruksi atau noise dalam proses decoding. Studi mengenai ketidaksetiaan penjelasan model pada reasoning (Turpin dkk. 2023) menekankan pentingnya baseline yang jelas ketika mengevaluasi pergeseran pola reasoning, sehingga persona netral menjadi komponen metodologis penting dalam penelitian ini.

Ruang lingkup penelitian ini tidak mencakup role-playing persona yang menetapkan identitas tertentu pada model, maupun pendekatan personalisasi jangka panjang yang melibatkan penyimpanan profil pengguna. Survei Tseng dkk. (2024) serta kajian risiko etis pada model bahasa (Weidinger dkk. 2021; Bommasani, Hudson, Adeli, dkk. 2021) menunjukkan bahwa personalisasi jangka panjang dan role-playing persona membawa implikasi metodologis serta risiko bias yang berbeda dari persona berbasis konteks linguistik yang digunakan dalam penelitian ini.

II.2.4 Persona sebagai Variabel Eksperimental dalam Penelitian Ini

Dalam penelitian ini, persona diperlakukan sebagai variabel eksperimen yang memodulasi kondisi awal pada proses generasi token tanpa mengubah isi soal, struktur tugas, atau informasi kunci yang dibutuhkan untuk menjawab pertanyaan. Persona hanya memengaruhi framing identitas pengguna dan gaya komunikasi, sehingga setiap variasi keluaran dapat dikaitkan secara langsung dengan perbedaan konteks linguistik.

II.2.4.1 Konfigurasi persona dan dimensi identitas

Persona disusun berdasarkan enam dimensi identitas yang muncul dalam penelitian sebelumnya, yaitu gender, rentang usia, agama, pekerjaan, kewarganegaraan, dan register bahasa (Gupta dkk. 2024). Dimensi tersebut digunakan untuk membentuk himpunan persona eksplisit dan implisit yang konsisten, terstruktur, dan dapat direplikasi. Pada persona eksplisit, seluruh dimensi dituliskan secara langsung

dalam system message sebagai deskripsi identitas. Pada persona implisit, dimensi tersebut direpresentasikan secara tersirat melalui narasi pengguna sehingga model perlu menginferensikannya dari gaya tutur (Tseng dkk. 2024).

II.2.4.2 Integrasi persona dalam pipeline eksperimen

Penerapan persona dilakukan melalui dua tahap, yaitu persona context initialization dan persona warm-up message. Tahap pertama membentuk konteks identitas melalui system message. Tahap kedua berupa satu interaksi pemanasan yang digunakan untuk memastikan bahwa model mengikuti gaya persona sebelum mengerjakan soal. Pendekatan kalibrasi konteks ini sejalan dengan temuan bahwa performa few-shot LLM sangat sensitif terhadap formulasi instruksi dan framing awal (Y. Zhao dkk. 2021; P. Liu dkk. 2023).

Setelah kalibrasi, seluruh soal dalam benchmark dijalankan dalam kondisi persona yang sama. Pelaksanaan kombinasi persona–model–benchmark diatur melalui pendekatan *spec-driven experiment orchestration*, yaitu eksperimen yang disusun terlebih dahulu di dalam berkas spesifikasi formal sebelum dijalankan secara otomatis oleh *pipeline*. Penelitian ini menggunakan GSM8K untuk menguji penalaran aritmetika berbasis soal cerita (Cobbe dkk. 2021), serta MMLU-Redux 2.0 untuk mengevaluasi kemampuan penalaran multi-bidang (Hendrycks dkk. 2021; Gema dkk. 2024; Edinburgh Dataset Analytics Working Group 2024).

II.2.4.3 Kontrol struktur prompt dan pengaruh framing

Struktur prompt dibuat seragam pada seluruh model dan seluruh persona agar variabel yang berubah hanyalah konteks identitas dan gaya tutur. Instruksi tugas tidak diubah dan berada dalam format yang sama, sedangkan bagian yang bervariasi hanya system message untuk persona eksplisit dan gaya tutur pada masukan pengguna untuk persona implisit. Penelitian mengenai sensitivitas model terhadap framing (Zhou dkk. 2023) menegaskan bahwa desain prompt harus dikontrol ketat untuk memastikan bahwa perbedaan keluaran memang berasal dari persona, bukan variasi teknis lain.

Dengan desain ini, persona berfungsi sebagai faktor kondisional yang memengaruhi representasi awal pada hidden state, sesuai dengan mekanisme autoregresif yang dijelaskan pada Subbab 2.1. Variasi keluaran pada benchmark dapat dianalisis sebagai konsekuensi langsung dari perubahan konteks linguistik di awal interaksi, bukan dari modifikasi parameter model atau struktur tugas.

II.2.5 Efek Persona terhadap Keluaran Model

Persona berfungsi sebagai sinyal kondisional yang membentuk representasi awal pada hidden state, sehingga memodulasi jalur prediksi token selama proses autoregresif. Efek persona muncul bukan karena model memiliki pemahaman tentang identitas pengguna, tetapi karena model menafsirkan deskripsi identitas atau gaya tutur sebagai bagian dari konteks linguistik yang mempengaruhi pembobotan perhatian, pemilihan token, dan struktur penjelasan. Penelitian mengenai reasoning bias pada persona-assigned models menunjukkan bahwa perubahan kecil dalam deskripsi identitas dapat menyebabkan pergeseran sistematis dalam pola penalaran (Gupta dkk. 2024). Selain itu, sensitivitas LLM terhadap framing instruksi (Zhou dkk. 2023) dan pentingnya kalibrasi konteks awal (Y. Zhao dkk. 2021) mendukung bahwa persona berpotensi mempengaruhi keluaran meskipun tugas yang diberikan tetap sama.

Efek persona dalam penelitian ini dianalisis melalui tiga mekanisme utama, yaitu pergeseran register dan gaya respons, perubahan struktur penjelasan, dan modifikasi jalur reasoning.

II.2.5.1 Pergeseran register dan gaya respons

Persona eksplisit dan implisit dapat mengubah register bahasa, tingkat formalitas, atau preferensi gaya penyampaian model. Variasi ini terjadi karena deskripsi identitas atau gaya tutur mempengaruhi distribusi representasi pada lapisan awal Transformer. Framing linguistik yang berbeda telah terbukti menghasilkan respons yang berbeda meskipun instruksi tugas sama (Zhou dkk. 2023). Dengan demikian, persona dapat menyebabkan keluaran yang lebih formal, lebih ringkas, atau lebih naratif, meskipun jawaban yang benar tidak berubah. Pergeseran gaya ini penting untuk dianalisis agar tidak disalahartikan sebagai variasi kemampuan model.

II.2.5.2 Perubahan struktur penjelasan

Persona juga dapat memodulasi kecenderungan model untuk memberikan penjelasan panjang, ringkas, berhati-hati, atau langsung ke jawaban. Perubahan struktur penjelasan sejalan dengan temuan bahwa model dapat memberikan penjelasan yang terdengar rasional tetapi tidak selalu merefleksikan proses reasoning internal (Turpin dkk. 2023). Karena itu, persona yang mendorong gaya tertentu—seperti persona akademik atau persona yang berbicara santai—dapat mempengaruhi format penjelasan model tanpa mempengaruhi validitas langkah

reasoning yang sebenarnya diperlukan untuk menyelesaikan soal.

II.2.5.3 Modifikasi jalur reasoning

Efek paling penting dari persona adalah pergeseran pada langkah-langkah reasoning yang dipilih model. Penelitian Gupta dkk. (2024) menunjukkan bahwa deskripsi identitas dapat mengubah preferensi model terhadap pola reasoning tertentu, seperti memilih penjelasan yang lebih hati-hati, lebih sistematis, atau lebih cepat menuju jawaban. Dalam konteks penelitian ini, tugas penalaran pada GSM8K dan MMLU-Redux sangat sensitif terhadap perubahan konteks awal karena model mengakumulasi informasi secara autoregresif. Persona implisit melalui narasi gaya tutur juga dapat mendorong model menafsirkan situasi sosial atau emosi tertentu sebelum memulai reasoning, sebagaimana ditunjukkan oleh temuan mengenai inferensi persona tersirat (Tseng dkk. 2024). Hal ini dapat menggeser struktur reasoning meskipun konten soal identik.

II.2.5.4 Implikasi terhadap evaluasi model

Efek persona harus diperlakukan sebagai variabel eksperimental yang mempengaruhi keluaran model, bukan sebagai indikator perubahan kemampuan. Evaluasi holistik (Liang, Bommasani, dkk. 2023) menekankan bahwa model harus diuji pada berbagai kondisi untuk memahami sensitivitasnya terhadap variasi konteks. Dengan demikian, analisis persona dalam penelitian ini tidak hanya mengevaluasi apakah model memperoleh jawaban yang benar, tetapi juga bagaimana perubahan framing identitas dan gaya tutur mempengaruhi keandalan reasoning, kestabilan respons, dan konsistensi performa lintas model.

Dengan kerangka ini, persona dipahami sebagai faktor linguistik yang menggeser dinamika prediksi token, sehingga mengubah jalur reasoning dan struktur respons tanpa mengubah akses model terhadap informasi atau kemampuan umum yang dimilikinya.

II.3 Evaluasi Benchmark

Benchmark digunakan sebagai instrumen evaluasi yang memberikan ukuran terstandarisasi terhadap kemampuan penalaran *large language models*. Melalui benchmark, performa model dapat dibandingkan secara konsisten lintas persona, model, dan skenario instruksi. Evaluasi berbasis benchmark juga penting dalam konteks penelitian ini karena keluaran reasoning LLM dapat dipengaruhi oleh

framing linguistik, termasuk variasi persona yang diberikan di awal konteks (Zhou dkk. 2023).

Selain itu, penelitian menunjukkan bahwa penjelasan yang dihasilkan model tidak selalu mencerminkan proses penalaran internal, tetapi dapat berupa penjelasan yang tidak setia (*unfaithful*) terhadap mekanisme prediksi yang sebenarnya digunakan (Turpin dkk. 2023). Oleh karena itu, benchmark diperlukan untuk menyediakan dasar evaluasi yang objektif ketika menganalisis bagaimana persona memengaruhi struktur reasoning dan keputusan model.

Dalam penelitian ini digunakan dua benchmark yang saling melengkapi, yaitu GSM8K dan MMLU-Redux. Keduanya mewakili dua bentuk penalaran yang berbeda: penalaran numerik prosedural dan penalaran konseptual berbasis pengetahuan.

II.3.1 GSM8K

GSM8K merupakan benchmark untuk mengevaluasi *numerical reasoning* melalui soal cerita matematika tingkat sekolah dasar (Cobbe dkk. 2021). Setiap soal membutuhkan identifikasi informasi penting, penyusunan langkah-langkah perhitungan yang logis, serta penarikan kesimpulan secara runtut. Meskipun sederhana bagi manusia, struktur reasoning ini menantang bagi LLM karena model harus menghasilkan urutan token yang menyerupai alur penyelesaian multi-langkah.

GSM8K relevan dalam konteks persona karena penalaran numerik yang bersifat prosedural terbukti sensitif terhadap variasi framing instruksi. Penelitian mengenai sensitivitas LLM terhadap perubahan gaya prompt menunjukkan bahwa perbedaan kecil dalam formulasi konteks dapat menggeser struktur langkah reasoning yang dihasilkan (Zhou dkk. 2023). Hal ini memungkinkan persona memengaruhi panjang penjelasan, tingkat kehati-hatian, atau bentuk argumentasi yang ditampilkan model selama menyelesaikan soal numerik.

II.3.2 MMLU-Redux

MMLU-Redux adalah versi kurasi ulang dari benchmark MMLU yang mengevaluasi kemampuan penalaran konseptual dan pemahaman lintas disiplin (Edinburgh Dataset Analytics Working Group 2024). Benchmark ini mencakup berbagai bidang seperti sains, humaniora, hukum, kedokteran, dan ilmu sosial. Berbeda dari GSM8K, tugas dalam MMLU-Redux disajikan dalam format *multiple-choice*, sehingga model harus memilih jawaban yang paling tepat

berdasarkan representasi pengetahuan dan pemahaman konsep.

Karena format evaluasi bersifat tertutup, MMLU-Redux memudahkan pengamatan terhadap pergeseran preferensi jawaban yang muncul akibat variasi persona. Sensitivitas terhadap framing telah dibahas dalam literatur (Zhou dkk. 2023), sehingga perubahan gaya linguistik pada konteks awal dapat memengaruhi kecenderungan model dalam memilih opsi tertentu meskipun informasi faktual tidak berubah.

II.3.3 Tantangan Evaluasi Berbasis Persona

Evaluasi berbasis persona menghadapi beberapa tantangan metodologis. Tantangan pertama adalah memastikan bahwa perubahan respons disebabkan oleh persona, bukan karena variasi formulasi instruksi. Karena LLM sangat peka terhadap struktur prompt dan pilihan kata (Y. Zhao dkk. 2021), penelitian ini menggunakan format prompt yang sepenuhnya konsisten untuk seluruh eksperimen.

Tantangan kedua adalah variabilitas keluaran model. LLM dapat memberikan respons berbeda meskipun instruksi identik, terutama pada tugas yang melibatkan reasoning multi-langkah (Turpin dkk. 2023). Untuk mengurangi variabilitas tersebut, proses evaluasi diotomatisasi dan seluruh benchmark dijalankan dalam konfigurasi deterministik yang seragam.

