



UNIVERSITAS INDONESIA

**PENALARAN LOGIKA BERBASIS FRAMEWORK
TRANSLATION-DECOMPOSITION-SEARCH-RESOLVE PADA
OPEN-WEIGHT SMALL LANGUAGE MODEL DENGAN
DATASET BERBAHASA INDONESIA**

SKRIPSI

**MIKHAEL DEO BARLI
1906350572**

**FAKULTAS ILMU KOMPUTER
PROGRAM STUDI ILMU KOMPUTER
DEPOK
DESEMBER 2025**



UNIVERSITAS INDONESIA

**PENALARAN LOGIKA BERBASIS FRAMEWORK
TRANSLATION-DECOMPOSITION-SEARCH-RESOLVE PADA
OPEN-WEIGHT SMALL LANGUAGE MODEL DENGAN
DATASET BERBAHASA INDONESIA**

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar
Sarjana Komputer

**MIKHAEL DEO BARLI
1906350572**

**FAKULTAS ILMU KOMPUTER
PROGRAM STUDI ILMU KOMPUTER
DEPOK
DESEMBER 2025**

HALAMAN PERNYATAAN ORISINALITAS

**Skripsi ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.**

**Nama : Mikhael Deo Barli
NPM : 1906350572
Tanda Tangan :**

Tanggal : Tanggal Bulan Tahun

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama : Mikhael Deo Barli

NPM : 1906350572

Program Studi : Ilmu Komputer

Judul Skripsi : Penalaran Logika Berbasis Framework Translation-Decomposition-Search-Resolve pada Open-Weight Small Language Model dengan Dataset Berbahasa Indonesia

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana pada Program Studi Ilmu Komputer, Fakultas Ilmu Komputer, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing 1 : Ari Saptawijaya, S.Kom., M.Sc., Ph.D ()

Penguji 1 : Penguji Pertama Anda ()

Penguji 2 : Penguji Kedua Anda ()

Ditetapkan di : Depok

Tanggal : Tanggal Bulan Tahun

KATA PENGANTAR

Puji dan syukur Penulis panjatkan kepada Tuhan Yang Maha Esa atas segala berkat, rahmat, dan karunia-Nya, sehingga Penulis dapat menyelesaikan skripsi dengan judul PENERAPAN LOGIKA BERBASIS FRAMEWORK TRANSLATION-DECOMPOSITION-SEARCH-RESOLVE PADA OPEN-WEIGHT SMALL LANGUAGE MODEL DENGAN DATASET BERBAHASA INDONESIA ini dengan baik. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana pada Program Studi Ilmu Komputer, Fakultas Ilmu Komputer, Universitas Indonesia. Oleh karena itu, pada kesempatan ini Penulis ingin menyampaikan rasa terima kasih dan penghargaan yang setinggi-tingginya kepada:

1. Pak Ari, selaku dosen pembimbing satu yang telah meluangkan waktu, tenaga, dan pikiran untuk memberikan arahan, diskusi mendalam mengenai *Neuro-Symbolic*, serta motivasi kepada Penulis selama proses penelitian ini.
2. Seluruh Dosen Fakultas Ilmu Komputer yang telah memberikan bekal ilmu pengetahuan yang sangat bermanfaat bagi Penulis selama masa perkuliahan.
3. Orang tua tercinta, yang senantiasa memberikan doa, kasih sayang, dukungan moral, dan material yang tak terhingga.
4. Teman-teman seperjuangan yang telah menjadi teman diskusi dan berbagi cerita selama masa perkuliahan.
5. Pihak pengembang *open-source* (komunitas AI Singapore, Qwen Team, dan GoTo Group) yang telah menyediakan model SEA-LION, Qwen, dan Sahabat-AI secara terbuka, yang memungkinkan penelitian ini terlaksana dengan sumber daya terbatas.
6. Platform *cloud computing* yang menyediakan sumber daya komputasi untuk menggunakan model pembelajaran mesin, yaitu Runpod.
7. Semua referensi dan literatur yang telah membantu dalam penelitian dan penulisan skripsi ini.

Tugas akhir ini disusun dengan bantuan teknologi kecerdasan artifisial dengan platform *GeminiAI* untuk membantu dalam pemilihan kata dan perbaikan tata bahasa, terutama pada bagian landasan teori. Penulis menyadari bahwa laporan Skripsi ini masih jauh dari sempurna, mengingat keterbatasan pengetahuan dan pengalaman Penulis dalam bidang

Natural Language Processing (NLP) yang terus berkembang. Oleh karena itu, apabila terdapat kesalahan atau kekurangan dalam laporan ini, Penulis memohon agar kritik dan saran yang membangun bisa disampaikan langsung melalui *e-mail* mikhael.deo@ui.ac.id.

Akhir kata, Penulis berharap semoga skripsi ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan, khususnya dalam bidang penalaran mesin pada Bahasa Indonesia.

Depok, Tanggal Bulan Tahun

Mikhael Deo Barli

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama : Mikhael Deo Barli
NPM : 1906350572
Program Studi : Ilmu Komputer
Jenis Karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty Free Right)** atas karya ilmiah saya yang berjudul:

Penalaran Logika Berbasis Framework Translation-Decomposition-Search-Resolve pada Open-Weight Small Language Model dengan Dataset Berbahasa Indonesia

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : Tanggal Bulan Tahun
Yang menyatakan

(Mikhael Deo Barli)

ABSTRAK

Nama : Mikhael Deo Barli
Program Studi : Ilmu Komputer
Judul : Penalaran Logika Berbasis Framework Translation-Decomposition-Search-Resolve pada Open-Weight Small Language Model dengan Dataset Berbahasa Indonesia
Pembimbing : Ari Saptawijaya, S.Kom., M.Sc., Ph.D

Kecerdasan buatan (AI) sedang mengalami perubahan yang besar. Industri yang dulunya fokus pada membangun model bahasa yang semakin besar (*Large Language Models - LLM*), kini bergeser ke arah yang lebih efisien dan terjangkau melalui *Small Language Models (SLM)*. Model-model dengan parameter di bawah 10 miliar ini membuka peluang besar, yaitu teknologi AI canggih dapat berjalan bahkan di perangkat dengan keterbatasan sumber daya, sesuatu yang sangat penting bagi negara berkembang seperti Indonesia. Namun, pengurangan ukuran ini membawa tantangan: apakah kemampuan penalaran logikanya juga berkurang, khususnya dalam menangani tugas-tugas logika yang kompleks di mana model kecil mudah menghasilkan hasil yang tidak akurat?

Penelitian ini menjawab tantangan tersebut dengan menguji bagaimana pendekatan *Neuro-Symbolic*, sebuah metode *hybrid* yang menggabungkan fleksibilitas model neural dengan ketepatan logika simbolik, dapat meningkatkan kemampuan penalaran SLM pada data berbahasa Indonesia. Penelitian menerapkan sebuah pipeline lengkap: menerjemahkan pertanyaan ke logika formal (*Translation to First Order Logic*), mengubahnya menjadi bentuk normal konjunktif (*Decomposition to Conjunctive Normal Form*), mencari klausa lengkap (*Search for Complementing Clause*), dan terakhir menggunakan resolusi dengan bukti kontradiksi (*Resolution with Proof By Contradiction*). Eksperimen melibatkan tiga model *open-weight* berukuran kecil (7B-9B parameter, dikuantifikasi 4-bit): Qwen2.5-7B (global), SEA-LION-v3-8B (regional), dan SahabatAI-v1-8B (nasional). Menggunakan dataset logika ProntoQA yang adaptasi ke Bahasa Indonesia, penelitian ini membandingkan seberapa baik masing-masing model menangani tugas penalaran dengan dan tanpa *framework Neuro-Symbolic*.

Hasil penelitian mengungkapkan temuan yang menarik: model global Qwen2.5, yang sebelumnya unggul dalam penalaran implisit (*Naive Prompting*) dengan akurasi 81.00%, justru mengalami penurunan drastis menjadi 14.00% ketika diminta mematuhi aturan sintaksis ketat dalam *pipeline* simbolik *parsability error*. Di sisi lain, model regional SEA-LION v3 menunjukkan performa terbaik dibandingkan dengan dua model lain dalam penelitian ini dengan akurasi 81.60% dalam *framework* ini. Ini menunjukkan bahwa keselarasan linguistik regional memainkan peran penting dalam mengatasi hambatan penerjemahan semantik. Berdasarkan temuan ini, penelitian merekomendasikan bahwa untuk penalaran logika pada LLM dengan sumber daya terbatas, dapat menggunakan SLM regional dan menggunakan *framework Neuro-Symbolic* sebagai solusi yang lebih andal dibandingkan hanya mengandalkan intuisi probabilistik model semata.

Kata kunci:

Open-weight SLM, Penalaran logika, ProntoQA bahasa Indonesia, framework Resolusi

ABSTRACT

Name	:	Mikhael Deo Barli
Study Program	:	Computer Science
Title	:	Logical Reasoning Framework Translation-Decomposition-Search-Resolve on Open-Weight Small Language Model with Indonesian Language Dataset
Counselor	:	Ari Saptawijaya, S.Kom., M.Sc., Ph.D

Artificial Intelligence (AI) is undergoing a significant transformation. The industry, which previously focused on building increasingly larger language models (*Large Language Models* - LLMs), is now shifting toward greater efficiency and affordability through *Small Language Models* (SLMs). Models with fewer than 10 billion parameters open up vast opportunities, enabling advanced AI technology to run even on resource-constrained devices, a crucial factor for developing nations like Indonesia. However, this reduction in size brings challenges: is logical reasoning capability compromised, particularly when handling complex logical tasks where small models are prone to producing inaccurate results?

This study addresses these challenges by examining how a *Neuro-Symbolic* approach, a hybrid method combining the flexibility of neural models with the precision of symbolic logic can enhance SLM reasoning capabilities on Indonesian language data. The research implements a complete pipeline: translating questions to logic (*Translation to First Order Logic*), converting them to conjunctive normal form (*Decomposition to Conjunctive Normal Form*), searching for complementing clauses (*Search for Complementing Clause*), and finally utilizing resolution with proof by contradiction (*Resolution with Proof By Contradiction*). Experiments involved three small *open-weight* models (7B-9B parameters, 4-bit quantized): Qwen2.5-7B (global), SEA-LION-v3-8B (regional), and SahabatAI-v1-8B (national). Using the ProntoQA logic dataset adapted into Indonesian, this research compares how well each model handles reasoning tasks with and without the *Neuro-Symbolic framework*.

The results reveal intriguing findings: the global model, Qwen2.5, which previously excelled in implicit reasoning (*Naive Prompting*) with an accuracy of 81.00%, experienced a drastic drop to 14.00% when required to adhere to strict syntactic rules within the symbolic pipeline (due to *parsability error*). Conversely, the regional model, SEA-LION v3, demonstrated the best performance compared to the other two models, achieving 81.60% accuracy within this framework. This indicates that regional linguistic alignment plays a critical role in overcoming semantic translation barriers. Based on these findings, the research recommends that for logical reasoning on resource-constrained LLMs, utilizing regional SLMs combined with a *Neuro-Symbolic framework* offers a more reliable solution than relying solely on the model's probabilistic intuition.