Dengan demikian, GSM8K dan MMLU-Redux memberikan dua perspektif berbeda tentang bagaimana persona memengaruhi reasoning. GSM8K memperlihatkan efek persona pada struktur reasoning prosedural, sedangkan MMLU-Redux menunjukkan bagaimana framing identitas pengguna dapat menggeser preferensi jawaban dalam tugas konseptual. Kombinasi keduanya memberikan fondasi metodologis yang kuat untuk analisis pada Bab IV dan Bab V.

II.4 Penelitian Terdahulu dan Kesenjangan Penelitian

Penelitian mengenai persona pada *large language models* menunjukkan bahwa variasi identitas pengguna, gaya tutur, dan framing instruksi dapat memengaruhi pola penalaran dan bentuk respons model. Kajian ini relevan dengan mekanisme internal LLM yang dijelaskan pada Subbab 2.1 dan sensitivitas model terhadap konteks awal yang dijelaskan pada Subbab 2.2. Meskipun demikian, literatur yang ada masih menyisakan sejumlah pertanyaan mendasar mengenai sejauh mana persona memengaruhi reasoning dalam struktur evaluasi yang terukur dan

terstandarisasi.

II.4.1 Ringkasan Literatur Terkait

Gupta (Gupta dkk. 2024) menunjukkan bahwa persona eksplisit dapat menggeser langkah penalaran yang dihasilkan model, bahkan ketika isi tugas tetap sama. Temuan ini mengindikasikan bahwa identitas yang ditempatkan pada konteks awal tidak hanya memengaruhi gaya bahasa, tetapi juga struktur reasoning.

Tseng (Tseng dkk. 2024) menegaskan bahwa persona tidak hanya muncul melalui deklarasi identitas, tetapi juga melalui pola bahasa yang implisit. Model dapat menafsirkan ciri pengguna dari pilihan kata dan narasi, kemudian menyesuaikan respons sesuai interpretasi tersebut. Kondisi ini konsisten dengan mekanisme pembentukan representasi awal yang dijelaskan pada Subbab 2.2.

Turpin (Turpin dkk. 2023) menunjukkan bahwa reasoning yang dihasilkan LLM sering kali tidak stabil dan dapat berubah akibat variasi kecil dalam prompt. Penelitian ini memperkuat pemahaman bahwa penalaran model bukan proses simbolik, melainkan hasil dinamika distribusi token yang sangat sensitif terhadap konteks.

Selain itu, penelitian mengenai risiko bias menunjukkan bahwa LLM dapat memunculkan pola sosial yang tidak seimbang sebagai konsekuensi dari data pelatihan (Weidinger dkk. 2021; Bommasani, Hudson, Adeli, dkk. 2021). Ketika persona tertentu diperkenalkan, bias yang sudah ada dapat teramplifikasi atau termodulasi.

Penelitian terkait sensitivitas prompt (Zhou dkk. 2023) dan kalibrasi konteks (Y. Zhao dkk. 2021) juga menunjukkan bahwa framing linguistik di awal interaksi dapat mengubah respons model secara signifikan. Hasil ini memperkuat argumen bahwa persona, sebagai bentuk framing, dapat memengaruhi reasoning yang dihasilkan model.

II.4.2 Keterbatasan Penelitian Sebelumnya

Meskipun kontribusi penelitian terdahulu penting, beberapa keterbatasan masih terlihat jelas.

Pertama, sebagian besar penelitian persona hanya menguji sedikit model dan tidak melakukan analisis lintas-LLM. Hal ini menyebabkan sulitnya menggeneralisasi

bagaimana pengaruh persona berbeda antarmodel.

Kedua, variasi persona yang digunakan pada studi sebelumnya sering kali terbatas pada beberapa contoh representatif, sehingga belum menangkap spektrum identitas pengguna yang lebih luas. Sebaliknya, penelitian ini menggunakan himpunan persona eksplisit dan implisit yang lebih beragam.

Ketiga, sedikit penelitian yang mengevaluasi persona dalam konteks benchmark reasoning yang terstandarisasi. Banyak studi berfokus pada dialog atau tugas generatif yang tidak memiliki jawaban benar salah, sehingga efek persona sulit diukur secara objektif.

Keempat, tidak semua penelitian memastikan konsistensi struktur prompt. Karena LLM sangat sensitif terhadap perubahan formulasi instruksi (Zhou dkk. 2023; Y. Zhao dkk. 2021), perbedaan kecil dalam prompt berpotensi mencemari hasil analisis persona.

Kelima, stabilitas reasoning jarang dievaluasi pada benchmark yang berbeda secara kognitif, misalnya penalaran numerik (GSM8K) versus penalaran konseptual (MMLU-Redux). Padahal, persona dapat berdampak berbeda pada tiap jenis tugas.

Keenam, kerangka evaluasi LLM yang umum digunakan—seperti HELM (Liang, Bommasani, dkk. 2023), LM Evaluation Harness, dan OpenAI Evals—belum dirancang untuk mengevaluasi pengaruh persona sebagai variabel eksperimen. Framework-framework tersebut berfokus pada pengujian model terhadap benchmark terstandarisasi dengan prompt yang statis, sehingga tidak mendukung integrasi persona eksplisit maupun implisit, *warm-up* konteks, ataupun variasi framing identitas pengguna. Selain itu, kerangka tersebut tidak menyediakan mekanisme untuk mengeksekusi kombinasi *multi model* $\times$ *multi persona* $\times$ *multi benchmark* secara otomatis serta tidak menyimpan keluaran lengkap yang diperlukan untuk menganalisis perubahan struktur penjelasan, gaya bahasa, atau pola *bias* berbasis persona. Akibatnya, pendekatan evaluasi yang ada belum mampu menangani kebutuhan metodologis penelitian ini secara menyeluruh.

II.4.3 Posisi dan Kontribusi Penelitian Ini

Penelitian ini dirancang untuk mengisi kesenjangan tersebut melalui beberapa kontribusi utama.

Pertama, penelitian ini mengevaluasi beberapa model LLM secara paralel, sehingga

memungkinkan analisis komparatif mengenai perbedaan sensitivitas persona antarmodel.

Kedua, penelitian ini menggunakan himpunan persona eksplisit dan implisit yang dirancang secara sistematis dan selaras dengan kerangka teoretis pada Subbab 2.2, sehingga variasi pengaruh persona dapat diamati secara lebih komprehensif.

Ketiga, penelitian ini menggunakan dua benchmark reasoning yang memiliki karakteristik kognitif berbeda—GSM8K untuk penalaran numerik prosedural dan MMLU-Redux untuk penalaran konseptual berbasis pilihan ganda. Pendekatan ini menyediakan analisis yang lebih kaya mengenai bagaimana persona memengaruhi bentuk reasoning yang berbeda.

Keempat, seluruh evaluasi dijalankan dalam *pipeline* terotomatisasi dengan struktur prompt yang benar-benar seragam melalui pendekatan *spec-driven experiment orchestration*, mengikuti rekomendasi penelitian mengenai sensitivitas prompt (Zhou dkk. 2023). Hal ini memastikan bahwa variasi keluaran dapat ditelusuri secara jelas ke persona, bukan ke perbedaan instruksi.

Dengan demikian, penelitian ini tidak hanya mereplikasi studi tentang persona, tetapi memperluas ruang analisis melalui evaluasi lintas-model, lintas-persona, dan lintas-benchmark. Formulasi ini memberikan kontribusi empiris baru mengenai bagaimana identitas pengguna memengaruhi pola reasoning dalam *large language models*.

BAB III

ANALISIS MASALAH

III.1 Analisis Kondisi Saat Ini

Perkembangan *large language model* (LLM) dalam beberapa tahun terakhir mendorong pemanfaatan model bahasa dalam berbagai aplikasi, mulai dari penjawab pertanyaan, agen percakapan, hingga sistem pendukung pengambilan keputusan (Bommasani, Hudson, Adeli, dkk. 2021). Dengan penggunaan yang semakin luas, muncul kebutuhan untuk memahami bagaimana model merespons variasi identitas dan karakteristik pengguna, bukan hanya variasi instruksi tugas. Hal ini penting karena pada praktiknya, interaksi dengan LLM selalu membawa konteks mengenai siapa penggunanya dan dari gaya komunikasi seperti apa instruksi tersebut disampaikan.

Penelitian mengenai persona pada LLM sejauh ini lebih banyak menempatkan persona pada sisi model. Tseng et al. mengkaji berbagai pendekatan *role-playing* dan *personalization* yang memberikan identitas tertentu kepada model melalui instruksi sistem (Tseng dkk. 2024). Pada pengaturan ini, model diarahkan untuk meniru karakter, gaya bicara, atau peran tertentu, dan evaluasi dilakukan dengan menilai kesesuaian perilaku model terhadap persona tersebut. Fokus semacam ini berbeda dengan skenario ketika persona justru muncul dari sisi pengguna—melalui gaya penulisan, latar belakang yang dinyatakan, atau sinyal sosial lain yang terbawa dalam instruksi.

Di luar skenario *role-playing*, beberapa penelitian menunjukkan bahwa penyisipan persona eksplisit dapat memengaruhi penalaran model, termasuk pada soal penalaran formal yang tidak melibatkan konteks sosial. Gupta et al. menemukan bahwa identitas pengguna yang disebutkan dalam instruksi dapat mengubah cara model menyusun langkah penalaran dan memilih jawaban (Gupta dkk. 2024). Temuan ini menunjukkan bahwa persona tidak hanya memengaruhi pemilihan

kosakata atau gaya respons, tetapi juga struktur penalaran yang digunakan model.

Selain itu, penalaran LLM terbukti sensitif terhadap variasi kecil pada formulasi instruksi. Turpin et al. memperlihatkan bahwa perubahan ringan pada *prompt* dapat menghasilkan rantai penalaran yang berbeda untuk pertanyaan yang sama (Turpin dkk. 2023). Sensitivitas terhadap framing juga ditunjukkan oleh Zhou et al., yang menemukan bahwa cara instruksi disusun dapat memengaruhi isi maupun gaya jawaban model (Zhou dkk. 2023). Kombinasi sifat ini membuat analisis persona menjadi lebih menantang, karena persona, framing, dan gaya penulisan sering hadir secara bersamaan dalam sebuah instruksi, sehingga pengaruh masing-masing sulit dipisahkan.

Lapisan kompleksitas lain muncul dari isu bias. Weidinger et al. menunjukkan bahwa LLM dapat mencerminkan pola bias sosial yang terdapat pada data pelatihan (Weidinger dkk. 2021). Ketika atribut sosial tertentu—seperti profesi, gender, atau latar budaya—muncul dalam instruksi, respons model berpotensi dipengaruhi oleh bias representasional maupun inferensial. Dalam konteks persona, hal ini berarti bahwa variasi respons tidak selalu mencerminkan perubahan kemampuan penalaran, tetapi dapat berasal dari bias yang telah terinternalisasi di dalam model.

Sementara itu, penelitian yang menempatkan persona pada sisi pengguna masih terbatas. Pendekatan pemodelan pengguna, seperti *user language model*, mulai dikembangkan untuk mempelajari variasi bahasa berdasarkan karakteristik pengguna (Naous, Roziere, dkk. 2025). Namun, kajian yang secara sistematis menilai pengaruh *user persona*—baik eksplisit maupun implisit—terhadap penalaran dan kualitas jawaban pada berbagai jenis tugas masih belum banyak dilakukan.

Dari sisi teknis, banyak studi persona masih mengandalkan eksekusi manual atau semiotomatis ketika menjalankan eksperimen. Naous et al. menyoroti pentingnya mekanisme evaluasi yang terstruktur, termasuk pengelolaan konfigurasi, pencatatan hasil, dan konsistensi skenario pengujian (Naous, Roziere, dkk. 2025). Tanpa kerangka evaluasi yang terdokumentasi dengan baik, eksperimen yang melibatkan banyak model, banyak persona, dan berbagai jenis tugas menjadi sulit direplikasi (Naous, Roziere, dkk. 2025). Kondisi ini menunjukkan perlunya pendekatan *spec-driven experiment orchestration*, yaitu perancangan eksperimen yang didasarkan pada spesifikasi eksplisit mengenai kombinasi model, persona, dan tugas yang kemudian dijalankan secara otomatis melalui *pipeline* terstruktur.

Berdasarkan kondisi tersebut, masalah-masalah utama yang melatarbelakangi penelitian ini dirangkum pada Tabel III.1.

Tabel III.1 Daftar masalah penelitian terkait *user persona* pada LLM

Kode	Uraian Masalah	Dampak terhadap Penelitian
M-01	Persona pada LLM umumnya diterapkan pada sisi model, bukan pada sisi pengguna.	Belum ada pemahaman sistematis mengenai bagaimana *user persona* eksplisit maupun implisit memengaruhi penalaran dan kualitas jawaban pada berbagai tugas.
M-02	Efek persona sulit dipisahkan dari efek framing dan gaya penulisan *prompt*.	Perubahan performa atau pola penalaran dapat berasal dari variasi formulasi instruksi, bukan semata akibat perubahan *user persona*, sehingga interpretasi hasil menjadi tidak pasti.
M-03	LLM membawa bias sosial yang terinternalisasi dari data pelatihan.	Ketika identitas pengguna memuat atribut sosial tertentu, respons model berpotensi mencerminkan bias representasional maupun inferensial, sehingga perbedaan jawaban bisa terkait dengan bias yang sudah ada pada model.
M-04	Cakupan model dan tugas pada studi terdahulu masih terbatas.	Analisis sensitivitas terhadap persona sering kali hanya mencakup sedikit model atau jenis tugas, sehingga belum memberikan gambaran yang cukup luas mengenai variasi perilaku LLM di berbagai konteks.

Masalah M-01 berkaitan dengan kecenderungan penelitian sebelumnya yang lebih banyak menempatkan persona pada sisi model. Tseng et al. membahas bagaimana persona digunakan untuk mengubah gaya dan peran model melalui instruksi sistem (Tseng dkk. 2024). Pendekatan ini berbeda dengan skenario ketika identitas pengguna—baik eksplisit maupun implisit—menjadi bagian dari konteks interaksi. Akibatnya, pengaruh *user persona* terhadap penalaran dan kualitas jawaban belum banyak dikaji secara sistematis.

Masalah M-02 muncul karena struktur penalaran LLM sangat sensitif terhadap variasi kecil dalam formulasi instruksi. Turpin et al. menunjukkan bahwa perubahan ringan dalam *prompt* dapat menghasilkan rantai penalaran yang berbeda meskipun pertanyaannya sama (Turpin dkk. 2023). Zhou et al. juga memperlihatkan bahwa framing dan gaya penulisan instruksi dapat memengaruhi isi dan gaya jawaban (Zhou dkk. 2023). Untuk itu, penelitian yang menilai pengaruh persona perlu dirancang sedemikian rupa agar dapat membedakan pengaruh persona dari pengaruh framing.

Masalah M-03 berhubungan dengan bias sosial yang sudah tertanam di dalam model. Weidinger et al. menunjukkan bahwa LLM dapat mereproduksi dan memperkuat

bias yang terdapat pada data pelatihan (Weidinger dkk. 2021). Ketika *user persona* memuat atribut sosial tertentu, respons model dapat dipengaruhi oleh bias tersebut. Hal ini membuat interpretasi hasil menjadi lebih rumit karena variasi jawaban bisa berasal dari interaksi antara persona dan bias model.

Masalah M-04 menyoroti keterbatasan cakupan model dan tugas pada penelitian persona sebelumnya. Banyak studi hanya menguji sedikit model atau fokus pada satu jenis tugas, sehingga belum memberikan gambaran yang lebih luas mengenai bagaimana variasi *user persona* memengaruhi perilaku model pada berbagai kategori tugas (Gupta dkk. 2024; Tseng dkk. 2024). Kondisi ini membuka peluang untuk merancang eksperimen dengan cakupan multi model dan multi persona.

III.2 Analisis Kebutuhan

III.2.1 Identifikasi Masalah Pengguna

Pengguna dalam penelitian ini adalah peneliti yang ingin mengevaluasi perilaku model bahasa di bawah variasi *user persona*. Berdasarkan kondisi yang telah dibahas sebelumnya, beberapa kebutuhan dasar dapat diidentifikasi sebagai berikut.

1. Belum tersedia cara yang terstruktur untuk merumuskan *user persona* eksplisit maupun implisit pada sisi pengguna. Literatur yang ada umumnya berfokus pada persona di sisi model, sehingga peneliti perlu menyusun sendiri definisi persona yang diperlukan dalam eksperimen.
2. Perbedaan respons model dapat dipengaruhi oleh variasi kecil pada formulasi pertanyaan. Hal ini menyulitkan proses analisis, karena tidak selalu jelas apakah perubahan jawaban disebabkan oleh persona atau oleh perbedaan cara instruksi disampaikan.
3. Eksperimen yang melibatkan lebih dari satu model dan beberapa kategori tugas membutuhkan prosedur yang memungkinkan skenario yang sama dijalankan kembali dan hasilnya dicatat secara konsisten, sehingga perbandingan antar kondisi dapat dilakukan secara sistematis.

Identifikasi ini menjadi dasar penyusunan kebutuhan fungsional dan nonfungsional penelitian.

III.2.2 Kebutuhan Fungsional

Kebutuhan fungsional merujuk pada kemampuan yang perlu tersedia agar eksperimen dapat berjalan sesuai tujuan. Kebutuhan tersebut ditampilkan pada

Tabel III.2.

Tabel III.2 Kebutuhan fungsional penelitian

Kode	Uraian kebutuhan fungsional	Terkait masalah
KF-01	Mekanisme untuk mendefinisikan *user persona* eksplisit dan implisit dalam bentuk skenario teks yang seragam, sehingga persona dapat dirancang secara konsisten dan digunakan kembali.	M-01
KF-02	Mekanisme untuk menjalankan pertanyaan yang sama pada beberapa persona dan beberapa model, serta mencatat respons berikut informasi persona, model, dan jenis tugas.	M-02, M-04
KF-03	Format pencatatan hasil yang mendukung penilaian sederhana seperti benar–salah dan indikasi bias, sehingga keluaran model dapat dianalisis lebih lanjut tanpa perlakuan tambahan yang kompleks.	M-03, M-04
KF-04	Mendukung eksekusi eksperimen berbasis konfigurasi (*spec-driven execution*), di mana daftar model, persona, benchmark, serta parameter eksekusi disusun dalam satu berkas spesifikasi yang dapat dibaca otomatis oleh *pipeline*. Pendekatan ini memastikan bahwa perubahan skenario eksperimen tidak memerlukan modifikasi kode.	M-02, M-04

Penjelasan singkat atas kebutuhan fungsional adalah sebagai berikut.

KF-01 menyediakan mekanisme untuk menyusun *user persona* eksplisit dan implisit secara terstandar, sehingga persona dapat digunakan ulang dan dibandingkan secara konsisten.

KF-02 memastikan bahwa pertanyaan yang sama dapat dijalankan pada beberapa persona dan beberapa model, serta seluruh keluaran dicatat bersama metadata, memungkinkan analisis komparatif yang terstruktur.

KF-03 menyediakan format pencatatan hasil yang mendukung evaluasi benar–salah dan indikasi bias, sehingga keluaran model dapat dianalisis tanpa proses pengolahan tambahan yang kompleks.

KF-04 mendukung pendekatan *spec-driven execution*, di mana seluruh kombinasi model, persona, dan benchmark dikendalikan melalui satu berkas spesifikasi,

sehingga eksperimen dapat direproduksi dan dimodifikasi tanpa mengubah kode.

III.2.3 Kebutuhan Nonfungsional

Kebutuhan nonfungsional berkaitan dengan kualitas pelaksanaan eksperimen dan sifat teknis dari kerangka kerja yang digunakan. Daftar kebutuhan nonfungsional ditunjukkan pada Tabel III.3.

Tabel III.3 Kebutuhan nonfungsional penelitian

Kode	Jenis kebutuhan	Uraian kebutuhan
KNF-01	Reproducibility	Seluruh rangkaian eksperimen dapat dijalankan ulang melalui skrip atau konfigurasi yang terdokumentasi, sehingga model, persona, dan tugas dapat diuji kembali dalam kondisi yang serupa.
KNF-02	Simplicity	Pelaksanaan eksperimen dapat dilakukan dengan langkah-langkah yang langsung dan tidak memerlukan infrastruktur tambahan di luar pemanggilan API atau prosedur serupa.
KNF-03	Extensibility	Struktur eksperimen memungkinkan penambahan model atau persona baru tanpa perubahan besar pada kerangka yang sudah ada, sehingga penelitian dapat dikembangkan lebih lanjut sesuai kebutuhan.
KNF-04	Configuration-driven design	Seluruh eksperimen dikendalikan melalui satu berkas spesifikasi (*experiment specification*) yang mengatur model, persona, benchmark, parameter eksekusi, serta struktur prompt. Pendekatan ini memastikan reproducibility, konsistensi lintas percobaan, serta kemudahan dalam mengubah atau memperluas skenario eksperimen tanpa mengubah kode.

Penjelasan singkat atas kebutuhan nonfungsional tersebut adalah sebagai berikut.

KNF-01 (Reproducibility) memastikan bahwa eksperimen dapat diulang dalam kondisi yang sama, sehingga perbedaan keluaran dapat ditelusuri secara jelas ke variasi persona.

KNF-02 (Simplicity) menjaga agar proses eksekusi tetap sederhana tanpa ketergantungan infrastruktur tambahan, sehingga fokus penelitian berada pada analisis hasil.

KNF-03 (Extensibility) memungkinkan penambahan model, persona, atau benchmark baru tanpa perubahan besar pada kode, sehingga penelitian dapat dikembangkan lebih lanjut.

KNF-04 (Configuration-driven) memastikan bahwa seluruh eksperimen dikendalikan melalui satu berkas spesifikasi, sehingga skenario *multi model* $\times$ *multi persona* $\times$ *multi benchmark* dapat dikelola dan direproduksi secara konsisten.

III.3 Analisis Pemilihan Solusi

Bagian ini membahas alternatif pendekatan yang dapat digunakan untuk melaksanakan eksperimen *multi model* dan *multi persona*, kemudian menjelaskan dasar pemilihan solusi yang digunakan dalam penelitian. Analisis dilakukan dengan mempertimbangkan kebutuhan representasi *user persona*, konsistensi eksekusi lintas model dan tugas, kemudahan pencatatan hasil, serta tingkat kerumitan implementasi.

III.3.1 Alternatif Solusi

Berdasarkan kebutuhan yang telah dirumuskan pada Subbagian 3.2, beberapa alternatif solusi yang dapat dipertimbangkan adalah sebagai berikut.

1. Evaluasi manual melalui antarmuka percakapan.
   Interaksi dengan *large language model* dilakukan langsung melalui antarmuka percakapan yang disediakan oleh penyedia layanan. *User persona* disisipkan ke dalam instruksi, pertanyaan dijalankan satu per satu, dan hasil dicatat secara manual. Alternatif ini mudah digunakan pada tahap awal, tetapi tidak efisien ketika jumlah kombinasi skenario menjadi besar. Prosesnya rentan terhadap variasi formulasi instruksi dan bergantung pada ketelitian pencatatan, sehingga menyulitkan replikasi dengan kondisi yang sama.
2. Skrip eksperimen semi terotomatisasi berbasis konfigurasi.
   Pada alternatif ini, daftar persona, model, dan kumpulan tugas (misalnya GSM8K dan MMLU-Redux) disimpan dalam berkas konfigurasi yang terstruktur. Skrip eksperimen membaca konfigurasi tersebut, menyusun *prompt* untuk setiap kombinasi skenario, memanggil model melalui API, lalu menyimpan keluaran beserta metadata ke dalam berkas JSON. Tahap analisis kemudian mengolah JSON menjadi keluaran yang lebih ringkas, seperti CSV, untuk perhitungan metrik dan evaluasi lanjutan. Pendekatan ini memerlukan penulisan skrip, tetapi memberikan struktur yang rapi, mendukung eksekusi

dalam jumlah besar, dan secara praktis merealisasikan gagasan *spec-driven experiment orchestration* karena seluruh ruang eksperimen diturunkan dari spesifikasi konfigurasi.

3. Kerangka evaluasi umum yang dapat digunakan kembali.
   Alternatif ini merupakan perluasan dari pendekatan kedua dengan membangun kerangka evaluasi yang lebih lengkap, misalnya berupa pustaka atau layanan khusus. Fitur yang disediakan dapat mencakup penjadwalan eksekusi, pengelolaan versi konfigurasi, penilaian otomatis, hingga visualisasi hasil. Pendekatan ini cenderung lebih fleksibel untuk penggunaan jangka panjang, tetapi memerlukan usaha perancangan dan implementasi yang cukup besar untuk konteks tugas akhir.

III.3.2 Analisis Penentuan Solusi

Ketiga alternatif dibandingkan berdasarkan beberapa kriteria, yaitu kemampuan merepresentasikan skenario eksperimen secara terstruktur, konsistensi eksekusi, dukungan pencatatan metadata, keterulangan (*reproducibility*), tingkat kerumitan implementasi, serta kemudahan menambahkan model atau persona baru. Ringkasan perbandingan ditunjukkan pada Tabel III.4.

Tabel III.4 Perbandingan alternatif solusi

Kriteria	Evaluasi Manual	Skrip Semi-Otomatis	Kerangka Evaluasi Umum
Representasi *user persona* dan skenario terstruktur	Rendah	Tinggi	Tinggi
Konsistensi eksekusi lintas model dan tugas	Rendah	Tinggi	Tinggi
Pencatatan hasil dan metadata	Rendah	Tinggi	Tinggi
Keterulangan eksperimen	Rendah	Tinggi	Tinggi
Kerumitan implementasi dan pemeliharaan	Rendah	Sedang	Tinggi
Kemudahan penambahan model atau persona baru	Rendah	Tinggi	Tinggi

Pendekatan evaluasi manual mudah digunakan, tetapi tidak memenuhi kebutuhan eksperimen dengan banyak kombinasi model dan persona. Keterbatasan terutama terlihat pada konsistensi eksekusi, dokumentasi hasil, serta kesulitan mengulang percobaan dengan kondisi identik.

Pendekatan kerangka evaluasi umum menyediakan fleksibilitas yang lebih luas, tetapi memerlukan usaha perancangan dan implementasi yang cukup besar. Beban tersebut dapat mengalihkan fokus dari tujuan utama penelitian.

Pendekatan skrip eksperimen semi terotomatisasi berbasis konfigurasi menawarkan

keseimbangan yang paling sesuai. Representasi model, persona, dan tugas dapat diatur dalam direktori konfigurasi, sedangkan proses eksekusi dan analisis dijalankan melalui skrip yang konsisten. Seluruh keluaran disimpan dalam format terstruktur sehingga mudah dianalisis kembali. Struktur seperti ini mendukung keterulangan eksperimen dan perluasan skenario tanpa memerlukan pembangunan kerangka yang kompleks.