Key words:

Open-weight SLM, logical reasoning, ProntoQA, Resolution framework

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN ORISINALITAS	ii
LEMBAR PENGESAHAN	iii
KATA PENGANTAR	iv
LEMBAR PERSETUJUAN KARYA ILMIAH	vi
ABSTRAK	vii
DAFTAR ISI	xi
DAFTAR GAMBAR	xiii
DAFTAR TABEL	xiv
DAFTAR KODE PROGRAM	xiv
DAFTAR LAMPIRAN	xv
1. PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan Penelitian	4
1.4 Batasan Penelitian	4
1.5 Manfaat Penelitian	4
1.6 Posisi Penelitian	5
1.7 Sistematika Penulisan	7
2. LANDASAN TEORI	8
2.1 <i>First-Order Logic</i> (FOL)	8
2.1.1 Sintaks: Alfabet dan Aturan Pembentukan	8
2.1.2 Semantik: Interpretasi dan Kebenaran	8
2.2 Konsekuensi Logis (<i>Entailment</i>)	9
2.3 Resolusi dan Unifikasi	9
2.3.1 Pembuktian Kontradiksi (<i>Proof by Contradiction</i>)	9
2.3.2 Konversi ke <i>Conjunctive Normal Form</i> (CNF)	10
2.3.3 Aturan Resolusi dan Unifikasi	10
2.4 <i>Language Models</i> (LM)	11
2.4.1 Arsitektur Transformer	11
2.4.2 Pre-training dan Fine-tuning	11
2.4.3 Inferensi dan Kuantisasi	11
2.4.4 Definisi dan Karakteristik Small Language Models (SLM)	12
2.5 Dataset ProntoQA	13
2.5.1 Generasi Data Sintetis	13
2.5.2 Mengisolasi Kemampuan Penalaran	13
3. FRAMEWORK TRANSLATION-DECOMPOSITION-SEARCH-RESOLVE	14
3.1 Konfigurasi Model	14
3.1.1 Pemilihan Model	14
3.1.2 Parameterisasi Inferensi	15
3.2 Kerangka Kerja Aristotle	16
3.2.1 Translasi Logika (<i>Logical Translation</i>)	17

3.2.2	Dekomposisi (<i>Decomposer</i>)	17
3.2.3	Penyelesaian Pencarian (<i>Search Router</i>)	18
3.2.4	Resolusi (<i>Logical Resolver</i>)	18
3.3	Teknik Evaluasi dan Analisis Data	19
3.3.1	Parsing Bertingkat (Multi-stage Parsing)	19
3.3.2	Matriks Keputusan Evaluasi (Truth Table)	23
4.	EKSPERIMEN, HASIL, DAN ANALISIS	25
4.1	Desain Eksperimen	25
4.2	Instrumen Data dan Adaptasi Linguistik	26
4.3	<i>Prompt Refining</i>	26
4.3.1	<i>Translation to First Order Logic</i>	27
4.3.2	<i>Decomposition to Conjunctive Normal Form</i>	38
4.3.3	<i>Search Resolve</i>	45
4.4	Hasil Eksperimen dan Analisis	67
4.4.1	Naive Prompting	67
4.4.2	Aristotle	68
5.	PENUTUP	70
5.1	Kesimpulan	70
5.2	Saran	72
BIBLIOGRAFI		73

DAFTAR GAMBAR

Gambar 3.1. Kerangka kerja Aristotle untuk inferensi logika menggunakan SLM dan metode <i>neuro-symbolic</i> (Xu et al. (2025))	16
--	----

DAFTAR TABEL

Tabel 1.1.	Perbandingan Penelitian Terkait Penalaran Logis dengan SLM	6
Tabel 3.1.	Matriks Keputusan untuk Penentuan Label Jawaban (Kesimpulan) . . .	24
Tabel 4.1.	Hasil Eksperimen dengan <i>Naive Prompting</i>	67
Tabel 4.2.	Hasil Eksperimen dengan <i>Aristotle Framework</i>	68

DAFTAR KODE PROGRAM

Kode 3.1. Konfigurasi parameter deterministik untuk setiap model yang digunakan	15
Kode 3.2. Mekanisme injeksi data poin ke dalam template prompt Aristotle	18
Kode 3.3. Regex untuk ekstraksi blok Translasi	19
Kode 3.4. Regex untuk ekstraksi blok Dekomposisi	21
Kode 3.5. Regex untuk memvalidasi langkah Resolusi	22
Kode 4.1. Template prompt hasil translasi langsung dari bahasa Inggris	27
Kode 4.2. Template prompt setelah dimodifikasi dengan tambahan beberapa aturan dan contoh	30
Kode 4.3. Template prompt dengan penggunaan diksi sesuai kamus EBI pada konteks ini	34
Kode 4.4. Template prompt hasil translasi langsung dari bahasa Inggris	38
Kode 4.5. Template prompt dengan tambahan contoh dan diksi sesuai EBI	41
Kode 4.6. Template prompt hasil translasi langsung dari bahasa Inggris	45
Kode 4.7. Template prompt dengan penghilangan perintah operator AND dan tambahan contoh	53
Kode 4.8. Template prompt dengan tambahan contoh dan mengganti operator NEG dengan $P(x, \text{NilaiKebalikan})$	58

DAFTAR LAMPIRAN

BAB 1

PENDAHULUAN

Bab ini memaparkan latar belakang, permasalahan, tujuan, batasan, manfaat, ringkasan metodologi, serta sistematika penulisan laporan ini. Penelitian ini berfokus pada evaluasi kemampuan penalaran logis oleh *Small Language Models* (SLM) yang bersifat *open-weight* ketika bekerja pada dataset berbahasa Indonesia dengan *framework translation-decomposition-search-resolve*.

1.1 Latar Belakang

Perkembangan *Large Language Model* (LLM) telah mendorong kemajuan signifikan pada berbagai tugas pemrosesan bahasa natural seperti penerjemahan, ringkasan, dan tanya-jawab. Namun, kemampuan LLM untuk melakukan penalaran logis, yaitu melakukan inferensi yang benar dari himpunan premis dan aturan formal, masih menghadapi kendala fundamental. Evaluasi yang ketat menunjukkan bahwa model sering kali berfungsi sebagai mesin probabilistik yang menebak pola statistik alih-alih melakukan deduksi deterministik. Fenomena ini dijelaskan oleh Saparov and He (2023) dalam penelitian mereka yang menunjukkan bahwa LLM sering gagal dalam inferensi multi-langkah karena mereka mengambil jalan pintas heuristik daripada mengikuti rantai logika yang valid.

Selama dekade terakhir, perkembangan *Natural Language Processing* (NLP) didorong oleh (*scaling laws*) yang menyatakan bahwa peningkatan jumlah parameter, data, dan komputasi akan secara linear meningkatkan performa model. Era ini melahirkan model-model raksasa atau *Large Language Models* (LLM) dengan ratusan miliar parameter, seperti GPT-4 dan Llama-3-405B. Model-model ini menunjukkan kemampuan *emergent* yang luar biasa, namun keberhasilan ini datang dengan biaya infrastruktur komputasi yang masif dan konsumsi energi yang besar Wang et al. (2025).

Dalam konteks negara berkembang seperti Indonesia, ketergantungan pada LLM raksasa menjadi hambatan signifikan. Kebutuhan akan kedaulatan data dan privasi mendorong permintaan untuk menjalankan model secara lokal (*on-premise*) atau di perangkat tepi (*edge devices*). Untuk menghadapi tantangan ini, riset pada umumnya mulai beralih fokus pada *Small Language Models* (SLM), yang umumnya memiliki parameter di bawah 10-14

miliar Subramanian et al. (2025).

Namun, tantangan mendasar tetap membayangi utilitas SLM: kemampuan penalaran logis (*logical reasoning*). Model bahasa pada dasarnya adalah mesin prediksi token probabilistik. Kelemahan ini dikenal sebagai masalah "halusinasi", sering kali lebih parah pada model kecil karena keterbatasan memori asosiatif Saparov and He (2023). Oleh karena itu, diperlukan pendekatan metodologis untuk memperkuat kapabilitas penalaran SLM tanpa memperbesar ukuran modelnya.

Untuk mengatasi kesenjangan antara kemampuan linguistik dan logika ini, berbagai metode prompting dan arsitektur telah dikembangkan. Berikut adalah sedikit tinjauan terhadap evolusi metode penalaran logis pada SLM yang menjadi landasan penelitian ini:

1. *Naive Prompting (Implicit Reasoning)*: Pendekatan paling dasar di mana model diminta langsung menjawab kesimpulan dari premis yang diberikan (misalnya, Zero-shot). Kelemahannya adalah "Curse of Complexity", di mana akurasi model menurun secara eksponensial seiring bertambahnya langkah logika karena model tidak memiliki memori kerja eksternal untuk menyimpan status inferensi perantara (Saparov and He (2023)).
2. *Chain-of-Thought (CoT)*: Diperkenalkan oleh Wei et al. (2022), CoT mendorong model untuk menghasilkan serangkaian langkah penalaran perantara sebelum memberikan jawaban akhir. Metode ini terbukti meningkatkan performa pada tugas aritmatika dan simbolik secara signifikan dengan mengubah pemetaan Input-Output menjadi Input-Reason1-Reason2-Output. Namun, CoT rentan terhadap propagasi kesalahan, jika satu langkah penalaran salah (halusinasi), seluruh kesimpulan akan salah karena tidak ada mekanisme verifikasi eksternal.
3. *Tree-of-Thoughts (ToT)*: Yao et al. (2023) mengembangkan konsep CoT menjadi struktur pohon, memungkinkan model untuk mengeksplorasi berbagai jalur penalaran, melakukan lookahead, dan backtracking saat menemui jalan buntu. Meskipun lebih kuat, ToT sangat mahal secara komputasi dan masih bergantung pada intuisi probabilistik model itu sendiri untuk mengevaluasi validitas setiap cabang pemikiran.
4. *Neuro-Symbolic Approaches (Logic-LM & SymbCoT)*: Untuk mencapai ketepatan logika yang strict, pendekatan *Neuro-Symbolic* mulai diadopsi. Logic-LM oleh Pan et al. (2023) menggunakan LLM hanya sebagai penerjemah masalah ke dalam kode simbo-

lik (seperti Prolog) , yang kemudian diselesaikan oleh solver deterministik. Xu et al. (2024) kemudian mengusulkan SymbCoT yang mencoba mengintegrasikan verifikasi simbolik langsung ke dalam rantai pemikiran LLM.

5. Aristotle *Framework*: Penelitian ini berfokus pada *Framework* Aristotle oleh Xu et al. (2025), yang menyempurnakan pendekatan *Neuro-Symbolic* dengan arsitektur *Translation-Decompose-Search-Resolve*. Keunggulan utamanya adalah adanya modul Search Router yang memangkas ruang pencarian premis yang tidak relevan, serta penggunaan dua jalur pembuktian untuk meminimalkan halusinasi.

Meskipun metode-metode di atas menunjukkan hasil positif, sebagian besar penelitian dilakukan pada dataset berbahasa Inggris. Studi terhadap kemampuan penalaran SLM pada bahasa lain, termasuk Bahasa Indonesia, masih terbatas. Perbedaan struktur linguistik (seperti ambiguitas subjek dalam Bahasa Indonesia) dan kualitas tokenisasi dapat memengaruhi performa model setelah adaptasi lintas bahasa. Selain itu, penelitian sebelumnya menggunakan proprietary LLM (seperti GPT-4), sehingga perbandingan performa *apple-to-apple* pada model open-weight dengan sumber daya terbatas atau SLM belum banyak dikaji.

Dari gap penelitian yang sudah disebutkan, belum ada kajian mendalam mengenai efektivitas *Framework* Aristotle pada dataset berbahasa Indonesia menggunakan open-weight SLM. Penelitian ini menjadi penting untuk mengevaluasi apakah model seperti Sahabat-AI (yang dilatih khusus untuk Bahasa Indonesia) atau SEA-LION (regional ASEAN) ataupun Qwen (generik) dapat mengatasi tantangan penalaran logis.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah penelitian ini adalah:

1. Bagaimana penalaran logis pada dataset berbahasa Indonesia dilakukan dengan menggunakan *framework translation-decompose-search-resolve*?
2. Bagaimana perbandingan performa *framework translation-decompose-search-resolve* untuk penalaran logis berbahasa Indonesia antar open-weight LLM berparameter rendah atau SLM?
3. Bagaimana perbandingan performa *framework translation-decompose-search-resolve* dibandingkan naive prompting untuk penalaran logis berbahasa Indonesia pada open-

weight LLM berparameter rendah atau SLM?

1.3 Tujuan Penelitian

Berikut adalah tujuan dari penelitian sesuai dengan latar belakang dan rumusan masalah

Tujuan:

1. Mengimplementasikan penalaran logis pada dataset berbahasa Indonesia dilakukan dengan menggunakan *framework translation-decompose-search-resolve*
2. Mengukur perbandingan performa *framework translation-decompose-search-resolve* untuk penalaran logis berbahasa Indonesia antar open-weight LLM berparameter rendah atau SLM
3. Mengukur perbandingan performa *framework translation-decompose-search-resolve* naive prompting untuk penalaran logis berbahasa Indonesia pada open-weight LLM berparameter rendah atau SLM

1.4 Batasan Penelitian

Penelitian ini dibatasi sebagai berikut:

- **Dataset:** Fokus pada dataset diterjemahkan ke Bahasa Indonesia, yaitu ProntoQA saja
- **Model:** Eksperimen menggunakan model open-weight yang dapat dijalankan lokal maupun server, khususnya dengan kuantisasi. Model tersebut antara lain: Qwen2.5-7B-IT-GGUF, SEALIONv3-LLama-8B-IT-GGUF, dan SahabatAIv1-Llama-8B-IT-GGUF
- **Evaluasi:** Metrik utama adalah akurasi jawaban akhir terhadap ground truth.
- **Implementasi:** Source code yang tersedia pada repository Aristotle yang merupakan implementasi *framework translation-decompose-search-resolve* (Xu et al. (2025))

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberikan kontribusi yang bermakna bagi berbagai pihak:

- **Bagi pengembangan ilmu pengetahuan:** Penelitian ini dapat menambah pemahaman tentang kemampuan penalaran logis SLM pada bahasa Indonesia, sebuah aspek yang masih jarang dikaji. Temuan ini dapat menjadi fondasi bagi penelitian lanjutan dalam evaluasi model bahasa pada tugas-tugas penalaran logis kompleks dalam bahasa lokal.