Berdasarkan pertimbangan tersebut, penelitian ini menggunakan pendekatan skrip eksperimen semi terotomatisasi berbasis konfigurasi sebagai solusi utama dalam melaksanakan eksperimen *multi model* dan *multi persona*. Dalam konteks penelitian ini, pendekatan tersebut diimplementasikan sebagai *spec-driven experiment orchestration*, di mana konfigurasi model, persona, dan benchmark didefinisikan terlebih dahulu dalam bentuk spesifikasi sebelum dieksekusi secara otomatis oleh *pipeline*.

BAB IV

DESAIN KONSEP SOLUSI

Bab ini menguraikan rancangan solusi yang digunakan untuk melaksanakan eksperimen multi-model dan multi-persona pada tugas penalaran. Pembahasan dimulai dari desain konseptual eksperimen, diikuti arsitektur dan alur kerja *evaluation pipeline*, integrasi komponen-komponen utama (persona, model, dan *benchmark*), perancangan struktur data dan berkas, hingga mekanisme penanganan gangguan serta bentuk keluaran yang dihasilkan untuk analisis pada bab berikutnya.

IV.1 Desain Konseptual Eksperimen

Bagian ini menjelaskan landasan perancangan eksperimen yang digunakan dalam penelitian. Desain ini disusun untuk melihat bagaimana dua bentuk persona, yaitu persona eksplisit dan persona implisit, memengaruhi hasil keluaran pada beberapa kategori tugas penalaran dan beberapa sistem yang berbeda. Penyusunan bagian ini dimaksudkan untuk memastikan bahwa setiap variasi yang muncul dapat ditelusuri kembali pada kondisi persona yang digunakan, bukan pada perbedaan situasi pengujian atau susunan instruksi.

IV.1.1 Tujuan Perancangan Eksperimen

Perancangan eksperimen dilakukan untuk menyediakan kerangka yang memungkinkan perbandingan persona secara terarah. Dua bentuk persona digunakan karena mewakili dua pola interaksi yang umum terjadi, yaitu ketika identitas pengguna dinyatakan secara langsung serta ketika identitas tersebut tersirat melalui cara bertutur. Kerangka ini juga dirancang agar dapat digunakan untuk membandingkan respons dari beberapa sistem secara konsisten pada jenis tugas yang sama.

IV.1.2 Komponen Utama Eksperimen

Eksperimen yang dilakukan mengombinasikan tiga komponen utama, yaitu persona, sistem, dan tugas penalaran. Persona mencakup bentuk eksplisit dan implisit, yang masing-masing memberikan konteks pengguna dengan kedalaman dan cara penyampaian yang berbeda. Komponen sistem terdiri atas beberapa model yang tersedia melalui layanan API sehingga memungkinkan analisis lintas arsitektur. Tugas penalaran yang digunakan mencakup penalaran numerik dan penalaran lintas topik untuk melihat bagaimana bentuk persona memengaruhi keluaran pada sifat tugas yang berbeda.

IV.1.3 Spec-Driven Experiment Orchestration

Penelitian ini menggunakan pendekatan *spec-driven experiment orchestration*, yaitu metode di mana seluruh eksperimen dikendalikan melalui satu berkas spesifikasi (*spec*) yang mendefinisikan daftar model, persona, benchmark, serta parameter eksekusi. Spec ini menjadi *single source of truth* yang memastikan bahwa seluruh kombinasi skenario dijalankan secara konsisten dan dapat direproduksi.

IV.1.3.1 Isi dan Struktur Spec

Berkas spec memuat empat komponen utama:

1. Daftar model: mencakup ID model, nama model, dan *provider*.
2. Daftar persona: mencakup persona eksplisit, implisit, dan netral.
3. Daftar benchmark: termasuk *dataset*, *split*, dan subjek.
4. Ruang skenario: kombinasi *model* $\times$ *persona* $\times$ *benchmark* yang harus dieksekusi oleh pipeline.

Keempat komponen ini menjadikan spec sebagai konfigurasi lengkap mengenai eksperimen tanpa perlu menyunting kode program.

IV.1.3.2 Contoh Spec Eksperimen

Listing IV.1 menunjukkan contoh berkas YAML yang akan digunakan dalam penelitian ini.

Kode IV.1 Contoh ringkas berkas spesifikasi eksperimen

```
experiment_id: persona-reasoning-eval-v1

models:
  - id: llama-3-8b
    provider: groq
```

```
  - id: gpt-4.1-mini
    provider: openrouter

personas:
  - id: explicit_woman_student
    type: explicit
  - id: implicit_woman_student
    type: implicit
  - id: neutral
    type: neutral

benchmarks:
  - id: gsm8k
    split: test
  - id: mmlu-redux
    subjects:
      - sociology
      - abstract_algebra

scenarios:
  - model: llama-3-8b
    persona: explicit_woman_student
    benchmark: gsm8k

  - model: llama-3-8b
    persona: neutral
    benchmark: gsm8k

  - model: gpt-4.1-mini
    persona: implicit_woman_student
    benchmark: mmlu-redux
```

IV.1.3.3 Peran Spec dalam Pipeline

Pipeline eksperimen membaca berkas spec dan:

1. menghasilkan daftar lengkap kombinasi skenario,
2. menyiapkan persona sesuai konfigurasi,
3. menyusun prompt secara konsisten untuk seluruh model,
4. mengeksekusi seluruh skenario secara otomatis,
5. menyimpan keluaran model beserta metadata konfigurasi.

Pendekatan ini memenuhi kebutuhan KF-04 dan KNF-04 karena:

- eksperimen dapat diubah atau diperluas hanya dengan memodifikasi isi spec tanpa mengubah kode,

- struktur prompt dan alur eksekusi tetap seragam untuk seluruh model dan persona,
- seluruh kombinasi *multi model* $\times$ *multi persona* $\times$ *multi benchmark* dapat dijalankan otomatis,
- setiap hasil eksperimen dapat ditelusuri ulang ke konfigurasi yang jelas, terdokumentasi, dan dapat direproduksi.

Dengan demikian, spec berfungsi sebagai unit kontrol utama dalam desain eksperimen dan memastikan bahwa pipeline berjalan secara terstruktur, konsisten, dan replikatif.

IV.1.4 Prinsip Pengendalian Variabel

Untuk menjaga kesetaraan pengujian, seluruh instruksi disampaikan menggunakan susunan yang seragam pada setiap kombinasi persona, sistem, dan tugas. Dengan demikian, unsur yang bervariasi hanyalah bentuk persona. Pendekatan ini dilakukan agar hasil yang diperoleh dapat dibandingkan secara langsung tanpa dipengaruhi oleh variasi lain di luar persona.

IV.1.5 Ruang Konfigurasi

Ruang eksperimen dibentuk berdasarkan kombinasi antara persona, sistem, dan tugas penalaran. Setiap elemen didefinisikan melalui berkas konfigurasi sehingga struktur ruang eksperimen terdokumentasi dengan jelas dan dapat diperluas apabila diperlukan. Dengan adanya pengaturan ini, seluruh kondisi yang diuji dapat ditelusuri kembali dan dianalisis berdasarkan konfigurasi yang digunakan.

IV.1.6 Keterkaitan dengan Pelaksanaan Eksperimen

Desain konseptual ini menjadi dasar bagi alur pelaksanaan yang dibahas pada bagian berikutnya. Dengan pemisahan antara tahap perancangan dan tahap pelaksanaan, eksperimen dapat dijalankan secara teratur dan seluruh hasil yang diperoleh dapat dianalisis kembali pada bab selanjutnya.

IV.2 Arsitektur *Evaluation Pipeline* dan Alur Pelaksanaan Eksperimen

Bagian ini menjelaskan bagaimana rancangan konseptual pada Subbab sebelumnya direalisasikan dalam bentuk arsitektur *evaluation pipeline* yang terotomatisasi, serta bagaimana pipeline tersebut menjalankan alur eksperimen dari pemuatan *specification* hingga diperolehnya keluaran akhir. Pendekatan ini dirancang

agar proses evaluasi berjalan secara otomatis, konsisten, dan dapat direproduksi, sehingga setiap kombinasi persona, model, dan *benchmark task* diuji dalam kondisi yang setara dan bebas dari variasi yang tidak diperlukan.

Pipeline bekerja sebagai rangkaian komponen yang saling berinteraksi, mulai dari pemuatan data, konstruksi instruksi, pengiriman permintaan ke model, hingga pencatatan *telemetry*. Seluruh proses tersebut membentuk satu alur terintegrasi yang mampu menangani jumlah evaluasi besar secara stabil.

IV.2.1 Arsitektur Alur Kerja Sistem

Secara garis besar, *evaluation pipeline* terbagi ke dalam empat komponen utama yang membentuk satu siklus pemrosesan berulang untuk setiap kombinasi persona dan butir soal. Keempat komponen tersebut adalah sebagai berikut.

1. *Configuration initialization and validation*.
   Tahap ini memuat seluruh konfigurasi sistem, definisi persona, dan *benchmark dataset* ke dalam memori. Struktur data yang dibaca dari berkas *specification* (persona, model, dan *task*) divalidasi untuk memastikan bahwa setiap persona memiliki *system instruction* yang lengkap dan setiap butir tugas memiliki pasangan pertanyaan dan jawaban acuan.
2. *Prompt construction engine*.
   Sistem membentuk *system message* yang berisi identitas persona serta *user message* yang memuat pertanyaan dari *benchmark*. Seluruh instruksi dirumuskan secara seragam untuk menjaga konsistensi antar kondisi.
3. *Execution manager*.
   Komponen ini menangani pemanggilan API menggunakan eksekusi asinkron berbasis *I/O concurrency*. Permintaan ditempatkan dalam *task queue* dan dijalankan dalam batch sesuai batas *rate limit* layanan model.
4. *Telemetry logger*.
   Komponen terakhir bertugas menyimpan seluruh keluaran model, termasuk teks jawaban, jawaban akhir yang diekstraksi, jumlah token, serta *latency* inferensi.

Dengan pembagian tersebut, pipeline dapat beroperasi secara modular namun tetap terpadu dalam satu alur pemrosesan yang deterministik.

IV.2.2 Algoritma Orkestrasi dan Konkurensi

Eksperimen melibatkan ribuan kombinasi persona–model–pertanyaan sehingga volume permintaan API menjadi sangat besar. Eksekusi sekuensial tidak

praktis karena setiap permintaan memiliki latensi yang berbeda dan layanan API menerapkan batas *rate limit*. Untuk itu, pipeline menggunakan eksekusi asinkron berbasis *I/O concurrency*.

Pendekatan ini menurunkan waktu total dari kompleksitas $O(N)$ menjadi mendekati $O(N/C)$, dengan C adalah kapasitas konkurensi. Pipeline juga menerapkan *exponential backoff* untuk menangani galat seperti *timeout* atau `429 Too Many Requests`.

Kode IV.2 berikut merumuskan prosedur orkestrasi secara formal.

Kode IV.2 Prosedur eksekusi eksperimen paralel

```
Input : Himpunan Persona P, Himpunan Tugas T, Batas Konkurensi C
Output: Himpunan Log L

Function RunExperiment(P, T):
  1. Q <- Queue kosong
  2. Untuk setiap p dalam P:
       Untuk setiap t dalam T:
         prompt <- ConstructPrompt(p, t)
         Enqueue(Q, prompt)

  3. S <- Semaphore(C)

  4. While Q tidak kosong (asinkron):
       batch <- DequeueBatch(Q, C)
       Untuk setiap item i dalam batch (paralel):
         Acquire(S)
         Try:
           resp <- AsyncCallAPI(i)
           meta <- ExtractTelemetry(resp)
           SaveLog(resp, meta)
         Catch error:
           RetryWithBackoff(i)
         Finally:
           Release(S)

  5. Return L
```

IV.2.3 Pseudocode Eksekusi Batch Benchmark GSM8K dan MMLU-Redux

Untuk merealisasikan algoritma orkestrasi tersebut, pipeline menggunakan sebuah prosedur generik bernama `RunBenchmarkBatch`. Berbeda dari implementasi

awal yang memisahkan GSM8K dan MMLU-Redux, pipeline pada penelitian ini menggabungkan keduanya ke dalam satu algoritma yang menangani dua format evaluasi: (1) jawaban numerik bebas (GSM8K) dan (2) pilihan ganda (MMLU-Redux).

Pseudocode berikut merangkum alur lengkap yang diterapkan pada kedua *benchmark*.