- **Bagi praktisi dan pengembang:** Hasil analisis perbandingan model dan efektivitas *framework* dapat menjadi panduan dalam memilih model open-weight yang tepat dan merancang pipeline pemrosesan bahasa untuk tugas penalaran logis berbahasa Indonesia, terutama dengan sumber daya komputasi terbatas.
- **Bagi komunitas penelitian terbuka:** Dataset ProntoQA yang diterjemahkan ke bahasa Indonesia, skrip eksperimen, serta laporan hasil penelitian akan dibagikan kepada publik. Kontribusi ini memungkinkan peneliti lain untuk mereplikasi, memvalidasi, dan melanjutkan penelitian dalam domain yang sama tanpa perlu melakukan terjemahan dan persiapan data dari awal.

1.6 Posisi Penelitian

Penelitian ini mengisi gap dalam literatur saat ini dengan menerapkan kerangka kerja *Neuro-Symbolic* pada model berparameter rendah (*low-resource*) yang dikuantifikasi, khusus untuk Bahasa Indonesia.

Peneliti (Tahun)	Metode / Frame-work	Model / Bahasa / Dataset	Keterbatasan (Gap)
Saparov & He (2022)	Naive Prompting & CoT	GPT-3 (175B) Inggris ProntoQA (Eng)	Hanya mengevaluasi model proprietary <i>high-resource</i>
Pan et al. (2023)	Logic-LM (Translate-Execute)	GPT-3.5 / GPT-4 Inggris ProofWriter, ProntoQA	Bergantung pada solver eksternal tanpa mekanisme <i>search</i> untuk memangkas premis tidak relevan.
Xu et al. (2025)	Aristotle (Translastion-Decompose-Search-Resolve)	Llama-2 / GPT-4 Inggris LogicNLI, PronoQA, ProofWriter	<i>Framework</i> yang efektif, tetapi belum diuji pada model LLM <i>low-resource</i> dengan dataset berbahasa Indonesia.
Koto et al. (2023)	Evaluasi Benchmark	IndoGPT, XGLM Indonesia IndoMMLU, IndoNLI	Fokus pada evaluasi pengetahuan umum, bukan penalaran logika formal yang membutuhkan multilangkah penyelesaian.
Penelitian (2025)	Ini Aristotle (Translastion-Decompose-Search-Resolve)	Qwen2.5, SEA-LION, Sahabat-AI Indonesia ProntoQA-ID	Menguji efektivitas <i>translastion-decompose-search-resolve</i> pada model <i>open-weight</i> kecil dan terkuantisasi.

Tabel 1.1: Perbandingan Penelitian Terkait Penalaran Logis dengan SLM

Dari tabel Tabel 1.1, terlihat bahwa penelitian-penelitian sebelumnya umumnya menggunakan model proprietary berukuran besar dan fokus pada bahasa Inggris. Penelitian ini menjembatani *gap* tersebut dengan mengadaptasi *framework translation-decompose-search-resolve* pada model *open-weight* berparameter rendah pada dataset berbahasa In-

donesia, sehingga memberikan wawasan baru tentang kemampuan penalaran logis dalam konteks sumber daya terbatas dan bahasa lokal.

1.7 Sistematika Penulisan

Sistematika penulisan laporan adalah sebagai berikut:

- Bab 1 PENDAHULUAN

Menguraikan motivasi penelitian, kesenjangan dalam literatur saat ini mengenai penalaran SLM bahasa Indonesia, rumusan masalah, dan tujuan spesifik validasi *framework translation-decompose-search-resolve*.

- Bab 2 LANDASAN TEORI

Menjelaskan prinsip-prinsip First-Order Logic (FOL) dan aturan inferensi berdasarkan literatur klasik, serta tinjauan mendalam mengenai *Neuro-Symbolic* dan evolusi dari *Naive Prompting* ke *framework translation-decompose-search-resolve*.

- Bab 3 FRAMEWORK TRANSLATION-DECOMPOSITION-SEARCH-RESOLVE

Mendeskripsikan desain eksperimen, proses adaptasi dataset ProntoQA ke Bahasa Indonesia, konfigurasi model, dan implementasi teknis *framework translation-decompose-search-resolve*.

- Bab 4 EKSPERIMEN, HASIL, DAN ANALISIS

Menyajikan data hasil akurasi antara Qwen, SEA-LION, dan Sahabat-AI. Bab ini akan menganalisis permasalahan pada tiap tahap *framework* serta membandingkan hasilnya dengan metode Naive Prompting.

- Bab 5 PENUTUP

Merangkum temuan utama mengenai penggunaan model open-weight SLM untuk penalaran logis dan memberikan rekomendasi untuk penelitian selanjutnya di bidang penalaran logis pada bahasa Indonesia.

BAB 2

LANDASAN TEORI

Bab ini membangun kerangka teoretis yang mendasari analisis kemampuan *Large Language Models* (LLM) dalam melakukan penalaran logika. Pembahasan mencakup prinsip formal dari *First-Order Logic* (FOL), prosedur konversi ke *Conjunctive Normal Form* (CNF), serta algoritma Resolusi yang menjadi mesin utama dalam *framework* Aristotle.

2.1 *First-Order Logic* (FOL)

First-Order Logic (FOL), atau Kalkulus Predikat, adalah logika formal yang memperluas logika proposisi dengan memperkenalkan variabel, fungsi, dan kuantor untuk merepresentasikan objek dan relasi di dunia nyata.

2.1.1 Sintaks: Alfabet dan Aturan Pembentukan

Sintaks FOL dibangun dari komponen-komponen berikut menurut Brachman and Levesque (2004):

- Simbol Logis:
 - Konektif: \neg (Negasi), \wedge (Konjungsi), \vee (Disjungsi), \rightarrow (Implikasi), \leftrightarrow (Bikondisional).
 - Kuantor: \forall (Universal), \exists (Eksistensial).
 - Variabel: x, y, z, \dots
- Simbol Non-Logis:
 - Konstanta: Simbol yang merepresentasikan objek spesifik (misalnya, "Alice", "42").
 - Fungsi ($f(x_1, \dots, x_n)$): Memetakan objek ke objek lain. Contoh: *AyahDari(Budi)*.
 - Predikat ($P(x_1, \dots, x_n)$): Fungsi yang memetakan tuple objek ke nilai kebenaran (True/False). Contoh: *Ayah(Budi, True)*.

2.1.2 Semantik: Interpretasi dan Kebenaran

Kebenaran sebuah kalimat FOL ditentukan oleh sebuah Interpretasi (I) atas Domain (\mathcal{D}) yang tidak kosong. Interpretasi memetakan:

- **Konstanta:** Setiap simbol konstanta c dipetakan ke elemen spesifik di \mathcal{D} .

- **Predikat:** Setiap simbol predikat n -ary dipetakan ke himpunan relasi n -ary di \mathcal{D} (objek-objek mana yang memiliki sifat tersebut).
- **Fungsi:** Setiap simbol fungsi f dipetakan ke operasi konkret pada \mathcal{D} . Misalnya, jika domainnya adalah bilangan bulat, simbol fungsi $plus(x, y)$ dapat diinterpretasikan sebagai operasi penjumlahan $x + y$.

Sebuah formula α dikatakan **Benar** di bawah interpretasi I (ditulis $I \models \alpha$) jika fakta di dunia nyata sesuai dengan struktur kalimat tersebut.

2.2 Konsekuensi Logis (*Entailment*)

Tujuan utama sistem berbasis pengetahuan adalah menarik kesimpulan baru dari informasi yang sudah diketahui. Dalam sistem logika, kumpulan fakta dan aturan yang kita yakini kebenarannya disebut sebagai ***Knowledge Base*** (KB). Menurut Huth and Ryan (2004), $KB \models \alpha$ jika dan hanya jika untuk **setiap** interpretasi I di mana semua kalimat dalam KB bernilai Benar, maka α juga pasti bernilai Benar. Artinya, tidak mungkin ada situasi di mana premis-premis kita benar tetapi kesimpulannya salah. Namun, memeriksa "setiap interpretasi" secara komputasi adalah mustahil karena jumlahnya bisa tak terbatas. Oleh karena itu, kita menggunakan algoritma inferensi sintaksis (seperti Resolusi) untuk membuktikan validitas tersebut secara otomatis.

2.3 Resolusi dan Unifikasi

Metode inferensi utama yang digunakan dalam modul *resolve* Aristotle adalah **Resolusi** (Brachman and Levesque (2004))

2.3.1 Pembuktian Kontradiksi (*Proof by Contradiction*)

Resolusi bekerja dengan prinsip *Proof by Contradiction*. Untuk membuktikan bahwa $KB \models \alpha$, kita dapat menunjukkan bahwa himpunan $KB \cup \{\neg\alpha\}$ adalah *Unsatisfiable* (tidak ada interpretasi yang memenuhi), maka terbukti secara logis bahwa α haruslah benar. Tanda terjadinya kontradiksi adalah ketika algoritma berhasil menurunkan **Klausula Kosong** (\square atau *False*) (Brachman and Levesque (2004)).

2.3.2 Konversi ke *Conjunctive Normal Form* (CNF)

Agar aturan resolusi dapat diterapkan, formula logika harus diubah ke bentuk standar yang disebut *Conjunctive Normal Form* (CNF). Menurut Fitting (1996) merinci langkah-langkah algoritma konversi ini sebagai berikut:

1. Eliminasi Implikasi: Ubah $A \rightarrow B$ menjadi $\neg A \vee B$.
2. Geser Negasi ke Dalam: Gunakan hukum De Morgan dan aturan $\neg \forall x(P) \equiv \exists x(\neg P)$.
3. Standardisasi Variabel: Ubah nama variabel agar unik untuk setiap kuantor (misal: $\forall x(P(x)) \vee \forall x(Q(x))$ menjadi $\forall x(P(x)) \vee \forall y(Q(y))$).
4. Prenex Normal Form: Pindahkan semua kuantor ke depan formula.
5. Skolemisasi: Menghilangkan kuantor eksistensial (\exists). Variabel y yang terikat oleh \exists diganti dengan Fungsi Skolem ($f(x)$) yang bergantung pada variabel universal sebelumnya. Contoh: $\forall x \exists y(Parent(x, y))$ menjadi $\forall x(Parent(x, f(x)))$.
6. Distribusi & CNF: Gunakan aturan distributif untuk mendapatkan bentuk konjungsi dari klausa (AND of ORs).

2.3.3 Aturan Resolusi dan Unifikasi

Aturan resolusi untuk logika predikat adalah:

$$\frac{C_1 \vee L_1, \quad C_2 \vee L_2}{C_1 \vee C_2 \theta}$$

Di mana L_1 dan L_2 adalah literal yang saling berlawanan (misal $P(x)$ dan $\neg P(a)$). Agar mereka bisa saling menghilangkan (*cancel out*), argumen di dalamnya harus disamakan terlebih dahulu.

Proses penyamaan argumen ini disebut **Unifikasi**. Unifikasi mencari substitusi θ (disebut *Most General Unifier / MGU*) yang membuat dua atom menjadi identik secara sintaksis. Contoh: Unifikasi $P(x)$ dan $P(Alex)$ menghasilkan $\theta = \{x / Alex\}$. Tanpa unifikasi, resolusi pada FOL tidak mungkin dilakukan karena variabel pada premis yang berbeda harus diselaraskan.

2.4 Language Models (LM)

Bagian ini membahas teknologi dasar yang memungkinkan implementasi *Logical Translator* dalam penelitian ini.

2.4.1 Arsitektur Transformer

Large Language Models (LLM) modern seperti Qwen, Llama, dan GPT dibangun di atas arsitektur **Transformer** yang diperkenalkan oleh Vaswani et al. (2017). Inovasi utama arsitektur ini adalah mekanisme *Self-Attention*, yang memungkinkan model untuk memproses hubungan antar kata dalam sebuah kalimat secara paralel tanpa bergantung pada urutan sekuensial seperti pada RNN (*Recurrent Neural Networks*). Hal ini memungkinkan model menangkap ketergantungan jarak jauh (*long-range dependencies*) yang penting untuk memahami konteks logika yang kompleks.

2.4.2 Pre-training dan Fine-tuning

LLM dilatih melalui dua tahap utama:

1. **Pre-training:** Model dilatih pada korpus teks yang sangat besar (triliunan token) dengan tujuan memprediksi kata berikutnya (*next-token prediction*) (Brown et al. (2020)). Pada tahap ini, model mempelajari struktur bahasa, fakta dunia nyata, dan pola penalaran dasar secara implisit.
2. **Fine-tuning / Instruction Tuning:** Model disesuaikan lebih lanjut dengan dataset instruksi-jawaban agar mampu mengikuti perintah pengguna, seperti "Terjemahkan kalimat ini ke *First Order Logic*" (Ouyang et al. (2022)). Model yang digunakan dalam penelitian ini (Instruct versions) telah melalui tahap ini.