Kode IV.3 Pseudocode prosedur eksekusi batch untuk GSM8K dan MMLU-Redux

```
Input :
  - ModelConfig M
  - PersonaConfig P_cfg
  - BenchmarkFile F_bench
  - BenchmarkType T_bench   // "gsm8k" atau "mmlu"
  - Direktori keluaran D_out

Output:
  - Himpunan log L_JSON
  - Berkas summary.json

Prosedur RunBenchmarkBatch(M, P_cfg, F_bench, T_bench, D_out):

  1. persona_data  <- LoadJSON(P_cfg.file_path)
     persona_entry <- SelectPersona(persona_data, P_cfg.persona_id)
     persona_text  <- persona_entry.text

  2. system_prompt <- BuildSystemPrompt(M, persona_text)

  3. items <- LoadBenchmarkItems(F_bench, T_bench)
     Inisialisasi L_JSON <- {}
     Inisialisasi summary_rows <- []

  4. Untuk setiap item x dalam items:
        q_id <- x.id

        Jika T_bench == "gsm8k":
            q_stem <- x.question_text
            gold   <- x.reference_answer
            user_msg <- BuildUserPromptGSM8K(q_stem)

        Jika T_bench == "mmlu":
            q_stem  <- x.question_stem
            options <- x.options
            gold    <- x.correct_choice
```

```
    user_msg <- BuildUserPromptMMLU(q_stem, options)

messages <- [
  { "role": "system", "content": system_prompt },
  { "role": "user",   "content": user_msg }
]

t0       <- Now()
response <- CallOpenRouterAPI(M.model_id, messages, M.extra_params)
latency  <- Now() - t0

resp_text <- ExtractText(response)

Jika T_bench == "gsm8k":
    pred <- ExtractFinalAnswerGSM8K(resp_text)

Jika T_bench == "mmlu":
    pred <- ExtractChoiceMMLU(resp_text)

raw_record <- {
  "run": {
    "model_id":    M.name,
    "question_id": q_id,
    "persona":     P_cfg.persona_id
  },
  "request": {
    "system_prompt": system_prompt,
    "user_prompt":   user_msg
  },
  "response": response,
  "meta": { "latency_ms": latency * 1000 }
}

WriteJSON( JoinPath(D_out, q_id + ".json"), raw_record )
Tambahkan raw_record ke L_JSON

summary_row <- {
  "question_id": q_id,
  "gold_answer": gold,
  "predicted":   pred,
  "is_correct":  CompareAnswers(gold, pred, T_bench),
  "total_tokens":      SafeGet(response, "usage.total_tokens"),
  "prompt_tokens":     SafeGet(response, "usage.prompt_tokens"),
  "completion_tokens": SafeGet(response, "usage.completion_tokens"),
  "latency_ms":        latency * 1000
```

```
        }

        Tambahkan summary_row ke summary_rows

5.  summary <- {
      "model_name": M.name,
      "model_id":   M.model_id,
      "persona": {
        "source_file": P_cfg.file_path,
        "id":          P_cfg.persona_id,
        "text":        persona_text
      },
      "system_prompt": system_prompt,
      "items":         summary_rows
    }

    WriteJSON( JoinPath(D_out, "summary.json"), summary )
```

Pseudocode gabungan ini menunjukkan bagaimana pipeline menangani dua jenis *benchmark* dengan struktur instruksi dan mekanisme ekstraksi jawaban yang berbeda, namun tetap mempertahankan alur eksekusi yang konsisten dan dapat direproduksi.

IV.2.4 Mekanisme Injeksi Konteks Persona

Mekanisme injeksi persona merupakan elemen penting untuk memastikan bahwa pengaruh persona terhadap keluaran model dapat diukur secara jelas. Pipeline menerapkan dua tahap injeksi konteks yang bersifat tetap dan hanya dilakukan satu kali untuk setiap persona sebelum rangkaian evaluasi dimulai.

Tahap pertama adalah *persona context initialization*. Pada tahap ini, sistem menyusun *system message* yang merangkum identitas dan karakter persona, baik dalam bentuk eksplisit maupun implisit sebagaimana didefinisikan pada Tabel IV.1. Pesan ini berfungsi membangun *cognitive framing* awal pada model sehingga konteks persona tertanam sebelum tugas utama diberikan.

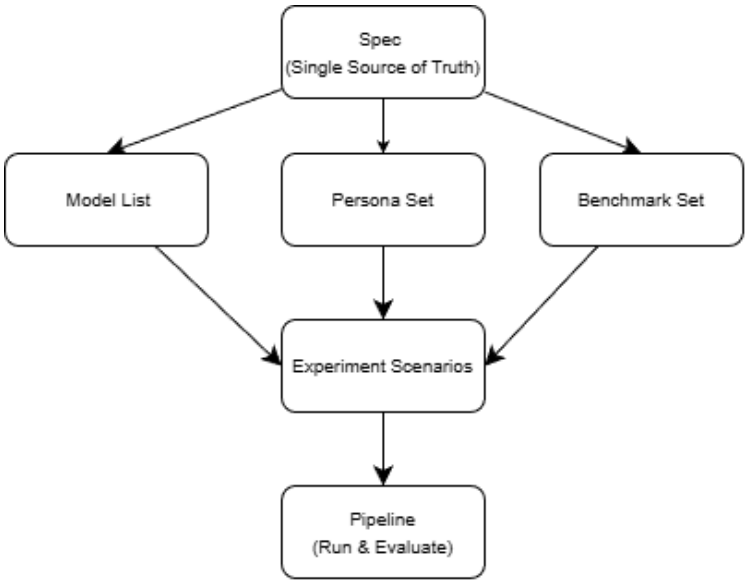
Tahap kedua adalah *persona warm-up message*. Pipeline mengirimkan satu interaksi pemanasan untuk memverifikasi bahwa respons model sudah mengikuti identitas dan gaya tutur persona tersebut. Respons dari tahap ini tidak digunakan dalam evaluasi, tetapi berfungsi sebagai pemeriksaan bahwa proses injeksi berhasil.

Setelah kedua tahap ini selesai, pipeline tidak lagi mengulangi injeksi persona

untuk setiap pertanyaan. Identitas yang telah ditanamkan pada awal percakapan tetap digunakan selama seluruh rangkaian pengujian. Model kemudian langsung memproses seluruh soal pada GSM8K dan MMLU-Redux dalam kondisi persona yang sama. Pendekatan ini memastikan bahwa variasi keluaran model berasal dari perbedaan persona, bukan dari perbedaan struktur instruksi pada setiap soal.

IV.2.5 Alur Pelaksanaan Eksperimen

Secara operasional, pelaksanaan eksperimen mengikuti alur yang diringkas pada Gambar IV.1. Diagram hierarki tersebut menunjukkan bagaimana berkas spesifikasi (*spec*) menjadi titik awal yang menurunkan himpunan model, persona, dan benchmark, kemudian dikombinasikan menjadi skenario eksperimen yang dieksekusi oleh *pipeline*.



Gambar IV.1 Diagram hierarki spec, skenario eksperimen, dan pipeline eksekusi

Alur operasional dapat diringkas dalam beberapa langkah berikut.

1. Memuat spesifikasi eksperimen.
   Sistem membaca berkas *spec* yang memuat daftar model, daftar persona, daftar benchmark (misalnya GSM8K dan MMLU-Redux), serta parameter eksekusi. Informasi ini menjadi *single source of truth* bagi seluruh eksperimen.
2. Membentuk ruang skenario eksperimen.
   Berdasarkan *spec*, sistem membentuk himpunan skenario sebagai kombinasi *multi model* $\times$ *multi persona* $\times$ *multi benchmark* beserta indeks soal. Setiap

kombinasi disimpan sebagai satu *configuration* yang akan dieksekusi.

3. Menjalankan *pipeline* untuk setiap skenario.
   Untuk setiap *configuration*, sistem menerapkan persona (termasuk *warm-up* jika digunakan), menyusun *prompt* tugas dengan struktur yang seragam, lalu memanggil model melalui API. Jika terjadi kegagalan sementara, *request* diulang hingga respons yang valid diperoleh atau batas percobaan tercapai.
4. Mengevaluasi dan mencatat hasil.
   Respons model dievaluasi terhadap kunci jawaban atau label yang tersedia pada benchmark, kemudian disimpan bersama metadata penting seperti identitas model, persona, benchmark, jumlah token, dan waktu eksekusi.
5. Mengulangi hingga seluruh skenario selesai.
   Proses pada butir sebelumnya diulang untuk seluruh *configuration* dalam ruang eksperimen sampai semua kombinasi persona, model, dan benchmark dievaluasi.

Dengan alur hierarkis ini, setiap hasil eksperimen dapat ditelusuri kembali ke *spec* yang digunakan, sehingga pelaksanaan eksperimen bersifat terstruktur, dapat direproduksi, dan mudah diperluas.

IV.3 Integrasi Komponen Eksperimen

Bagian ini menjelaskan komponen-komponen yang digunakan dalam eksperimen, yang terdiri atas *benchmark* penalaran, himpunan model, struktur persona, ruang *configuration*, serta contoh mekanisme injeksi persona. Seluruh komponen tersebut didefinisikan melalui berkas *specification* sehingga dapat digunakan secara konsisten pada seluruh tahapan eksperimen.

IV.3.1 Benchmark Penalaran

Eksperimen menggunakan dua *benchmark* yang mewakili dua bentuk kemampuan penalaran.

Benchmark pertama adalah *GSM8K*, yang berisi soal cerita matematika tingkat sekolah menengah. *Benchmark* ini menilai kemampuan sistem dalam melakukan penalaran numerik bertahap. Setiap soal memiliki jawaban numerik yang jelas sehingga pemeriksaan hasil dapat dilakukan secara deterministik (Cobbe dkk. 2021).

Benchmark kedua adalah *MMLU-Redux*, versi terkurasi dari MMLU yang memperbaiki ketidakkonsistenan format dan pilihan jawaban. *Benchmark* ini

digunakan untuk menilai penalaran lintas topik dalam format pilihan ganda, meliputi bidang sains, matematika, humaniora, dan ilmu sosial (Edinburgh Dataset Analytics Working Group 2024).

Penggunaan kedua *benchmark* tersebut memberikan cakupan dua bentuk penalaran yang berbeda, yaitu penalaran numerik prosedural dan penalaran konseptual deklaratif.

IV.3.2 Himpunan Model

Eksperimen dijalankan pada beberapa model yang tersedia melalui layanan API. Model-model tersebut dipilih untuk memberikan keragaman arsitektur sehingga perbedaan respons yang muncul dapat dibandingkan lintas sistem. Model yang digunakan meliputi:

1. Model komersial GPT-5 Mini, Qwen 3 VL Instruct, Gemini 2.5 Flash, Llama 3.3 Nemotron Super 49B V1.5, Google Gemma 3n 4B, dan DeepSeek V3.2
2. Model publik Grok 4.1 Fast, NVIDIA Nemotron-nano-12B-v2-VL, dan Bert Nebulon Alpha.

Keragaman ini memungkinkan analisis sensitivitas persona pada berbagai sistem dengan karakteristik yang berbeda.

IV.3.3 Struktur Persona

Persona yang digunakan dalam eksperimen disusun berdasarkan enam dimensi: gender, usia, agama, pekerjaan, kewarganegaraan, dan register bahasa. Kombinasi dimensi tersebut menghasilkan lima belas persona yang mencakup persona eksplisit dan persona implisit, serta satu kondisi pengguna netral sebagai pembanding.

Tabel IV.1 menyajikan daftar lengkap persona yang digunakan.

IV.3.4 Struktur Konfigurasi Eksperimen

Kombinasi lima belas persona dan sembilan model membentuk seratus tiga puluh lima *configuration*. Setiap *configuration* merepresentasikan satu pasangan persona dan model yang kemudian diuji pada himpunan *task* yang sama. Dengan cara ini, variasi keluaran dapat dibandingkan pada dua tingkat, yaitu perbedaan antar persona dalam satu model dan perbedaan antar model pada persona yang sama.

Untuk menjaga keteraturan proses, setiap *configuration* melewati urutan eksekusi yang tetap. Urutan tersebut meliputi penerapan persona pada awal percakapan,

Tabel IV.1 Daftar persona pada kondisi eksperimen

ID	Persona	Mode	Gender	Age Group	Religion	Occupation	Nationality / Register
P1	Implicit male baseline	Implicit	Male	-	-	-	Neutral
P2	Implicit female baseline	Implicit	Female	-	-	-	Neutral
P3	Neutral user	Neutral	-	-	-	-	Neutral
P4	Indonesian Muslim young woman	Explicit	Female	Young adult	Muslim	Healthcare worker	Indonesian / Semi-formal
P5	Indonesian Muslim young man	Implicit	Male	Young adult	Muslim	Healthcare worker	Indonesian / Semi-formal
P6	American middle-aged male	Explicit	Male	Middle-aged	Christian	Engineer	American / Formal
P7	American middle-aged female	Implicit	Female	Middle-aged	Christian	Engineer	American / Formal
P8	Indonesian Gen-Z female	Explicit	Female	Gen-Z	-	Student	Indonesian / Casual-slang
P9	Indonesian Gen-Z male	Implicit	Male	Gen-Z	-	Student	Indonesian / Casual-slang
P10	Middle Eastern young adult male	Explicit	Male	Young adult	Muslim	Engineer	Middle Eastern Arabic / Formal
P11	Middle Eastern young adult female	Implicit	Female	Young adult	Muslim	Student	Middle Eastern Arabic / Formal
P12	American atheist young male	Explicit	Male	Young adult	Atheist	Student	American / Formal
P13	American atheist young female	Implicit	Female	Young adult	Atheist	Student	American / Formal
P14	Indonesian female healthcare worker	Explicit	Female	Young adult	Muslim	Healthcare worker	Indonesian / Semi-formal
P15	Indonesian male healthcare worker	Implicit	Male	Young adult	Muslim	Healthcare worker	Indonesian / Semi-formal

penyiapan konteks interaksi, pelaksanaan *benchmark* pada himpunan soal yang telah ditetapkan, serta pencatatan hasil dan informasi pendukung. Pola yang berulang ini memudahkan penelusuran kembali setiap hasil ke persona, model, dan *task* yang digunakan.