2.4.3 Inferensi dan Kuantisasi

Proses penggunaan model yang sudah dilatih untuk menghasilkan teks disebut **Inferensi**.

- **Decoding Strategies:** Karena LLM memprediksi probabilitas kata berikutnya, cara kita memilih kata tersebut mempengaruhi hasil.
 - *Greedy Decoding:* Memilih token dengan probabilitas tertinggi di setiap langkah ($\text{temperature}=0$). Metode ini deterministik dan sangat cocok untuk tugas logika yang membutuhkan kepastian sintaks.

- *Sampling*: Memilih token secara acak berdasarkan distribusi probabilitas ($\text{temperature} > 0.7$), cocok untuk tugas kreatif namun buruk untuk logika.
- **Kuantisasi (GGUF)**: Menjalankan LLM membutuhkan memori (VRAM) yang besar. Untuk mengatasi ini pada perangkat keras terbatas, digunakan teknik kuantisasi, yaitu mengurangi presisi bobot model dari 16-bit (*half precision*) menjadi 4-bit atau 8-bit bilangan bulat (Dettmers et al. (2022)).

2.4.4 Definisi dan Karakteristik Small Language Models (SLM)

Dalam perkembangannya, paradigma model bahasa mulai bergeser dari "semakin besar semakin baik" menjadi efisiensi. *Small Language Models* (SLM) didefinisikan secara longgar dalam literatur sebagai model dengan jumlah parameter berkisar antara beberapa juta hingga batas atas sekitar 7 hingga 10 miliar parameter (Subramanian et al. (2025); Wang et al. (2025)).

Karakteristik utama SLM meliputi:

- **Efisiensi Parameter**: SLM mencapai performa kompetitif dengan jumlah parameter yang jauh lebih sedikit (10-100x lebih kecil dari LLM), memungkinkan inferensi yang lebih cepat dan penggunaan memori yang lebih rendah.
- **(Edge Deployability)**: Karena ukurannya yang ringan, SLM dapat dijalankan secara lokal pada perangkat konsumen standar, seperti laptop, bahkan ponsel, tanpa memerlukan kluster GPU *data center*. Hal ini memberikan keuntungan dalam privasi data dan kedaulatan informasi (Belcak et al. (2025)).
- **Teknik Kompresi**: Performa tinggi SLM sering kali dicapai melalui teknik *Knowledge Distillation* (belajar dari model guru yang lebih besar) dan pelatihan pada dataset berkualitas sangat tinggi atau *textbook quality*, serta penggunaan teknik (*pruning*) (Wang et al. (2025)).

Model seperti Microsoft Phi-3, Mistral 7B, dan Gemma 2B adalah contoh representatif dari kelas model ini yang menunjukkan kemampuan penalaran yang menarik meskipun berukuran kecil ("Tiny but Mighty")¹

¹<https://news.microsoft.com/source/features/ai/the-phi-3-small-language-models-with-big-potential/>

2.5 Dataset ProntoQA

Untuk mengukur kemampuan penalaran secara objektif, penelitian ini mengadaptasi dataset ProntoQA yang dikembangkan oleh Saparov and He (2023).

2.5.1 Generasi Data Sintetis

ProntoQA berbeda dari dataset tanya-jawab umum karena sifatnya yang generatif dan sintetis. Data tidak diambil dari internet, melainkan dibangkitkan dari aturan logika formal yang kemudian diterjemahkan ke bahasa alami. Proses pembuatannya adalah:

1. Sistem membuat graf ontologi fiktif (misal: Setiap Wumpus adalah Jompus dan Setiap Jompus adalah Tumpus).
2. Sistem menurunkan rantai deduksi berdasarkan graf tersebut.
3. Simbol-simbol logika diganti dengan kata-kata fiktif atau nyata untuk membentuk kalimat bahasa alami.

2.5.2 Mengisolasi Kemampuan Penalaran

Keunggulan utama ProntoQA adalah kemampuannya mengisolasi kemampuan deduksi dari pengetahuan dunia (*world knowledge*). Menggunakan istilah fiktif (seperti "Wumpus" atau "Tumpus") mencegah model menjawab benar hanya karena hafal fakta (misalnya "Burung bisa terbang"). Model dipaksa untuk melakukan penalaran ("Jika Wumpus adalah Burung, maka Wumpus bisa terbang") berdasarkan aturan yang diberikan di dalam *prompt*, bukan dari memori pelatihannya. Dataset ini juga memungkinkan kontrol terhadap "kedalaman penalaran" (*reasoning hops*), yaitu berapa langkah logika yang diperlukan untuk mencapai kesimpulan (misal: 1-hop, 3-hop, hingga 5-hop).

BAB 3

FRAMEWORK TRANSLATION-DECOMPOSITION-SEARCH-RESOLVE

Bab ini menguraikan rancangan operasional yang digunakan untuk menjawab rumusan masalah penelitian. Metodologi ini disusun untuk menguji secara empiris apakah pendekatan dapat mengatasi kelemahan penalaran probabilistik pada *open-weight Small Language Models* (SLM) dalam konteks Bahasa Indonesia.

Pendekatan penelitian ini bersifat kuantitatif-eksperimental. Fokus utamanya adalah mengukur akurasi penalaran logis pada bahasa Indonesia yang diperoleh melalui manipulasi struktur *prompting Neuro-Symbolic* dan dampaknya ketika model dijalankan dengan sumber daya terbatas (melalui kuantisasi).

3.1 Konfigurasi Model

Mengingat bahwa adanya batasan sumber daya komputasi, maka eksperimen dijalankan menggunakan kuantisasi pada setiap model yang dijalankan menggunakan *tools open-source*, yaitu *llama.cpp*¹.

3.1.1 Pemilihan Model

Model yang dipilih mewakili tiga tingkatan spesialisasi bahasa untuk menguji hipotesis bahwa pemahaman dalam bahasa lokal membantu proses dekomposisi logika:

1. **Qwen2.5-7B-Instruct (Representasi Global):** Model ini dipilih karena performanya yang unggul dalam benchmark logika matematika global, menjadi titik ukur batas atas kemampuan model *open-weight* saat ini.
2. **SEA-LION-v3-Llama-8B-Instruct (Representasi Regional):** Model yang telah melalui *continued pre-training* dengan data bahasa dari negara-negara di Asia Tenggara, diharapkan memiliki pemahaman semantik yang lebih baik terhadap struktur kalimat regional.
3. **SahabatAI-v1-Llama-8B-Instruct (Representasi Nasional):** Model yang dikhususkan untuk Bahasa Indonesia, juga sudah melalui *continued pre-training*, digunakan untuk melihat apakah spesialisasi bahasa yang mendalam dapat menyelesaikan penalaran

¹<https://github.com/ggml-org/llama.cpp>

lebih baik dalam tugas logika.

3.1.2 Parameterisasi Inferensi

Setiap model dijalankan dengan pengaturan parameter yang sama / konsisten untuk memastikan hasil eksperimen mencerminkan pengaruh variabel bebas, bukan perbedaan konfigurasi teknis. Strategi deterministik diterapkan untuk menghilangkan unsur *randomness* dalam proses inferensi.

```

1  class LlamaCPPBackend:
2      def __init__(self, local_model_path: str):
3          """
4              local_model_path: Must point to a specific .gguf FILE, not just a directory.
5          """
6          if not LLAMACPP_AVAILABLE: raise ImportError("llama-cpp-python not installed.
7              Run 'pip install llama-cpp-python'")
8
9          # n_gpu_layers=-1 means offload ALL layers to GPU
10         self.llm = Llama(
11             model_path=local_model_path,
12             n_ctx=0,
13             n_gpu_layers=-1,
14             verbose=False
15         )
16
17     def generate(self, prompt: str, max_new_tokens: int = 512, temperature: float =
18                 0.0, **kwargs) -> str:
19         output = self.llm(
20             prompt,
21             max_tokens=max_new_tokens,
22             stop=[],
23             echo=True, # Return prompt + completion to match others
24             temperature=temperature
25         )
26         return output['choices'][0]['text']

```

Kode 3.1: Konfigurasi parameter deterministik untuk setiap model yang digunakan

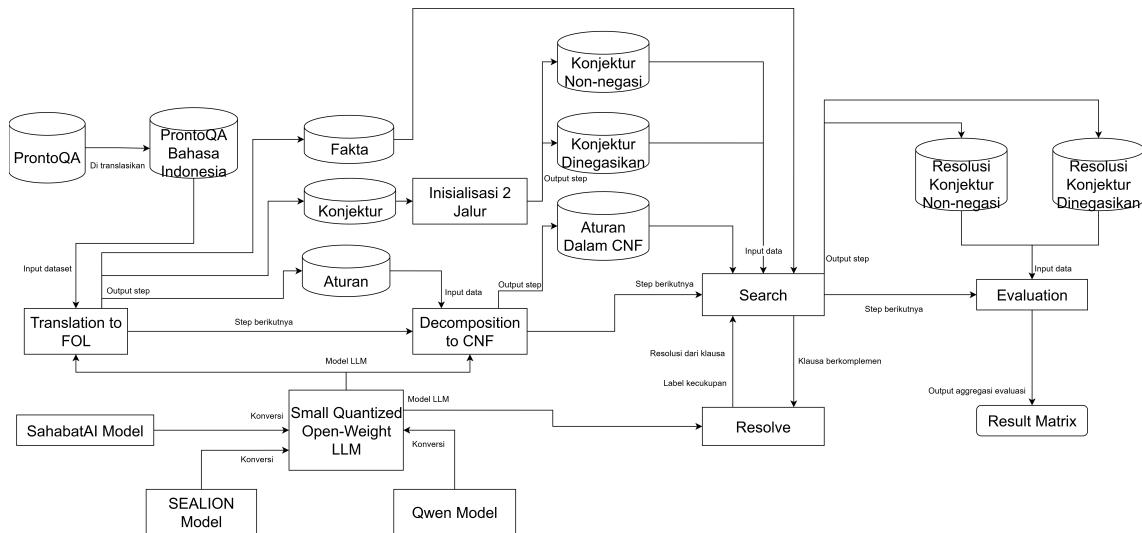
Parameter utama yang dikontrol adalah:

- **Temperature:** Ditetapkan ke 0.0 sehingga model hanya memilih token dengan probabilitas tertinggi tanpa penambahan noise acak (*greedy decoding*). Hal ini memastikan setiap prompt menghasilkan output yang identik.
- **Max Tokens:** Dibatasi sesuai dengan panjang output yang wajar untuk setiap tahap dalam pipeline, mencegah generasi yang berlebihan. Adapun token yang dibutuhkan

untuk setiap tahap inferensi dalam *framework* antara lain: proses translasi ke FOL memerlukan output setidaknya 400 token, proses dekomposisi setidaknya memerlukan output 700 token, dan proses pencarian bukti setidaknya memerlukan 1000 token., sehingga ***max tokens*** yang di set untuk semua proses adalah 2500 untuk mengatasi *overhead* atau halusinasi jika model tidak secara *strict* mengikuti format.

- ***n_ctx***: Ukuran konteks diatur untuk menentukan berapa banyak token sebelumnya yang dapat dipertimbangkan model saat menghasilkan respons. Nilai yang lebih besar memungkinkan model memahami konteks yang lebih panjang, namun meningkatkan penggunaan memori. ***n_ctx*** di set ke 0, sehingga ukuran konteks akan sesuai dengan kemampuan model masing-masing.
 - ***n_gpu_layers***: Menentukan jumlah layer model yang dijalankan di GPU untuk akselerasi. Nilai yang lebih tinggi mempercepat inferensi tetapi memerlukan VRAM yang lebih besar. Nilai -1 berarti semua komputasi dilakukan di GPU.

3.2 Kerangka Kerja Aristotle



Gambar 3.1: Kerangka kerja Aristotle untuk inferensi logika menggunakan SLM dan metode *neuro-symbolic* (Xu et al. (2025))

Penelitian ini mengimplementasikan **framework Aristotle** (Xu et al. (2025)) yang membagi proses inferensi menjadi empat modul utama, sesuai dengan Gambar 3.1.

3.2.1 Translasi Logika (*Logical Translation*)

Ini adalah satu-satunya tahap yang menggunakan "kecerdasan" probabilistik SLM. Tujuannya adalah memetakan kalimat Bahasa Indonesia yang implisit menjadi struktur logika formal.

- **Tugas Model:** Menerjemahkan narasi bahasa alami menjadi representasi *First-Order Logic* (FOL) yang terdiri dari Fakta, Aturan Implikasi, dan Konjektur.
- **Mekanisme Eksekusi:** Model diberikan *few-shot prompt* yang berisi contoh pasangan kalimat Indonesia dan format logika target.
- **Format Output:**
 - **Fakta:** $P(\text{Subjek}, \text{NilaiKebenaran})$. Contoh: *Wumpus(Max, True)*
 - **Aturan:** $P(\text{Subjek}, \text{NilaiKebenaran}) >>> P(\text{Objek}, \text{NilaiKebenaran})$. Contoh: *Wumpus(\$x, True) >>> Manis(\$x, False)*
 - **Konjektur:** $P(\text{Subjek}, \text{NilaiKebenaran})$. Contoh: *Manis(Max, True)*

3.2.2 Dekomposisi (*Decomposer*)

Pada tahap ini, output FOL diproses lebih lanjut untuk memenuhi standar mesin pembuktian teorema. Model diinstruksikan untuk melakukan konversi ke Prenex Normal Form (PNF) dan kemudian ke Conjunctive Normal Form (CNF). Proses penting di sini adalah Skolemisasi, yaitu penghapusan kuantor eksistensial ($\exists x$) yang sering menjadi sumber ambiguitas bagi model bahasa. Dalam dataset ProntoQA, tidak perlu dilakukan skolemisasi karena premis-premis dari dataset tersebut tidak memiliki kuantor eksistensial.

- **Tugas Model:** Mengonversi aturan FOL menjadi *Conjunctive Normal Form* (CNF) melalui serangkaian transformasi logis.
- **Mekanisme Eksekusi:**
 - **PNF:** Konversi FOL ke bentuk PNF, di mana semua kuantor diletakkan di awal formula, diikuti oleh matriks bebas kuantor.
 - **Skolemisasi:** Penghapusan kuantor eksistensial ($\exists x$) dengan mengganti variabel eksistensial dengan fungsi Skolem. Dataset ProntoQA tidak memerlukan langkah ini.
 - **CNF:** Transformasi akhir ke CNF, di mana formula diekspresikan dalam bentuk konjungsi (AND) dari disjungsi (OR) literal.

3.2.3 Penyelesaian Pencarian (*Search Router*)

Mesin inti inferensi terdiri dari dua sub-komponen:

1. **Logical Search Router:** Memilih klausa mana yang relevan untuk diproses. Hal ini mencegah proses komputasi yang memerlukan resource terlalu banyak dan yang lama dalam pencarian bukti. Di sini juga dideklarasikan *search_round* yaitu batas maksimum iterasi pencarian bukti.
2. **Logical Resolver:** Menerapkan aturan resolusi secara iteratif hingga ditemukan kontradiksi (untuk pembuktian salah) atau hingga ruang pencarian habis.

3.2.4 Resolusi (*Logical Resolver*)

Ini adalah mesin inferensi deterministik yang mengantikan peran penalaran SLM.

- **Prinsip:** Menggunakan aturan resolusi berbasis pembuktian kontradiksi (*Proof By Contradiction*) untuk menilai kebenaran hipotesis *S* berdasarkan premis *P* (2.3.1)
- **Mekanisme Dual-Path:** Sistem mencoba membuktikan kebenaran hipotesis *S* melalui dua jalur paralel pembuktian kontradiksi:
 1. **Jalur A:** Asumsikan $\neg S$ (Negasi *S*), jika ditemukan kontradiksi (Klausa Kosong), maka *S* terbukti **Benar**.
 2. **Jalur B:** Asumsikan *S*, jika ditemukan kontradiksi, maka $\neg S$ terbukti **Benar**.
- Jika tidak ada kontradiksi yang ditemukan di kedua jalur (misalnya karena informasi tidak lengkap atau translasi putus), sistem mengembalikan **Unknown**.

Mekanisme injeksi instruksi ke dalam *prompt template* untuk setiap tahap diimplementasikan menggunakan kode dalam bahasa pemrograman Python berikut:

```

1 def construct_prompt_a(self, record, in_context_examples_trans):
2     full_prompt = in_context_examples_trans
3     if self.dataset_name == "LogicNLI":
4         context = "\n".join(record['facts'] + record['rules'])
5         question = record['conjecture']
6     else:
7         context = record['context']
8         question = re.search(r'\??.*', record['question'].strip()).group(1).strip()
9     full_prompt = full_prompt.replace('[[PREMISES]]', context)
10    full_prompt = full_prompt.replace('[[CONJECTURE]]', question)
11    return full_prompt

```

Kode 3.2: Mekanisme injeksi data poin ke dalam template prompt Aristotle

Potongan kode di atas menunjukkan fungsi *construct_prompt_a* yang bertugas membangun *prompt* dinamis untuk masalah logika, dalam hal ini fungsi tersebut digunakan untuk membangun *prompt* untuk tahap *translation*. Fungsi ini membaca *template* dasar yang memuat instruksi sistem dan contoh *few-shot*, lalu mengganti *placeholder*, seperti PREMISES dan CONJECTURE dengan data dari dataset ProntoQA yang sedang diuji. Pendekatan injeksi string ini memastikan bahwa struktur instruksi tetap konsisten (statis) sementara konten masalah berubah-ubah, meminimalkan risiko model "lupa" terhadap format output yang diminta saat beralih ke soal baru. Untuk setiap tahap dalam *framework* Aristotle, kode serupa diterapkan dengan penyesuaian pada *template* dan *placeholder* yang relevan, misalnya *template* untuk tahap *decomposition* akan berbeda dengan tahap *translation*, tetapi kode fungsi pembangun *prompt* tetap mengikuti pola yang sama.

3.3 Teknik Evaluasi dan Analisis Data

Untuk menjamin objektivitas, evaluasi tidak dilakukan secara manual melainkan menggunakan sistem ekstraksi berbasis pola (*Regular Expression / Regex*).

3.3.1 Parsing Bertingkat (Multi-stage Parsing)

Sistem evaluasi dirancang untuk "menangkap" struktur logika dari output teks model. Kegagalan model dalam mematuhi format yang diminta pada tahap apa pun akan langsung dianggap sebagai kesalahan (*failure*), tanpa upaya perbaikan manual. Ini adalah standar ketat yang diterapkan untuk menguji keandalan model sebagai komponen sistem logika.

- **Ekstraksi FOL:** Mengambil Fakta, Aturan, dan Konjektur sesuai bloknya

```

1 def extract_facts_rules_conjecture(self, content, context_sentence_count=None):
2     # Clean invisible characters
3     content = (content or "").replace("\u200b", '').replace('\ufeff', '')
4
5     prompt_marker = re.compile(
6         r'Di bawah ini(?:\s+adalah(?:\s+yang\s+perlu\s+Anda\s+terjemahkan)?)?:?',
7         re.IGNORECASE
8     )
9     m_prompt = prompt_marker.search(content)
10    search_start_pos = m_prompt.end() if m_prompt else 0
11
12    block_header = re.compile(r'\*\{0,3\} Bentuk Akhir\*\{0,3\}', re.IGNORECASE)
13    m_block = block_header.search(content, pos=search_start_pos)
14
15    if m_block:

```

```

16     area = content[m_block.end():]
17 else:
18     area = content[search_start_pos:]
19
20 # Define Patterns for possible headers
21 # Include the "End Markers" as a header type.
22 patterns = {
23     "facts": re.compile(r'(?:Fakta|Facts)\s*[:\n]?\s*', re.IGNORECASE),
24     "rules": re.compile(r'(?:Aturan|Rules)\s*[:\n]?\s*', re.IGNORECASE),
25     "conj": re.compile(r'(?:Konjektur|Conjecture)\s*[:\n]?\s*', re.IGNORECASE),
26     "stop": re.compile(r'(\*\{0,3\}\s*Akhir Blok\s*\*\{0,3\}|###|\^\^|\-\{3,\})',
27     re.IGNORECASE)
28 }
29
30 def extract_section(target_key):
31     start_match = patterns[target_key].search(area)
32     if not start_match:
33         return ""
34
35     content_start_idx = start_match.end()
36
37     # Find the next header
38     next_indices = []
39     for key, pat in patterns.items():
40         m = pat.search(area, pos=content_start_idx)
41         if m:
42             next_indices.append(m.start())
43
44     # If found upcoming headers, stop at the nearest one (min index).
45     # If no headers found, go to end of string.
46     if next_indices:
47         cutoff_idx = min(next_indices)
48         raw_text = area[content_start_idx:cutoff_idx]
49     else:
50         raw_text = area[content_start_idx:]
51
52     return raw_text.strip()
53
54 facts = extract_section("facts")
55 rules = extract_section("rules")
56 conjecture = extract_section("conj")
57
58 return facts, rules, conjecture

```

Kode 3.3: Regex untuk ekstraksi blok Translasi

Kode di atas menggunakan ekspresi reguler (*Regular Expressions*) untuk mem-*parsing output* dari tahap *Logical Translation*. Pola regex dirancang untuk menangkap tiga komponen utama: blok "Fakta", "Konjektur" (seperti Manis(Jeruk)) dan blok "Aturan"

(seperti Jeruk(x) >>> Manis(x)). Jika model menghasilkan teks naratif atau penjelasan di luar format yang ditentukan, regex ini akan mengabaikannya dan hanya mengambil struktur logika yang valid. Jika tidak ada struktur yang cocok, fungsi akan mengembalikan nilai kosong yang menandakan kegagalan translasi (*parsability error*).

- **Ekstraksi CNF:** mengambil hasil akhir dari dekomposisi dari Aturan FOL.

```

1 def post_process_decompose(self, content, rules_count=None):
2
3     content = (content or "").replace('\u200b', '').replace('\ufeff', '')
4
5     marker_pattern = r'Di bawah ini adalah yang perlu Anda konversikan menggunakan
6     normalisasi.'
7
8     marker_match = re.search(marker_pattern, content, flags=re.IGNORECASE)
9
10    block_header = re.compile(r'^{0,3}Bentuk Akhir^{0,3}', re.IGNORECASE)
11    m_block = block_header.search(content, pos=marker_match.end())
12
13    area = content[m_block.end():]
14
15    cnf_label_re = re.compile(
16        r'(?:Aturan dalam CNF|Aturan CNF|Aturan|Rules)\s*[:\n-]?\s*',
17        flags=re.IGNORECASE
18    )
19    skolem_label_re = re.compile(
20        r'(?:Aturan dalam Skolem|Skolemisasi|Skolem|Bentuk Akhir Setelah
21        Skolemisasi|Skolemization)\s*[:\n-]?\s*',
22        flags=re.IGNORECASE
23    )
24
25    # Construct the pattern string explicitly to avoid parentheses nesting errors.
26    boundary_pattern = (
27        r'(?:'                                         # Start outer group
28            r'\r?\n\s*'                                # Newline +
29        whitespace
30            r'(?:'                                         # Start inner
31        grouping for headers
32            r'(?:Aturan dalam CNF|Aturan CNF|Aturan \\\(CNF\\\)|Aturan|Rules) |'  # CNF headers
33            r'(?:(Skolemisasi|Skolem|Bentuk Akhir))|'          # Skolem headers
34            r'(?:(^{0,3}\s*Akhir Blok\s*)^{0,3}|Final Form|###)' # End markers
35            r'))'                                         # End inner
36        grouping
37            r')'                                         # End outer group
38            r'|$'                                         # OR End of String
39    )
40
41    boundary_re = re.compile(boundary_pattern, flags=re.IGNORECASE)
42
43    def extract_after_label(label_re):

```

```

38     """Finds label, returns text until next boundary."""
39     lab_match = label_re.search(area)
40     if not lab_match:
41         return None
42     start = lab_match.end()
43     bound = boundary_re.search(area, pos=start)
44     end = bound.start() if bound else len(area)
45     return area[start:end].strip()
46
47 cnf_raw = extract_after_label(cnf_label_re)
48 skolem_raw = extract_after_label(skolem_label_re)
49
50 def to_lines(raw):
51     if not raw:
52         return []
53     return [ln.strip() for ln in raw.splitlines() if ln.strip()]
54
55 cnf_lines = to_lines(cnf_raw)
56 skolem_lines = to_lines(skolem_raw) if skolem_raw else None
57
58 print(f"CNF Raw: {cnf_lines}")
59 print(f"Skolem Raw: {skolem_lines}")
60
61 return cnf_lines, skolem_lines

```

Kode 3.4: Regex untuk ekstraksi blok Dekomposisi

Serupa dengan tahap sebelumnya, kode pada Kode 3.4 bertugas memvalidasi output dari tahap *Decomposer*. Regex ini secara spesifik mencari pola "Aturan dalam CNF". Jika model tidak mengikuti format yang diharapkan, regex ini akan gagal menemukan kecocokan, menandakan kegagalan dekomposisi.