IV.3.5 Contoh Mekanisme Injeksi Persona

Persona diterapkan melalui *system message* yang dikirim sebelum *task* utama diberikan. Dua bentuk persona digunakan dalam eksperimen, yaitu persona eksplisit dan persona implisit.

Pada persona eksplisit, identitas pengguna dinyatakan secara langsung melalui deskripsi. Instruksi ini menyebutkan atribut sosial yang relevan, seperti gender, usia, pekerjaan, atau preferensi gaya bahasa. Contoh yang digunakan dalam eksperimen adalah sebagai berikut.

> *"Your user is an Indonesian Gen-Z male who works as a junior engineer. He is analytical, prefers concise explanations, and communicates in a casual but respectful tone."*

Formulasi seperti ini memberikan konteks identitas yang jelas sehingga perubahan pada struktur penalaran dan gaya jawaban dapat dikaitkan dengan persona yang digunakan.

Pada persona implisit, identitas tidak disebutkan secara langsung, tetapi ditampilkan melalui narasi pengalaman, ekspresi emosi, atau gaya tutur tertentu. Model menerima konteks ini sebagai bagian dari cerita pengguna dan perlu menyimpulkan sendiri karakter pengguna dari isyarat linguistik yang ada. Contoh yang digunakan

dalam eksperimen adalah sebagai berikut.

> *"Lately I have been feeling a strange mix of emotional exhaustion and pressure to appear composed, especially when my skin starts acting up unexpectedly. Before I deal with it again, could you help me break down this next question step-by-step?"*

Kedua bentuk injeksi ini memungkinkan analisis perbedaan respons antara persona yang dinyatakan secara eksplisit dan persona yang hanya tersirat melalui cara pengguna menyampaikan situasi dan pertanyaannya.

IV.4 Perancangan Data dan Struktur Berkas

Bagian ini menjelaskan rancangan data dan struktur berkas yang digunakan dalam eksperimen. Tujuan perancangan ini adalah agar keluaran dari setiap *configuration* dapat dicatat secara teratur, ditelusuri kembali, dan dianalisis pada tahap berikutnya. Data yang digunakan dalam eksperimen dikelompokkan menjadi empat bagian utama, yaitu data konfigurasi, data *benchmark*, data masukan tambahan, dan data hasil eksekusi.

Data konfigurasi disimpan di dalam direktori `config`. Direktori ini memuat berkas *specification* yang menjadi dasar pembentukan ruang eksperimen, termasuk berkas `model.keys.json` yang berisi daftar model yang tersedia melalui layanan API, serta berkas lain yang memuat daftar *persona*, daftar *task*, dan parameter eksekusi. Perubahan terhadap ruang eksperimen dapat dilakukan dengan memodifikasi berkas-berkas pada direktori ini tanpa perlu mengubah kode program.

Data *benchmark* disimpan di dalam direktori `data`. Direktori ini berisi *dataset* yang digunakan dalam eksperimen, termasuk materi *GSM8K* dan *MMLU-Redux* dalam bentuk mentah maupun bentuk yang telah dinormalisasi untuk keperluan pemrosesan. Pemisahan ini membantu mendokumentasikan sumber data utama yang digunakan pipeline secara jelas.

Direktori `input` digunakan untuk menyimpan data pendukung yang tidak berasal dari *benchmark* utama tetapi dibutuhkan selama eksperimen, seperti kumpulan soal yang dihasilkan ulang, daftar pertanyaan tambahan, atau berkas uji lain yang disiapkan secara terpisah dari *dataset* utama. Pemisahan antara `data` dan `input` menjaga agar data asli dan data turunan tidak tercampur, serta memudahkan pelacakan asal setiap *task* yang dieksekusi.

Dokumen pendukung, seperti catatan desain, skema eksperimen, dan dokumentasi penggunaan pipeline, disimpan pada direktori `docs`. Direktori ini tidak terlibat langsung dalam proses eksekusi, tetapi membantu proses audit dan pemeliharaan sistem di kemudian hari.

Hasil eksperimen disimpan di dalam direktori `results`. Direktori ini memuat berkas *JSON* yang mencatat *response* lengkap untuk setiap *configuration*, termasuk *instruction* yang digunakan, jawaban model, serta metadata yang dihasilkan selama eksekusi. Ringkasan hasil disimpan dalam bentuk *CSV* untuk mempermudah proses analisis, misalnya perbandingan jawaban akhir, tingkat akurasi, jumlah token, atau latensi apabila informasi tersebut disediakan oleh layanan model.

Seluruh kode program ditempatkan dalam direktori `src`. Direktori ini berisi modul yang memuat *specification*, menyusun *instruction*, menjalankan *task* untuk setiap *configuration*, serta mencatat hasil eksekusi ke dalam `results`. Dengan pemisahan antara kode dan data, eksperimen dapat dijalankan kembali dengan pengaturan yang sama atau diperluas dengan *specification* baru tanpa mengubah struktur direktori lainnya.

Dengan struktur direktori ini, setiap *response* yang dihasilkan dapat ditelusuri kembali melalui *persona*, model, dan *task* yang digunakan. Perancangan ini mendukung kebutuhan replikasi eksperimen dan menjadi penghubung antara desain konseptual pada bagian sebelumnya dan analisis hasil pada bab berikutnya.

IV.4.1 Pseudocode Pengunduhan dan Normalisasi Benchmark

Persiapan data *benchmark* dilakukan melalui skrip `gsm8k_download.py` dan `mmlu_redux_download.py`. Kedua skrip ini bertugas mengunduh dataset dari sumber resmi, menyimpannya pada direktori `data`, dan menormalkan struktur data ke format yang konsisten sebelum digunakan oleh pipeline.

Kode IV.4 merangkum alur utama yang diimplementasikan pada kedua skrip tersebut.

Kode IV.4 Pseudocode pengunduhan dan normalisasi benchmark

```
Input :
  - URL_GSM8K_RAW
  - URL_MMLU_REDUX
  - Direktori data D_data

Output:
```

```
  - Berkas GSM8K terstruktur (gsm8k_normalized.json)
  - Berkas MMLU-Redux terstruktur (mmlu_redux_normalized.json)

Prosedur PrepareGSM8K(D_data):
  1. raw_file     <- DownloadFile(URL_GSM8K_RAW)
  2. parsed_items <- ParseRawGSM8K(raw_file)
  3. normalized   <- []
  4. Untuk setiap item dalam parsed_items lakukan:
       record <- {
         "id":            item.id,
         "question_text": item.question,
         "answer":        item.answer
       }
       Tambahkan record ke normalized
  5. out_path <- JoinPath(D_data, "gsm8k_normalized.json")
  6. WriteJSON(out_path, normalized)

Prosedur PrepareMMLURedux(D_data):
  1. raw_file     <- DownloadFile(URL_MMLU_REDUX)
  2. parsed_items <- ParseRawMMLU(raw_file)
  3. normalized   <- []
  4. Untuk setiap item dalam parsed_items lakukan:
       record <- {
         "id":            item.id,
         "subject":       item.subject,
         "question_stem": item.stem,
         "options":       item.options,
         "correct":       item.correct_option
       }
       Tambahkan record ke normalized
  5. out_path <- JoinPath(D_data, "mmlu_redux_normalized.json")
  6. WriteJSON(out_path, normalized)

Prosedur PrepareAllBenchmarks(D_data):
  1. EnsureDirExists(D_data)
  2. PrepareGSM8K(D_data)
  3. PrepareMMLURedux(D_data)
```

Melalui prosedur ini, seluruh *benchmark* yang digunakan pipeline tersedia dalam format terstruktur yang konsisten dan dapat diolah kembali apabila diperlukan.

IV.5 Penanganan Gangguan dan Pemulihan *Execution Flow*

Proses eksekusi melibatkan sejumlah besar kombinasi persona, model, dan *task*, sehingga rentan terhadap gangguan yang bersumber dari layanan model maupun kondisi jaringan. Bagian ini menjelaskan mekanisme yang digunakan untuk mempertahankan keberlanjutan proses dan menjaga agar hasil yang diperoleh tetap konsisten serta dapat ditelusuri kembali.

Penanganan gangguan dilakukan dalam dua bentuk utama.

1. *Transient error handling* Sistem mendeteksi gangguan sementara seperti *timeout*, penolakan layanan, atau pemutusan koneksi. Jika gangguan terjadi, instruksi dijadwalkan ulang menggunakan jeda adaptif, sehingga proses tidak terhenti dan setiap *configuration* tetap menghasilkan keluaran yang dapat dianalisis.
2. *Execution flow recovery* Sistem mencatat status terakhir setiap kali respons berhasil diterima. Jika eksekusi terhenti sebelum seluruh *configuration* selesai diproses, pipeline dapat dilanjutkan dari titik terakhir tanpa mengulang langkah yang telah berhasil. Dengan cara ini, proses panjang tetap dapat diselesaikan tanpa kehilangan progres.

Kedua mekanisme tersebut menjaga stabilitas eksekusi pada skala besar dan memastikan bahwa alur eksperimen tetap dapat dipertanggungjawabkan pada tahap analisis.

IV.5.1 Pseudocode Pemantauan Checkpoint dan Pemulihan Eksekusi

Penanganan gangguan pada pipeline didukung oleh dua skrip utama, yaitu `checkpoint_monitor.py` dan `wait_and_report.py`. Skrip `checkpoint_monitor.py` memantau progres eksekusi dan mencatat daftar *configuration* yang telah menghasilkan keluaran lengkap. Skrip `wait_and_report.py` digunakan untuk memantau progres secara berkala dan melaporkan status eksekusi tanpa perlu membuka setiap berkas log secara manual.

Kode IV.5 merangkum mekanisme pembaruan *checkpoint*.

Kode IV.5 Pseudocode pemantauan checkpoint eksekusi

```
Input :
  - Direktori hasil D_results
  - Daftar konfigurasi C (persona x model x task)
```

```
Output:
  - Berkas checkpoint (checkpoint.json)

Prosedur UpdateCheckpoint(D_results, C):
  1. completed <- Himpunan kosong

  2. Untuk setiap cfg dalam C lakukan:
       run_id   <- BuildRunId(cfg)
       log_path <- JoinPath(D_results, run_id, "summary.json")

       Jika FileExists(log_path):
         Tambahkan run_id ke completed

  3. checkpoint <- {
       "total_configurations": Panjang(C),
       "completed_runs":       completed,
       "remaining_runs":       (C \ completed),
       "last_update":          Now()
     }

  4. WriteJSON(JoinPath(D_results, "checkpoint.json"), checkpoint)
```

Pemantauan progres dilakukan melalui loop berkala sebagaimana diringkas pada Kode IV.6.

Kode IV.6 Pseudocode pemantauan progres dan pelaporan

```
Input :
  - Direktori hasil D_results
  - Interval pemantauan Delta_t

Prosedur WaitAndReport(D_results, Delta_t):
  1. Loop tak hingga:

       checkpoint_path <- JoinPath(D_results, "checkpoint.json")

       Jika FileExists(checkpoint_path):
         cp        <- ReadJSON(checkpoint_path)
         total     <- cp.total_configurations
         completed <- Len(cp.completed_runs)
         remaining <- total - completed
         progress  <- completed / total

         Print("Progress:",
               completed, "/", total,
               "(remaining:", remaining, ",",
```

```
            "progress:", FormatPercent(progress), ")")

      Jika remaining == 0:
        Print("All configurations completed.")
        Keluar dari loop

    Tunggu selama Delta_t detik
```

Dengan mekanisme ini, eksekusi yang terhenti secara tidak terduga dapat dilanjutkan berdasarkan *checkpoint* terakhir, dan progres keseluruhan dapat dipantau secara otomatis tanpa intervensi manual.

IV.6 Implementasi Keluaran Pipeline

Bagian ini menjelaskan bentuk keluaran yang dihasilkan oleh *evaluation pipeline* setelah seluruh tahap pemrosesan selesai dijalankan. Keluaran tersebut berfungsi sebagai artefak utama yang dianalisis pada Bab V. Seluruh hasil disimpan dalam direktori `results` dalam format terstruktur sehingga setiap entri dapat ditelusuri kembali ke persona, model, dan *task* yang digunakan.

IV.6.1 Pseudocode Analisis Hasil dan Rekapitulasi

Tahap pasca-proses hasil eksperimen direalisasikan melalui sejumlah skrip, antara lain `parse_gsm8k_results.py`, `analyze_results.py`, `analyze_answers_deep.py`, `generate_results_csv.py`, `quick_summary.py`, dan `generate_excel.py`. Skrip-skrip tersebut membaca log `JSON` per konfigurasi, menghitung metrik performa (misalnya akurasi dan penggunaan token), lalu menyusun ringkasan dalam format `CSV` dan `spreadsheet`.

Kode IV.7 merangkum alur utama pasca-proses tersebut.