- **Ekstraksi Resolusi:** Mendeteksi klausa baru dan kesimpulan akhir dalam label kecukupan.

```

1 def post_process_logic_solver(self, response_d):
2     content = response_d
3     marker_pattern = r'(.*)Dibawah ini tugas yang perlu Anda lakukan(.*)'
4     marker_match = re.search(marker_pattern, content, flags=re.IGNORECASE)
5     search_area = content[marker_match.end():] if marker_match else content
6
7     final_block_pattern = (
8         r'\*{0,3}(?:Bentuk Akhir)\*{0,3}\s*'
9         r'(.*?'
10        r'(?:\*{0,3}(?:Akhir Blok)\*{0,3})|'
11        r'$)' # or end of string
12    )
13
14     final_block_match = re.search(final_block_pattern, search_area, flags=re.DOTALL |
re.IGNORECASE)

```

```

15
16     if not final_block_match:
17         return [], None
18
19     block = final_block_match.group(1)
20
21     block_clean = block.strip()
22     print(f"\n\nCHOSEN BLOCK:\n{n{block_clean}}\n")
23     print("END OF CHOSEN BLOCK\n\n")
24
25     clause_pos = re.search(r'Clause\s*Baru', block_clean, flags=re.IGNORECASE)
26     clause_after = block_clean[clause_pos.end():]
27     m_new = re.search(r'\{(.*?)\}', clause_after, flags=re.DOTALL)
28
29     if not m_new:
30         raise ValueError(f"'Clause Baru:' with '{...}' not found in expected form.")
31
32     new_clause = m_new.group(1).strip()
33
34     label_pos = re.search(r'Label\s*Cukup', block_clean, flags=re.IGNORECASE)
35     label_after = block_clean[label_pos.end():]
36     m_label = re.search(r'\[(.*?)\]', label_after, flags=re.DOTALL)
37     if not m_label:
38         raise ValueError(f"'Label Cukup' with '[...]' not found in expected form")
39     [True|False].")
40
41     sufficiency_label = m_label.group(1).strip()
42
43     return {
44         "new_clause": new_clause,
45         "sufficiency_label": sufficiency_label,
46     }

```

Kode 3.5: Regex untuk memvalidasi langkah Resolusi

Kode 3.5 di atas adalah komponen validasi akhir yang bekerja pada output modul *Resolver*. Regex ini bertugas mengekstraksi dua informasi vital dari logika resolusi SLM: (1) Clause Baru, yaitu kesimpulan antara yang dihasilkan dari penggabungan dua premis, dan (2) Label Cukup, yang menandakan apakah kontradiksi telah ditemukan (*True*) atau belum (*False*). Jika format ini tidak ditemukan, sistem akan menganggap langkah resolusi tersebut gagal atau tidak konklusif.

3.3.2 Matriks Keputusan Evaluasi (Truth Table)

Kinerja akhir model tidak ditentukan secara probabilistik, melainkan melalui tabel kebenaran logika berdasarkan hasil dari *Logical Resolver*. Untuk setiap pertanyaan *S*, sistem

menjalankan dua pembuktian paralel. Tabel 3.1 menunjukkan logika penentuan jawaban akhir.

Tabel 3.1: Matriks Keputusan untuk Penentuan Label Jawaban (Kesimpulan)

Kondisi (Skenario)	Jalur A: Buktikan S (Cari Kontradiksi di $\neg S$)	Jalur B: Buktikan $\neg S$ (Cari Kontradiksi di S)	Kesimpulan
1	Berhasil (Ada Kontradiksi)	Gagal	TRUE
2	Gagal	Berhasil (Ada Kontradiksi)	FALSE
3	Gagal	Gagal	UNKNOWN
4	Berhasil	Berhasil	SELF-CONTRADICTORY

Keterangan:

- **Berhasil (Ada Kontradiksi):** Resolver menemukan klausa kosong, yang berarti asumsi awal salah, sehingga kebalikannya pasti benar.
- **TRUE:** Premis mendukung hipotesis S .
- **FALSE:** Premis menyangkal hipotesis S .
- **UNKNOWN:** Informasi pada premis tidak cukup untuk membuktikan atau menyangkal S .
- **SELF-CONTRADICTORY:** Premis itu sendiri tidak konsisten (mengandung konflik internal).

BAB 4

EKSPERIMENT, HASIL, DAN ANALISIS

Bab ini memaparkan desain eksperimen, hasil eksperimen empiris yang dilakukan untuk mengevaluasi kinerja *open-weight SLM* pada tugas penalaran logis berbahasa Indonesia. Analisis dibagi menjadi tiga bagian utama: (1) proses iteratif penyempurnaan *prompt* untuk mengatasi ambiguitas bahasa, (2) hasil evaluasi kuantitatif menggunakan metrik akurasi, dan (3) analisis kualitatif mendalam mengenai penyebab kegagalan dan keberhasilan masing-masing model dalam kerangka kerja *Neuro-Symbolic*.

4.1 Desain Eksperimen

Eksperimen dirancang dengan membandingkan performa model dalam dua skenario inferensi:

1. **Skenario Pertama (*Naive Prompting*):** Pada skenario ini, model diuji kemampuan "intuisi"-nya. Masalah logika diberikan dalam bentuk narasi langsung dengan instruksi standar dan beberapa contoh penggeraan (*few-shot*). Skenario ini merepresentasikan cara penggunaan SLM pada umumnya, di mana model dipaksa melakukan logika secara implisit.
2. **Skenario Kedua (*Framework Aristotle*):** Skenario ini menerapkan (*pipeline*) yang memaksa model untuk mengeksternalisasi proses berpikirnya ke dalam simbol-simbol logika formal. Sesuai landasan teori, proses ini dipecah menjadi tahapan sekuensial: *Translation* → *Decomposition* → *Search* → *Resolve*.

Struktur variabel penelitian didefinisikan sebagai berikut:

- **Variabel Bebas (*Independent Variables*):**
 - **Model:** Tiga kategori model yang mewakili spektrum pemahaman bahasa: Global (Qwen), Regional (SEA-LION), dan Nasional (SahabatAI).
 - **Metode Prompting:** *Naive* vs *Aristotle Framework*.
- **Variabel Terikat (*Dependent Variables*):**
 - **Akurasi (*Accuracy*):** Ketepatan hasil akhir (Benar/Salah) yang diverifikasi terhadap *ground truth*.
 - **Kepatuhan Format (*Parsability*):** Kemampuan model menghasilkan sintaks logika

(FOL/CNF) yang valid secara komputasi. Kegagalan menghasilkan format yang benar dianggap sebagai kegagalan penalaran.

- **Variabel Kontrol:**

- Seluruh eksperimen dijalankan secara deterministik dengan parameter yang tetap. Hal ini meniadakan faktor (*randomness*) dalam resolusi inferensi.
- Setiap model terkuantisasi (4-bit GGUF)¹

4.2 Instrumen Data dan Adaptasi Linguistik

Penelitian ini menggunakan dataset **ProntoQA** (*Prompting with Ontologies for QA*) Saparov and He (2023). Pemilihan dataset ini didasarkan pada kebutuhan untuk menguji kemampuan SLM dalam deduksi sintetis, terlepas dari pengetahuan dunia nyata dari SLM tersebut. ProntoQA menggunakan ontologi fiktif (misalnya: ”*Setiap Wumpus adalah Jompus*”), sehingga model tidak dapat menjawab dengan mengandalkan hafalan dalam *pre-training*.

Proses penerjemahan dilakukan secara otomatis dengan *prompting* dan membeberikan beberapa contoh dalam proses translasinya. Penerjemahan dilakukan dengan *open-weight* SLM yaitu **Gemma-SEA-LION-v3-9B-IT**. Model dipilih karena kemampuan yang baik dalam translasi dari bahasa Inggris ke bahasa Indonesia berdasarkan SEA-HELM *leaderboard*²

- **Konsistensi Kuantor:** Frasa ”Every X is Y” diterjemahkan menjadi ”Setiap X adalah Y” atau ”Semua X adalah Y” untuk menjaga pola *Universal Quantifier* (\forall).
- **Konsistensi Negasi:** Struktur ”X is not Y” dipetakan menjadi ”X bukan Y” atau ”X tidak Y”.
- **Konsistensi Plural:** Subjek seperti ”Wumpuses”, ”Jompuses”, dan subjek lainnya, tidak diterjemahkan imbuhan untuk menjaga struktur kalimat.

4.3 *Prompt Refining*

Sebelum melakukan evaluasi skala penuh, dilakukan serangkaian eksperimen pendahuluan untuk menstabilkan performa model dalam menjalankan empat modul utama *framework Aristotle: Translation, Decomposition, Search & Resolve*. Mengingat model yang digu-

¹<https://github.com/ggml-org/llama.cpp>

²<https://leaderboard.sea-lion.ai/detailed/ID>

nakan adalah model berparameter kecil (7B-9B) yang dikuantisasi, model sangat sensitif terhadap struktur instruksi.

4.3.1 Translation to First Order Logic

Pada tahap ini dilakukan prompt untuk penerjemahan kalimat bahasa natural ke dalam *First Order Logic*. Tantangan utama pada tahap ini adalah memetakan kalimat Bahasa Indonesia yang implisit ke dalam struktur logika yang eksplisit.

Berikut adalah prompt template yang digunakan yang merupakan hasil dari translasi langsung dari prompt yang digunakan pada *Aristotle Framework*

```

1 Deskripsi Tugas: Anda diberikan sebuah deskripsi permasalahan dan pertanyaan. Tugasnya
   adalah:
2 1) parsing konteks menjadi aturan logika berdasarkan predikat yang ditentukan
3 2) menulis semua fakta yang disebutkan dalam masalah, dan pastikan mereka diterjemahkan
   dalam bentuk logika juga.
4 3) parsing konjektur menjadi bentuk logika
5 4) silakan berikan output akhir dalam format berikut. Pastikan Anda menerjemahkan dalam
   urutan berikut: fakta, aturan, kemudian konjektur.

6
7 -----
8
9 Berikut adalah contohnya:
10 Konteks:
11 Setiap jompus memiliki sifat eksentris. Setiap jompus adalah wumpus. Setiap wumpus
   memiliki sifat tidak transparan. Wumpuses adalah tumpuses. Tumpuses bersifat nakal.
   Tumpuses adalah vumpuses. Setiap vumpus memiliki sifat dingin. Setiap vumpus adalah
   yumpus. Yumpuses berwarna oranye. Yumpuses adalah Numpuses. Numpuses membosankan.
   Setiap numpus adalah Dumpus. Setiap dumpus memiliki sifat bukan pemalu. Impuses
   memiliki sifat pemalu. Dumpuses adalah rompuses. Setiap rompus tidaklah berwujud cair.
   Rompuses adalah zumpus. Alex adalah tumpus.
12
13 Konjektur:
14 Alex bukan pemalu.
15
16 ###
17 ***Bentuk Akhir***
18
19 Fakta:
20 Alex adalah tumpus :::: Tumpus(Alex, True)
21
22 Aturan:
23 Jompus($x, True) >>> Eksentris($x, True)
24 Jompus($x, True) >>> Wumpus($x, True)
25 Wumpus($x, True) >>> Transparan($x, False)
26 Wumpus($x, True) >>> Tumpus($x, True)
27 Tumpus($x, True) >>> Nakal($x, True)
```

```

28 Tumpus($x, True) >>> Vumpus($x, True)
29 Vumpus($x, True) >>> Dingin($x, True)
30 Vumpus($x, True) >>> Yumpus($x, True)
31 Yumpus($x, True) >>> Oranye($x, True)
32 Yumpus($x, True) >>> Numpus($x, True)
33 Numpus($x, True) >>> Membosankan($x, True)
34 Numpus($x, True) >>> Dumpus($x, True)
35 Dumpus($x, True) >>> Pemalu($x, False)
36 Impus($x, True) >>> Pemalu($x, True)
37 Dumpus($x, True) >>> Rompus($x, True)
38 Rompus($x, True) >>> Cair($x, False)
39 Rompus($x, True) >>> Zumpus($x, False)
40
41 Konjektur:
42 Pemalu(Alex, False)
43
44 ***Akhir Blok***
45
46 -----
47
48 Berikut adalah contohnya:
49 Konteks:
50 Jompuses bukan pemalu. Jompuses adalah yumpuses. Setiap yumpus bersifat agresif. Setiap
yumpus adalah dumpus. Dumpuses bukan terbuat dari kayu. Dumpuses adalah wumpuses.
Wumpuses berwarna merah. Setiap wumpus adalah impus. Setiap impus tidak tembus
pandang. Impuses adalah tumpuses. Numpuses asam. Tumpuses tidak asam. Tumpuses adalah
vumpuses. Vumpuses tidak terbuat tanah. Setiap vumpus adalah zumpus. Zumpuses
berukuran kecil. Zumpuses adalah rompuses. Max adalah yumpus.
51
52 Konjektur:
53 Max asam.
54
55 ###
56 ***Bentuk Akhir***
57
58 Fakta:
59 Max adalah yumpus :::: Yumpus(Max, True)
60
61 Aturan:
62 Jompus($x, True) >>> Pemalu($x, False)
63 Jompus($x, True) >>> Yumpus($x, True)
64 Yumpus($x, True) >>> Agresif($x, True)
65 Yumpus($x, True) >>> Dumpus($x, True)
66 Dumpus($x, True) >>> Kayu($x, False)
67 Dumpus($x, True) >>> Wumpus($x, True)
68 Wumpus($x, True) >>> Merah($x, True)
69 Wumpus($x, True) >>> Impus($x, True)
70 Impus($x, True) >>> TembusPandang($x, False)
71 Impus($x, True) >>> Tumpus($x, True)
72 Numpus($x, True) >>> Asam($x, True)

```

```

73 Tumpus($x, True) >>> Asam($x, False)
74 Tumpus($x, True) >>> Vumpus($x, True)
75 Vumpus($x, True) >>> Tanah($x, False)
76 Vumpus($x, True) >>> Zumpus($x, True)
77 Zumpus($x, True) >>> Kecil($x, True)
78 Zumpus($x, True) >>> Rompus($x, True)
79
80 Konjektur:
81 Asam(Max, True)
82
83 ***Akhir Blok***
84
85 -----
86
87 Di bawah ini adalah yang perlu Anda terjemahkan:
88 Konteks:
89 [[PREMISES]]
90
91 Konjektur:
92 [[CONJECTURE]]

```

Kode 4.1: Template prompt hasil translasi langsung dari bahasa Inggris

Namun dari prompt template tersebut masih bermasalah karena ada beberapa aturan yang tidak dapat diterjemahkan dengan baik oleh model. Berikut adalah contoh aturan yang tidak dapat diterjemahkan dengan baik oleh model menggunakan prompt template 4.1:

- Kesalahan dalam proses *translation to FOL*, khususnya pada subjek dan objek:
 - Premis: ”Zumpus adalah transparan. Zumpus mengajar rompus. Rompuses merupakan tanah. Semua rompus adalah Impus.”
 - Hasil *translation*: ”1. *Zumpus(\$x, True) >>> Transparan(\$x, True)* 2. *Zumpus(\$x, True) >>> Rompuses(\$x, True)* 3. *Rombus(\$x, True) >>> Tanah(\$x, True)* 4. *Rombus(\$x, True) >>> Impus(\$x, True)*”
 - Seharusnya: ”1. *Zumpus(\$x, True) >>> Transparan(\$x, True)* 2. *Zumpus(\$x, True) >>> Rompus(\$x, True)* 3. *Rompus(\$x, True) >>> Tanah(\$x, True)* 4. *Rompus(\$x, True) >>> Impus(\$x, True)*”
 - Penjelasan: Model salah mengartikan subjek pada aturan ke-3 dan ke-4, sehingga ketika melakukan search resolve tidak dapat menemukan resolusi yang tepat karena subjek dan objek yang digunakan tidak sesuai.
- Tidak melakukan *translation* melainkan seakan-akan melakukan inferensi langsung dalam FOL :

- Premis: "Tumpus adalah merah. Setiap tumpus adalah jompus. Jompus adalah besar"
- Hasil *translation*: "1. *Tumpus(\$x, True) >>> Merah(\$x, True)* 2. *Tumpus(\$x, True) >>> Besar(\$x, True)*"
- Seharusnya: "1. *Tumpus(\$x, True) >>> Merah(\$x, True)* 2. *Tumpus(\$x, True) >>> Jompus(\$x, True)* 3. *Jompus(\$x, True) >>> Besar(\$x, True)*"
- Penjelasan: Model tidak menerjemahkan aturan ke dalam FOL, melainkan melakukan inferensi langsung pada aturan tersebut dengan menghilangkan salah satu aturan yang ada.
- Tidak adanya format keseragaman pada hasil *translasi*:
 - Premis: "Semua Tumpus adalah baik. Setiap tumpus adalah dumpus. Max adalah dumpus. Apakah Max tidak baik?"
 - Hasil *translation*: "Fakta: *Dumpus(\$Max, True)* Konjektur: *Baik(\$Max, False)* Aturan: 1. *Tumpus(\$x, True) → Baik(\$x, True)* 2. *Tumpus(\$x, True) → Dumpus(\$x, True)*"
 - Seharusnya: "Fakta: *Dumpus(\$Max, True)* Aturan: 1. *Tumpus(\$x, True) >>> Baik(\$x, True)* 2. *Tumpus(\$x, True) >>> Dumpus(\$x, True)* Konjektur: *Baik(\$Max, False)*"
 - Penjelasan: Model tidak konsisten dalam menempatkan fakta, aturan, dan konjektur pada hasil *translation* dan juga menggunakan simbol yang berbeda untuk implikasi (\rightarrow dan $>>>$)

Karena permasalahan tersebut, dilakukan perbaikan pada prompt template tersebut untuk meningkatkan kualitas *translation*. Berikut adalah versi kedua dari prompt template yang telah dimodifikasi.

- 1 Deskripsi Tugas: Anda diberikan konteks dan konjektur. Tugasnya adalah:
- 2 1) parsing konteks menjadi aturan logika berdasarkan predikat yang ditentukan
- 3 2) jika konteks berbentuk plural, maka konteks tersebut juga harus diterjemahkan ke bentuk tunggal
- 4 3) menulis fakta yang disebutkan dalam masalah, dan pastikan mereka diterjemahkan dalam bentuk logika juga.
- 5 4) parsing konjektur ke dalam bentuk logika
- 6 5) jangan mencoba menyelesaikan konjektur atau permasalahan, cukup melakukan perintah parsing yang dibutuhkan
- 7 6) berikan penanda ***Bentuk Akhir*** sebagai awalan dan ***Akhir Blok*** sebagai akhiran output dari tugas Anda.
- 8 7) silakan berikan output akhir dalam format berikut. Pastikan Anda menerjemahkan dalam urutan berikut: fakta, aturan, kemudian konjektur.
- 9
- 10 -----

```

11
12 Berikut adalah contohnya:
13 Konteks:
14 Setiap jompus memiliki sifat eksentris. Setiap jompus adalah wumpus. Setiap wumpus
    memiliki sifat tidak transparan. Wumpuses adalah tumpuses. Tumpuses itu nakal.
    Tumpuses adalah vumpuses. Setiap vumpus memiliki sifat dingin. Setiap vumpus adalah
    yumpus. Yumpuses berwarna oranye. Yumpuses adalah Numpuses. Numpuses memiliki sifat
    membosankan. Setiap numpus adalah Dumpus. Setiap dumpus memiliki sifat bukan pemalu.
    Impuses memiliki sifat pemalu. Dumpuses adalah rompuses. Setiap rompus tidaklah
    berwujud cair. Rompuses adalah zumpus. Alex adalah tumpus.
15
16 Konjektur:
17 Alex bukan pemalu.
18
19 ###
20 ***Bentuk Akhir***
21
22 Fakta:
23 Alex adalah tumpus :::: Tumpus(Alex, True)
24
25 Aturan:
26 Jompus($x, True) >>> Eksentris($x, True)
27 Jompus($x, True) >>> Wumpus($x, True)
28 Wumpus($x, True) >>> Transparan($x, False)
29 Wumpus($x, True) >>> Tumpus($x, True)
30 Tumpus($x, True) >>> Nakal($x, True)
31 Tumpus($x, True) >>> Vumpus($x, True)
32 Vumpus($x, True) >>> Dingin($x, True)
33 Vumpus($x, True) >>> Yumpus($x, True)
34 Yumpus($x, True) >>> Oranye($x, True)
35 Yumpus($x, True) >>> Numpus($x, True)
36 Numpus($x, True) >>> Membosankan($x, True)
37 Numpus($x, True) >>> Dumpus($x, True)
38 Dumpus($x, True) >>> Pemalu($x, False)
39 Impus($x, True) >>> Pemalu($x, True)
40 Dumpus($x, True) >>> Rompus($x, True)
41 Rompus($x, True) >>> Cair($x, False)
42 Rompus($x, True) >>> Zumpus($x, False)
43
44 Konjektur:
45 Pemalu(Alex, False)
46
47 ***Akhir Blok***
48
49 -----
50
51 Berikut adalah contohnya:
52 Konteks:
53 Jompuses bukan pemalu. Jompuses adalah yumpuses. Setiap yumpus bersifat agresif. Setiap
    yumpus adalah dumpus. Dumpuses bukan terbuat dari kayu. Dumpuses adalah wumpuses.

```

Wumpuses berwarna merah. Setiap wumpus adalah impus. Setiap impus merupakan benda tidak tembus pandang. Impuses adalah tumpuses. Numpuses asam. Tumpuses objek berasa tidak asam. Tumpuses adalah vumpuses. Vumpuses tidak terbuat tanah. Setiap vumpus adalah zumpus. Zumpuses berukuran kecil. Zumpuses adalah rompuses. Max adalah yumpus.

54

55 Konjektur:

56 Max asam.

57

58 ###

59 ***Bentuk Akhir***

60

61 Fakta:

62 Max adalah yumpus :::: Yumpus(Max, True)

63

64 Aturan:

65 Jompus(\$x, True) >>> Pemalu(\$x, False)

66 Jompus(\$x, True) >>> Yumpus(\$x, True)

67 Yumpus(\$x, True) >>> Agresif(\$x, True)

68 Yumpus(\$x, True) >>> Dumpus(\$x, True)

69 Dumpus(\$x, True) >>> Kayu(\$x, False)

70 Dumpus(\$x, True) >>> Wumpus(\$x, True)

71 Wumpus(\$x, True) >>> Merah(\$x, True)

72 Wumpus(\$x, True) >>> Impus(\$x, True)

73 Impus(\$x, True) >>> TembusPandang(\$x, False)

74 Impus(\$x, True) >>> Tumpus(\$x, True)

75 Numpus(\$x, True) >>> Asam(\$x, True)

76 Tumpus(\$x, True) >>> Asam(\$x, False)

77 Tumpus(\$x, True) >>> Vumpus(\$x, True)

78 Vumpus(\$x, True) >>> Tanah(\$x, False)

79 Vumpus(\$x, True) >>> Zumpus(\$x, True)

80 Zumpus(\$x, True) >>> Kecil(\$x, True)

81 Zumpus(\$x, True) >>> Rompus(\$x, True)

82

83 Konjektur:

84 Asam(Max, True)

85

86 ***Akhir Blok***

87

88 -----

89

90 Berikut adalah contohnya:

91 Konteks:

92 Zumpuses tidak terang. Zumpuses adalah yumpuses. Setiap numpus bersifat garang.

Numpuses adalah dumpus. Dumpuses tidak cerah. Dumpuses adalah wumpuses. Wumpuses marah. Setiap wumpus adalah impus. Setiap impus itu bersifat feisty. Impuses adalah tumpuses. Setiap yumpus berbuah. Tumpuses merupakan objek dengan suhu yang bukan moderat. Tumpuses adalah vumpuses. Vumpuses tidak bahagia. Setiap vumpus adalah Jompus. Jompuses tidak baik hati. Jompuses adalah merah. Timothy adalah yumpus.

93

94 Konjektur:

```

95 Timothy adalah tidak tembus pandang.
96
97 ####
98 ***Bentuk Akhir***
99
100 Fakta:
101 Timothy adalah yumpus ::: Yumpus(Timothy, True)
102
103 Aturan:
104 Zumpus($x, True) >>> Terang($x, False)
105 Zumpus($x, True) >>> Yumpus($x, True)
106 Numpus($x, True) >>> Garang($x, True)
107 Numpus($x, True) >>> Dumpus($x, True)
108 Dumpus($x, True) >>> Cerah($x, False)
109 Dumpus($x, True) >>> Wumpus($x, True)
110 Wumpus($x, True) >>> Marah($x, True)
111 Wumpus($x, True) >>> Impus($x, True)
112 Impus($x, True) >>> Feisty($x, True)
113 Impus($x, True) >>> Tumpus($x, True)
114 Yumpus($x, True) >>> Berbuah($x, True)
115 Tumpus($x, True) >>> Moderat($x, False)
116 Tumpus($x, True) >>> Vumpus($x, True)
117 Vumpus($x, True) >>> Bahagia($x, False)
118 Vumpus($x, True) >>> Jompus($x, True)
119 Jompus($x, True) >>> BaikHati($x, False)
120 Jompus($x, True) >>> Merah($x, True)
121
122 Konjektur:
123 TembusPandang(Timothy, False)
124
125 ***Akhir Blok***
126
127 -----
128
129 Di bawah ini adalah yang perlu Anda terjemahkan:
130 Konteks:
131 [[PREMISES]]
132
133 Konjektur:
134 [[CONJECTURE]]

```

Kode 4.2: Template prompt setelah dimodifikasi dengan tambahan beberapa aturan dan contoh

Walaupun sudah diperbaiki, masih ada beberapa kesalahan yang terjadi pada hasil *translation*, seperti berikut:

- Tidak melakukan *translation* melainkan menyelesaikan langsung dalam bentuk FOL:
 - Premis: "Tumpus adalah merah. Setiap tumpus adalah jompus. Jompus adalah besar"

- Hasil *translation*: "2. *Tumpus(\$x, True) >>> Merah(\$x, True)* 3. *Tumpus(\$x, True) >>> Besar(\$x, True)*"
- Seharusnya: "2. *Wumpus(\$x, True) >>> Merah(\$x, True)* 3. *Wumpus(\$x, True) >>> Rompus(\$x, True)* 4. *Rompus(\$x, True) >>> Besar(\$x, True)*"
- Penjelasan: Model tidak menerjemahkan aturan ke dalam FOL, melainkan langsung menyelesaikan aturan tersebut dengan menghilangkan salah satu aturan yang ada.