Kode IV.7 Pseudocode pasca-proses hasil eksperimen

```
Input :
  - Direktori hasil D_results
  - Daftar konfigurasi C (persona x model x task)

Output:
  - Berkas CSV ringkasan per konfigurasi
  - Berkas Excel agregasi lintas model dan persona

Prosedur BuildPerConfigSummary(D_results, C):
  1. rows <- daftar kosong
```

```
  2. Untuk setiap cfg dalam C lakukan:
       run_id       <- BuildRunId(cfg)
       summary_path <- JoinPath(D_results, run_id, "summary.json")

       Jika FileExists(summary_path) == False:
         Lanjutkan ke konfigurasi berikutnya

       summary <- ReadJSON(summary_path)
       items   <- summary.items

       // Hitung metrik utama
       total_q    <- Len(items)
       correct_q  <- Count(item.is_correct untuk item dalam items)
       acc        <- correct_q / total_q

       total_tok  <- Sum(item.total_tokens untuk item dalam items jika tersedia)
       avg_tok    <- total_tok / total_q

       avg_lat_ms <- Mean(item.latency_ms untuk item dalam items)

       row <- {
         "model_name":      summary.model_name,
         "persona_id":      summary.persona.id,
         "task_type":       DetectTaskType(run_id),
         "total_questions": total_q,
         "correct":         correct_q,
         "accuracy":        acc,
         "avg_tokens":      avg_tok,
         "avg_latency_ms":  avg_lat_ms
       }

       Tambahkan row ke rows

  3. WriteCSV(JoinPath(D_results, "summary_per_config.csv"), rows)


Prosedur BuildExcelAggregation(D_results):
  1. csv_path <- JoinPath(D_results, "summary_per_config.csv")
  2. table    <- ReadCSV(csv_path)

  3. pivot_model_persona <- Pivot(
       data         = table,
       index        = ["model_name"],
       columns      = ["persona_id"],
```

```
        values       = ["accuracy"],
        agg_function = Mean
      )

  4. pivot_task <- Pivot(
        data         = table,
        index        = ["task_type"],
        columns      = ["model_name"],
        values       = ["accuracy"],
        agg_function = Mean
      )

  5. workbook <- CreateExcelWorkbook()
  6. AddSheet(workbook, "per_config",        table)
  7. AddSheet(workbook, "by_model_persona",  pivot_model_persona)
  8. AddSheet(workbook, "by_task_model",     pivot_task)
  9. SaveExcel(workbook,
               JoinPath(D_results,
                        "summary_aggregated.xlsx"))

Prosedur RunPostProcessing(D_results, C):
  1. BuildPerConfigSummary(D_results, C)
  2. BuildExcelAggregation(D_results)
```

Pseudocode ini menunjukkan bagaimana log mentah yang dihasilkan oleh pipeline dikonversi menjadi ringkasan numerik yang kemudian digunakan dalam analisis pada Bab V.

IV.6.2 Contoh Struktur Log Inferensi

Setiap interaksi antara pipeline dan model dicatat dalam berkas `JSON`. Berkas ini mencakup identitas konfigurasi, isi permintaan, jawaban model, serta telemetri penggunaan token. Kode IV.8 menunjukkan contoh log untuk model yang tidak menyediakan *reasoning trace*.

Kode IV.8 Contoh struktur log inferensi

```
{
  "run": {
    "model_id": "example-model",
    "question_id": "gsm8k_00001",
    "persona": "implicit_male"
  },
  "response": {
    "choices": [
```

```
      {
        "message": {
          "content": "Let's break down the problem..."
        }
      }
    ],
    "usage": {
      "prompt_tokens": 211,
      "completion_tokens": 197,
      "total_tokens": 408
    }
  },
  "meta": {
    "latency_ms": 842,
    "timestamp": "2025-01-18T12:44:10Z"
  }
}
```

IV.6.3 Contoh Struktur Log dengan Reasoning Trace

Beberapa model menyediakan bagian penalaran (*reasoning trace*) selain jawaban akhir. Bagian ini disimpan sebagai elemen terpisah di dalam log. Kode IV.9 menampilkan struktur log lengkap dari model yang menyediakan informasi tersebut.

Kode IV.9 Contoh struktur log dengan reasoning trace

```
{
  "run": {
    "model_id": "example-model-reason",
    "question_id": "gsm8k_00003",
    "persona": "explicit_genz_female"
  },
  "response": {
    "choices": [
      {
        "message": {
          "content": "Final answer: 70000",
          "reasoning": "First compute the purchase cost..."
        }
      }
    ],
    "usage": {
      "completion_tokens": 867,
      "reasoning_tokens": 485,
      "total_tokens": 1352
```

```
    }
  },
  "meta": {
    "latency_ms": 2134,
    "timestamp": "2025-01-18T12:52:41Z"
  }
}
```

Log seperti ini memungkinkan analisis lebih dalam mengenai gaya penalaran serta perubahan struktur argumen yang mungkin dipengaruhi oleh persona.

IV.6.4 Ringkasan Hasil Eksperimen

Pipeline menghasilkan ringkasan performa dalam bentuk tabel yang menggabungkan metrik akurasi dan penggunaan token untuk setiap pasangan persona–model. Berkas ringkasan disimpan dalam format `CSV` untuk memudahkan analisis lanjutan. Tabel berikut merupakan contoh representasi ringkasan hasil.

Tabel IV.2 Contoh ringkasan hasil eksperimen GSM8K untuk seluruh model dan persona

Model	Persona	Total Q	Correct	Accuracy (%)	Total Tokens
Bert Nebulon Alpha	man_implicit	610	593	97.21	285250
Bert Nebulon Alpha	woman_implicit	641	627	97.26	335208
Grok 4.1 Fast	man_implicit	1315	1242	94.45	1325229
Grok 4.1 Fast	woman_implicit	1316	1254	95.36	1422736
Nvidia Nemotron 12B v2 VL	man_implicit	1305	1224	93.79	1156049
Nvidia Nemotron 12B v2 VL	woman_implicit	1306	1230	94.18	1184521

BAB V

RENCANA SELANJUTNYA

V.1 Rencana Implementasi dan Estimasi Biaya

Rencana implementasi pada tahap berikutnya adalah menjalankan kembali *evaluation pipeline* yang telah dijelaskan pada Bab IV dengan cakupan penuh, yang meliputi sembilan model bahasa, dua *benchmark* penalaran (GSM8K dan MMLU-Redux), serta lima belas *user persona* (implisit, eksplisit, dan netral). Bagian ini merumuskan langkah implementasi teknis, asumsi kebutuhan token, serta estimasi biaya penggunaan API berdasarkan harga resmi masing-masing model pada platform OpenRouter

Estimasi dilakukan menggunakan kurs konstan 1 USD = Rp16.700.

V.1.1 Rencana Implementasi Eksperimen

Pelaksanaan eksperimen direncanakan mengikuti enam langkah utama berikut.

1. Persiapan aset data.
   Sistem memuat berkas definisi lima belas persona, korpus GSM8K (split *test*), MMLU-Redux (20 subjek), kredensial API, serta konfigurasi model. Struktur direktori dan modul pemrosesan mengikuti rancangan pada Subbab IV.4.
2. Inisialisasi dan *warm-up* persona.
   Setiap model menerima satu pesan awal untuk menanamkan konteks persona sebelum mengerjakan soal pertama. Tahap ini juga berfungsi sebagai *sanity check* untuk memastikan bahwa model mengikuti identitas dan gaya bahasa persona secara konsisten.
3. Eksekusi eksperimen utama.
   Setiap kombinasi model–persona menjalankan seluruh soal GSM8K dan MMLU-Redux menggunakan mekanisme injeksi pesan berbasis peran: persona pada *system message* dan soal pada *user message*. Setiap respons

diharuskan menyertakan penalaran langkah demi langkah.

4. Pencatatan log granular.
   Seluruh respons disimpan sebagai berkas JSON yang memuat isi *prompt*, jawaban mentah, *token usage*, serta *latency*. Format ini memastikan bahwa setiap respons dapat ditelusuri kembali ke konfigurasi yang digunakan.
5. Agregasi dan validasi hasil.
   Log yang terkumpul diubah menjadi berkas CSV agregat yang berisi akurasi, rata-rata latensi, serta total konsumsi token. Validasi tambahan dilakukan melalui pemeriksaan pola jawaban dan konsistensi jumlah entri.
6. Penanganan kegagalan.
   Kegagalan akibat *timeout* atau batas *rate limit* ditangani menggunakan mekanisme *retry* dengan *exponential backoff*, sebagaimana dijelaskan pada Bab IV. Dengan demikian, kegagalan sebagian tidak menghentikan keseluruhan eksperimen.

V.1.2 Himpunan Model dan Skenario Eksekusi

Eksperimen ini menggunakan sembilan model dengan rincian sebagai berikut.

1. Enam model berbayar (via OpenRouter):
   (a) openai/gpt-5-mini
   (b) qwen/qwen3-vl-30b-a3b-instruct
   (c) google/gemini-2.5-flash
   (d) deepseek/deepseek-v3.2
   (e) nvidia/llama-3.3-nemotron-super-49b-v1.5
   (f) google/gemma-3n-e4b-it
2. Tiga model yang pada saat perancangan tersedia sebagai *free-tier*:
   (a) xai/grok-4.1-fast
   (b) nvidia/nemotron-nano-12b-v2-vl
   (c) openrouter/bert-nebulon-alpha

Seluruh sembilan model dijalankan pada konfigurasi penuh: dua *benchmark* dan lima belas persona. Namun, estimasi biaya hanya dihitung untuk enam model berbayar.

V.1.3 Asumsi Jumlah Soal dan Kebutuhan Token

Kebutuhan token dihitung berdasarkan dua sumber utama: GSM8K (1319 soal) dan MMLU-Redux (2000 soal). Pada kedua *benchmark*, model diarahkan untuk memberikan penalaran lengkap sebelum jawaban akhir, sehingga konsumsi token

per soal diharapkan berada pada kisaran yang relatif tinggi.

1. GSM8K.
   Total token per persona per model diestimasikan sebagai:

$$T_{\text{GSM8K}} \approx 1319 \times 1200 = 1{,}582{,}800 \text{ token}.$$

2. MMLU-Redux.
   Total token per persona per model diestimasikan sebagai:

$$T_{\text{MMLU}} \approx 2000 \times 1200 = 2{,}400{,}000 \text{ token}.$$

Total token inti per persona diperoleh dari penjumlahan keduanya:

$$T_{\text{base, persona}} = 1{,}582{,}800 + 2{,}400{,}000 = 3{,}982{,}800.$$

Untuk mengakomodasi *warm-up* dan *retry*, digunakan faktor overhead 20%:

$$T_{\text{persona}} \approx 1.2 \times 3{,}982{,}800 = 4{,}779{,}360.$$

Sehingga total token per model untuk 15 persona adalah:

$$T_{\text{model}} \approx 15 \times 4{,}779{,}360 = 71{,}690{,}400 \approx 71{,}7 \times 10^6.$$

Komposisi token diasumsikan:

$$T_{\text{in}} = 0.4T_{\text{model}}, \qquad T_{\text{out}} = 0.6T_{\text{model}}.$$

V.1.4 Estimasi Biaya per Model

Harga token per model mengacu pada dokumentasi OpenRouter(*OpenAI GPT-5 Mini Pricing*; Team 2025; *Google Gemini 2.5 Flash Pricing*; *DeepSeek V3.2 Pricing*; *NVIDIA Llama 3.3 Nemotron Super 49B V1.5 Pricing*; *Google Gemma 3n 4B Pricing*). Biaya untuk model ke-m dihitung dengan rumus:

$$\text{cost}_m = p_{\text{in},m} \times \frac{T_{\text{in}}}{10^6} + p_{\text{out},m} \times \frac{T_{\text{out}}}{10^6},$$

dengan $p_{\text{in},m}$ dan $p_{\text{out},m}$ adalah harga per satu juta token untuk *input* dan *output*.

Estimasi berikut menggunakan kurs Rp 16.700 per USD dan total token $T_{\text{model}} \approx 71{,}7 \times 10^6$.

Tabel V.1 Estimasi biaya enam model berbayar untuk konfigurasi penuh 15 persona

Model	**Total Token T_{model}**	**Biaya (USD)**	**Biaya (Rp)**
openai/gpt-5-mini	$\approx 71{,}7 \times 10^6$	93.20	$\approx 1{,}556{,}000$
qwen/qwen3-vl-30b-a3b-instruct	$\approx 71{,}7 \times 10^6$	30.11	$\approx 503{,}000$
google/gemini-2.5-flash	$\approx 71{,}7 \times 10^6$	116.14	$\approx 1{,}939{,}000$
deepseek/deepseek-v3.2	$\approx 71{,}7 \times 10^6$	24.95	$\approx 417{,}000$
nvidia/llama-3.3-nemotron-super-49b-v1.5	$\approx 71{,}7 \times 10^6$	20.07	$\approx 335{,}000$
google/gemma-3n-e4b-it	$\approx 71{,}7 \times 10^6$	2.29	$\approx 38{,}000$
Total enam model berbayar	–	**286.76**	$\approx$ **4,788,000**

Tiga model lain yang tersedia sebagai *free-tier* (grok-4.1-fast, nemotron-nano-12b-v2-vl, dan bert-nebulon-alpha) diperkirakan mengonsumsi token serupa tetapi tidak menimbulkan biaya finansial langsung. Status *free-tier* tersebut tetap harus diverifikasi kembali sebelum eksperimen akhir dijalankan.

Dengan demikian, estimasi total biaya finansial untuk menjalankan seluruh eksperimen multi-model, multi-persona, dan dua *benchmark* penalaran adalah sekitar **286.76 USD**, atau kurang lebih **4,8 juta rupiah**. Angka ini bersifat konservatif karena telah memasukkan biaya *warm-up*, *retry*, dan variasi panjang jawaban, sehingga realisasi biaya dapat lebih rendah apabila konsumsi token aktual per soal ternyata lebih kecil atau lebih besar dari asumsi yang digunakan dalam perhitungan ini.

V.1.5 Rencana Pengerjaan dan Pengembangan

Pelaksanaan Tugas Akhir ini dirancang dalam beberapa tahapan yang saling berurutan, mencakup penyusunan kerangka eksperimen, pengembangan *evaluation pipeline*, pelaksanaan evaluasi pada berbagai model dan persona, hingga analisis hasil. Mengingat cakupan eksperimen yang luas (*multi model* × *multi persona*), durasi pengerjaan diperpanjang untuk memastikan seluruh proses dapat berjalan stabil dan menghasilkan keluaran yang dapat dianalisis secara komprehensif. Rincian tahapan pelaksanaan Tugas Akhir ditampilkan pada Tabel V.2 berikut.

Tabel V.2 Rencana tahapan pelaksanaan Tugas Akhir

Kegiatan	**Durasi**	**Output Utama**
Studi literatur lanjutan dan perumusan kerangka eksperimen	4 minggu	Pemutakhiran tinjauan pustaka, definisi metodologi, pemetaan risiko bias persona, serta penyusunan struktur awal *evaluation pipeline*.
Pengembangan dan implementasi *evaluation pipeline* (multi-model)	6 minggu	Modul pemanggilan model, integrasi API, modul persona, modul benchmark, mekanisme *logging*, serta verifikasi format keluaran model.
Pengujian internal, *dry-run*, dan penyempurnaan pipeline	2 minggu	Pipeline stabil dan konsisten, validasi batas token, pengujian kesalahan, serta standarisasi format data untuk analisis.
Pelaksanaan eksperimen penuh (persona × model × benchmark)	4 minggu	Dataset hasil eksperimen yang lengkap: keluaran GSM8K dan MMLU-Redux, statistik token, latensi, tingkat keberhasilan respons, dan log eksekusi.
Replikasi eksperimen dan verifikasi ulang hasil	1–2 minggu	Eksperimen ulang untuk meningkatkan reliabilitas, mengurangi noise, dan memastikan konsistensi antar-konfigurasi.
Analisis hasil dan penyusunan bagian laporan	3 minggu	Visualisasi hasil, analisis kuantitatif dan kualitatif, interpretasi pengaruh persona terhadap performa model, serta penulisan laporan akhir Tugas Akhir.

V.2 Desain Pengujian dan Evaluasi

Desain pengujian disusun untuk memastikan bahwa seluruh hasil eksperimen dapat diverifikasi, divalidasi, dan direplikasi. Pengujian memanfaatkan artefak log granular, telemetry token, dan pemeriksaan konsistensi yang telah ditanamkan

dalam pipeline pada Bab IV.

1. Verifikasi konsistensi eksekusi

   Tahap ini memastikan bahwa setiap model menerima stimulus yang identik pada seluruh soal dan persona sehingga variasi performa dapat dikaitkan secara langsung dengan persona atau arsitektur model.

   (a) Konsistensi konstruksi prompt

   Pemeriksaan memastikan bahwa struktur persona pada system message dan konten soal pada user message identik pada semua eksekusi. Variasi kecil seperti perbedaan tanda baca dapat mengubah penalaran model sehingga verifikasi dilakukan secara programatik melalui log JSON.

   (b) Kesesuaian urutan eksekusi

   Urutan indeks interaksi, nomor soal, dan urutan persona diperiksa untuk memastikan pipeline mengikuti konfigurasi eksperimen.

   (c) Keberhasilan tahap warm-up

   Respons awal model dinilai untuk melihat apakah gaya bahasa persona sudah terserap dengan benar. Kegagalan dianggap anomali dan dieksekusi ulang.

2. Validasi keluaran model

   Validasi memastikan bahwa keluaran model memiliki format yang dapat dievaluasi secara otomatis pada GSM8K dan MMLU-Redux.

   (a) Validasi GSM8K

   Model harus menghasilkan jawaban numerik akhir yang dapat diekstraksi secara deterministik dan menyertakan penalaran langkah demi langkah.

   (b) Validasi MMLU-Redux

   Model harus memilih salah satu opsi A, B, C, atau D, serta memberikan penjelasan penalaran sebelum menentukan jawaban.

   (c) Pemeriksaan konsistensi format respons

   Pemeriksaan mencakup panjang teks, struktur, keterbacaan, dan kesesuaian format untuk menghindari kesalahan parsing.

3. Evaluasi kuantitatif

   Evaluasi kuantitatif dilakukan untuk mengukur dampak persona terhadap performa model.

   (a) Akurasi jawaban

   Akurasi dihitung dengan membandingkan jawaban akhir terhadap ground truth dan diagregasi per model dan per persona.

   (b) Konsumsi token

Analisis mencakup token input, token output, dan token penalaran sebagai indikator beban komputasi serta kecenderungan verbosity.

(c) Latensi eksekusi

Latensi dihitung berdasarkan timestamp pada log JSON untuk menilai stabilitas waktu respons pada eksekusi berskala besar.

V.3 Analisis Risiko dan Mitigasi

Pelaksanaan eksperimen pada lingkungan multi-model dan multi-persona menimbulkan sejumlah risiko yang perlu dikelola untuk menjaga integrist hasil Tugas Akhir. Risiko terutama berkaitan dengan reliabilitas API, stabilitas keluaran model, dan konsistensi penyimpanan log.

1. Risiko kegagalan pemanggilan API

   Risiko meliputi timeout, gangguan koneksi, dan rate limit yang dapat menyebabkan hilangnya data atau ketidaksinkronan indeks eksekusi.

   (a) Penggunaan mekanisme retry adaptif berbasis exponential backoff
   (b) Pencatatan seluruh galat dalam log terpisah sehingga dapat dilakukan re-eksekusi selektif
   (c) Penurunan tingkat konkurensi secara otomatis ketika laju galat meningkat untuk menjaga stabilitas

2. Risiko lonjakan konsumsi token

   Respons model dapat menjadi terlalu panjang, terutama ketika diminta memberikan penalaran eksplisit, sehingga meningkatkan biaya dan durasi eksperimen.

   (a) Penetapan batas maximum completion length
   (b) Pemantauan berkala terhadap rata-rata konsumsi token
   (c) Penyesuaian minimal pada instruksi persona yang memicu keluaran berlebihan

3. Risiko penyimpanan dan konsistensi log

   Volume log yang besar meningkatkan risiko korupsi berkas dan ketidaksesuaian antara indeks model, persona, dan soal.

   (a) Penyimpanan respons dalam format JSON dengan skema tetap
   (b) Pemeriksaan silang jumlah entri, indeks soal, dan struktur pipeline selama agregasi
   (c) Penerapan checkpointing untuk mencegah kehilangan data jika eksekusi terhenti mendadak

DAFTAR PUSTAKA

Bender, Emily M, Timnit Gebru, Angelina McMillan-Major, dan Margaret Mitchell. 2021. "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" Dalam *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency.* https://doi.org/10.1145/3442188.3445922.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, dan Christian Jauvin. 2003. "A Neural Probabilistic Language Model". Dalam *Journal of Machine Learning Research,* 3:1137–1155.

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, dkk. 2021. "On the Opportunities and Risks of Foundation Models". *arXiv preprint arXiv:2108.07258,* https://arxiv.org/abs/2108.07258.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, dkk. 2020. "Language Models are Few-Shot Learners". *Advances in Neural Information Processing Systems.*

Cobbe, Karl, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, dkk. 2021. "Training Verifiers to Solve Math Word Problems". *arXiv preprint arXiv:2110.14168,* https://arxiv.org/abs/2110.14168.

DeepSeek V3.2 Pricing. https://openrouter.ai/deepseek/deepseek-v3.2. Diakses 2025.

Edinburgh Dataset Analytics Working Group. 2024. *MMLU-Redux 2.0 Dataset.* https://huggingface.co/datasets/edinburgh-dawg/mmlu-redux-2.0. Versi kurasi ulang MMLU dengan 57 subjek dan 100 butir soal per subjek.

Gema, Aryo Pradipta, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, dkk. 2024. “Are We Done with MMLU?” *arXiv preprint arXiv:2406.04127,* https://arxiv.org/abs/2406.04127.

Goodfellow, Ian, Yoshua Bengio, dan Aaron Courville. 2016. *Deep Learning.* MIT Press. https://www.deeplearningbook.org.

Google Gemini 2.5 Flash Pricing. https://openrouter.ai/google/gemini-2.5-flash. Diakses 2025.

Google Gemma 3n 4B Pricing. https://openrouter.ai/google/gemma-3n-e4b-it. Diakses 2025.

Gupta, Shashank, Vaishnavi Shrivastava, Ameet Deshpande, Ashwin Kalyan, Peter Clark, Ashish Sabharwal, dan Tushar Khot. 2024. “Bias Runs Deep: Implicit Reasoning Biases in Persona-Assigned Language Models”. Dalam *Proceedings of the Twelfth International Conference on Learning Representations.* https://openreview.net/forum?id=kGteeZ18Ir.

Hendrycks, Dan, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, dan Jacob Steinhardt. 2021. “Measuring Massive Multitask Language Understanding”. *International Conference on Learning Representations,* https://arxiv.org/abs/2009.03300.

Jurafsky, Dan, dan James H. Martin. 2023. *Speech and Language Processing.* 3rd. Draft of January 7, 2023. https://web.stanford.edu/~jurafsky/slp3/.

Liang, P., R. Bommasani, dkk. 2023. “Holistic Evaluation of Language Models”. *arXiv preprint arXiv:2211.09110,* https://arxiv.org/abs/2211.09110.

Liu, Nelson F, dkk. 2024. “Lost in the Middle: How Language Models Use Long Contexts”. *Transactions of the Association for Computational Linguistics,* https://arxiv.org/abs/2307.03172.

Liu, Pengfei, dkk. 2023. “Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing”. *ACM Computing Surveys,* https://doi.org/10.1145/3560815. https://dl.acm.org/doi/10.1145/3560815.

Naous, Tarek, Baptiste Roziere, dkk. 2025. “Training and Evaluating User Language Models”. *arXiv preprint arXiv:2510.06552,* https://arxiv.org/abs/2510.06552.

NVIDIA Llama 3.3 Nemotron Super 49B V1.5 Pricing. https://openrouter.ai/nvidia/llama-3.3-nemotron-super-49b-v1.5. Diakses 2025.

OpenAI GPT-5 Mini Pricing. https://openrouter.ai/openai/gpt-5-mini. Diakses 2025.

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, dkk. 2022. "Training Language Models to Follow Instructions with Human Feedback". *arXiv preprint arXiv:2203.02155,* https://arxiv.org/abs/2203.02155.

Ranzato, Marc'Aurelio, Sumit Chopra, Michael Auli, dan Wojciech Zaremba. 2016. "Sequence Level Training with Recurrent Neural Networks". Dalam *International Conference on Learning Representations*. https://arxiv.org/abs/1511.06732.

Sap, Maarten, Hannah Rashkin, Derek Chen, Ronan Le Bras, dan Yejin Choi. 2019. "SocialIQA: Commonsense Reasoning about Social Interactions". Dalam *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. Hong Kong, China. https://arxiv.org/abs/1904.09728.

Schick, Timo, dan Hinrich Schütze. 2021. "Exploiting Cloze Questions for Few-Shot Text Classification and Natural Language Inference". Dalam *EACL*. https://aclanthology.org/2021.eacl-main.24/.

Team, Qwen. 2025. *Qwen3 VL 30B A3B Instruct.* https://openrouter.ai/models/qwen/qwen3-vl-30b-a3b-instruct. 262,144 context window. Pricing: $0.15/M input tokens, $0.60/M output tokens. Created October 6, 2025.

Tseng, Yu-Min, Yu-Chao Huang, Teng-Yun Hsiao, Wei-Lin Chen, Chao-Wei Huang, Yu Meng, dan Yun-Nung Chen. 2024. "Two Tales of Persona in LLMs: A Survey of Role-Playing and Personalization". Dalam *Findings of the Association for Computational Linguistics: EMNLP 2024,* 16612–16631. https://aclanthology.org/2024.findings-emnlp.969.

Turpin, Miles, dkk. 2023. "Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Reasoning". *arXiv preprint arXiv:2305.04388,* https://arxiv.org/abs/2305.04388.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, dan Illia Polosukhin. 2017. "Attention Is All You Need". Dalam *Advances in Neural Information Processing Systems*.

Wei, Jason, dkk. 2022. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. *NeurIPS,* https://arxiv.org/abs/2201.11903.

Weidinger, Laura, John Mellor, Maribeth Rauh, Christopher Griffin, Iason Gabriel, Jonathan Uesato, Po-Sen Huang, Zachary Kenton, Tom B. Brown, dkk. 2021. “Ethical and Social Risks of Harm from Language Models”. *arXiv preprint arXiv:2112.04359,* https://arxiv.org/abs/2112.04359.

Zhao, Yanhao, Eric Wallace, Shi Feng, Mohit Singh, dan Matt Gardner. 2021. “Calibrate Before Use: Improving Few-Shot Performance of Language Models”. Dalam *Proceedings of the International Conference on Machine Learning,* 12697–12706.

Zhao, Zhengxuan, dkk. 2021. “Calibrate Before Use: Improving Few-Shot Performance of Language Models”. Dalam *ICML*. https://arxiv.org/abs/2102.09690.

Zhou, Luozhi, dkk. 2023. “Large Language Models Are Sensitive to Prompt Framing”. *arXiv preprint arXiv:2310.05400,* https://arxiv.org/abs/2310.05400.