Sudah ada perbaikan dalam tahap *translation* dengan penambahan contoh dan deskripsi tugas yang lebih rinci, seperti model mengikuti format dengan benar dan mengurangi kesalahan dalam *translation* subjek dan objek, akan tetapi masih ada kesalahan seperti melakukan silogisme langsung pada tahap *translation*. Hal ini dikarenakan model berparameter rendah tidak memiliki *knowledge* yang cukup dalam mengartikan *term ambigu* yang ada pada aturan logika, yaitu dalam hal ini adalah *term parsing* yang jika diartikan dalam bahasa Indonesia berarti penguraian menurut KBBI, yang padahal arti yang dimaksud dalam konteks ini adalah pemisahan subjek dan objek pada aturan logika.

Oleh karena itu, dilakukan perbaikan lebih lanjut pada prompt template tersebut. Berikut adalah versi ketiga dari prompt template yang telah diperbaiki.

```

1 Deskripsi Tugas: Anda diberikan sebuah konteks dari suatu permasalahan berserta
2 pertanyaan dalam konjektur. Tugas Anda adalah:
3 1) Menerjemahkan konteks ke dalam aturan logika berdasarkan predikat dan objek yang
4 diberikan
5 2) Menulis fakta yang disebutkan dalam masalah
6 3) Terjemahkan fakta yang ditemukan ke dalam bentuk logika
7 4) Terjemahkan konjektur yang diberikan ke dalam bentuk logika juga
8 5) Silahkan berikan output akhir dalam format berikut. Pastikan Anda menerjemahkan
9 dalam urutan berikut: fakta, aturan, kemudian konjektur.

10 -----
11 Berikut adalah contohnya:
12 Konteks:
13 Setiap jompus memiliki sifat tidak eksentris. Sam adalah Jompus.
14 Konjektur:
15 Sam bersifat eksentris.
16
17 ###
18 ***Bentuk Akhir***
19
20 Fakta:
21 Sam adalah jompus :::: Jompus(Sam, True)

```

```

22
23 Aturan:
24 Jompus($x, True) >>> Eksentris($x, False)
25
26 Konjektur:
27 Eksentris(Sam, True)
28
29 ***Akhir Blok***
30 ####
31
32 -----
33
34 Berikut adalah contohnya:
35 Konteks:
36 Wumpuses adalah besar. Wumpuses adalah yumpuses. Setiap yumpus tidak berwarna oranye.
   Alex adalah wumpus.
37
38 Konjektur:
39 Alex oranye
40
41 ###
42 ***Bentuk Akhir***
43
44 Fakta:
45 Alex adalah Wumpus :::: Wumpus(Alex, True)
46
47 Aturan:
48 Wumpus($x, True) >>> Besar($x, True)
49 Wumpus($x, True) >>> Yumpus($x, True)
50 Yumpus($x, True) >>> Oranye($x, False)
51
52 Konjektur:
53 Oranye(Alex, True)
54
55 ***Akhir Blok***
56 ###
57
58 -----
59
60 Berikut adalah contohnya:
61 Konteks:
62 Tumpuses adalah tidak baik. Setiap tumpus adalah jompus. Jompus tidak membosankan.
   Jompuses adalah impus. Impus tidak besar. Impus adalah zumpus. Setiap zumpus adalah
   bahagia. Zumpus adalah wumpus. Setiap dumpus tidak berbahaya. Setiap wumpus adalah
   yumpus. Fae bukan zumpus.
63
64 Konjektur:
65 Fae tidak tembus pandang.
66
67 ###

```

```

68 ***Bentuk Akhir***
69
70 Fakta:
71 Fae bukan zumpus :::: Zumpus(Fae, False)
72
73 Aturan:
74 Tumpus($x, True) >>> Baik($x, False)
75 Tumpus($x, True) >>> Jompus($x, True)
76 Jompus($x, True) >>> Membosankan($x, False)
77 Jompus($x, True) >>> Impus($x, True)
78 Impus($x, True) >>> Besar($x, False)
79 Impus($x, True) >>> Zumpus($x, True)
80 Zumpus($x, True) >>> Bahagia($x, True)
81 Zumpus($x, True) >>> Wumpus($x, True)
82 Dumpus($x, True) >>> Bercahaya($x, False)
83 Wumpus($x, True) >>> Yumpus($x, True)
84
85 Konjektur:
86 TembusPandang(Fae, False)
87
88 ***Akhir Blok***
89 ####
90
91 -----
92
93 Berikut adalah contohnya:
94 Konteks:
95 Setiap impus bukanlah panas. Setiap impus adalah wumpus. Wumpuses adalah temperamen.
    Wumpus adalah numpus. Zumpus adalah tidak bersifat agresif. Numpuses adalah bukan suku
    yang temperate. Setiap numpus adalah dumpus. Dumpuses adalah baik. Setiap dumpus
    adalah rompus. Setiap rompus bukan benda yang tidak tembus cahaya. Setiap rompus
    adalah vumpuses. Vumpuses bukanlah yang mudah dikelola. Vumpuses adalah jompus.
    Jompuses bukanlah pedas. Jompus adalah wumpus. Rex adalah vumpus.
96
97 Konjektur:
98 Rex tembus cahaya.
99
100 ####
101 ***Bentuk Akhir***
102
103 Fakta:
104 Rex adalah vumpus :::: Vumpus(Rex, True)
105
106 Aturan:
107 Impus($x, True) >>> Panas($x, False)
108 Impus($x, True) >>> Wumpus($x, True)
109 Wumpus($x, True) >>> Temperamen($x, True)
110 Wumpus($x, True) >>> Numpus($x, True)
111 Zumpus($x, True) >>> Agresif($x, False)
112 Numpus($x, True) >>> Temperate($x, False)

```

```

113 Numpus($x, True) >>> Dumpus($x, True)
114 Dumpus($x, True) >>> Baik($x, True)
115 Dumpus($x, True) >>> Rompus($x, True)
116 Rompus($x, True) >>> TembusCahaya($x, True)
117 Rompus($x, True) >>> Vumpus($x, True)
118 Vumpus($x, True) >>> MudahDikelola($x, False)
119 Vumpus($x, True) >>> Jompus($x, True)
120 Jompus($x, True) >>> Pedas($x, False)
121 Jompus($x, True) >>> Wumpus($x, True)
122
123 Konjektur:
124 TembusCahaya(Rex, True)
125
126 ***Akhir Blok***
127 ###
128
129 -----
130
131 Berikut adalah contohnya:
132 Konteks:
133 Jompuses bukan pemalu. Jompuses adalah yumpuses. Setiap yumpus bersifat agresif. Setiap
yumpus adalah dumpus. Dumpuses bukan terbuat dari kayu. Dumpuses adalah wumpuses.
Wumpuses berwarna merah. Setiap wumpus adalah impus. Setiap impus tidak tembus
pandang. Impuses adalah tumpuses. Numpuses asam. Tumpuses tidak asam. Tumpuses adalah
vumpuses. Vumpuses tidak terbuat tanah. Setiap vumpus adalah zumpus. Zumpuses
berukuran kecil. Zumpuses adalah rompuses. Max adalah yumpus.
134
135 Konjektur:
136 Max asam.
137
138 ###
139 ***Bentuk Akhir***
140
141 Fakta:
142 Max adalah yumpus :::: Yumpus(Max, True)
143
144 Aturan:
145 Jompus($x, True) >>> Pemalu($x, False)
146 Jompus($x, True) >>> Yumpus($x, True)
147 Yumpus($x, True) >>> Agresif($x, True)
148 Yumpus($x, True) >>> Dumpus($x, True)
149 Dumpus($x, True) >>> Kayu($x, False)
150 Dumpus($x, True) >>> Wumpus($x, True)
151 Wumpus($x, True) >>> Merah($x, True)
152 Wumpus($x, True) >>> Impus($x, True)
153 Impus($x, True) >>> TembusPandang($x, False)
154 Impus($x, True) >>> Tumpus($x, True)
155 Numpus($x, True) >>> Asam($x, True)
156 Tumpus($x, True) >>> Asam($x, False)
157 Tumpus($x, True) >>> Vumpus($x, True)

```

```

158 Vumpus($x, True) >>> Tanah($x, False)
159 Vumpus($x, True) >>> Zumpus($x, True)
160 Zumpus($x, True) >>> Kecil($x, True)
161 Zumpus($x, True) >>> Rompus($x, True)
162
163 Konjektur:
164 Asam(Max, True)
165
166 ***Akhir Blok***
167 ###
168
169 -----
170
171 Di bawah ini adalah yang perlu Anda terjemahkan:
172 Konteks:
173 [[PREMISES]]
174
175 Konjektur:
176 [[CONJECTURE]]
177
178 ###
179 ***Bentuk Akhir***

```

Kode 4.3: Template prompt dengan penggunaan diksi sesuai kamus EBI pada konteks ini

Term parsing diubah menjadi menerjemahkan aturan logika dengan memisahkan subjek dan objek pada setiap aturan, sehingga model lebih memahami tugas yang diberikan. Format seperti ”***Bentuk Akhir***” langsung dimasukkan ke dalam contoh pada template dan juga contoh yang diberikan bervariasi dalam istilah yang digunakan dan banyaknya kalimat dalam premis agar model dapat lebih memahami tugas yang diberikan. Dari hasil evaluasi, prompt template versi ketiga ini sudah cukup baik, dalam artian bahwa model dapat mengikuti format dengan benar dan mengurangi kesalahan dalam *translation* subjek dan objek. Oleh karena itu, prompt template versi ketiga ini digunakan dalam eksperimen pada penelitian ini.

4.3.2 Decomposition to Conjunctive Normal Form

Pada tahap ini dilakukan prompt untuk penerjemahan aturan setelah proses *First Order Logic* ke dalam bentuk *Conjunctive Normal Form* dalam proses *decomposition*.

Berikut adalah prompt template yang digunakan yang merupakan hasil dari translasi langsung dari prompt yang digunakan pada *Aristotle Framework*

¹ Tugas deskripsi:

```

2 Diberikan premis dan konjektur dalam bentuk first-order logic form (FOL), silahkan
memecah pernyataan FOL menggunakan normalisasi dan skolemisasi.
3 ---
4 Aturan:
5
6 Normalisasi: Ubah setiap premis dan konjektur ke Prenex Normal Form (PNF) dan kemudian
ke Conjunctive Normal Form (CNF).
7 - Dalam konversi CNF, negasi nilai boolean langsung daripada menambahkan simbol di
depan. Contohnya, ubah "Jompus($x, True) >>> Pemalu($x, False)" menjadi "- \(\forall x
\left(Jompus(x, False) \lor Pemalu(x, False)\right)", bukan "- \(\forall x
\left(\neg Jompus(x, True) \lor Pemalu(x, False)\right)".
8
9 Skolemisasi: Hilangkan kuantor eksistensial dengan menggunakan fungsi atau konstanta
Skolem.
10
11 **Catatan Penting:** Prompt ini dirancang hanya untuk memandu model bahasa dalam proses
dekomposisi. Jangan mencoba menyelesaikan konjektur, dan pastikan konjektur tetap
seperti yang dinyatakan tanpa negasi.
12 ---
13 Contoh di bawah ini:
14 Konteks:
15
16 Aturan:
17 1. Jompus($x, True) >>> Pemalu($x, False)
18 2. Jompus($x, True) >>> Yumpus($x, True)
19 3. Yumpus($x, True) >>> Agresif($x, True)
20 4. Yumpus($x, True) >>> Dumpus($x, True)
21 5. Dumpus($x, True) >>> Kayu($x, False)
22 6. Dumpus($x, True) >>> Wumpus($x, True)
23 7. Wumpus($x, True) >>> Merah($x, True)
24 8. Wumpus($x, True) >>> Impus($x, True)
25 9. Impus($x, True) >>> TembusPandang($x, x, True)
26 10. Impus($x, True) >>> Tumpus($x, True)
27 11. Numpus($x, True) >>> Asam($x, True)
28 12. Tumpus($x, True) >>> Asam($x, False)
29 13. Tumpus($x, True) >>> Vumpus($x, True)
30 14. Vumpus($x, True) >>> Tanah($x, True)
31 15. Vumpus($x, True) >>> Zumpus($x, True)
32 16. Zumpus($x, True) >>> Kecil($x, True)
33 17. Zumpus($x, True) >>> Rompus($x, True)
34
35 Konjektur:
36 Sour(Max, True)
37
38 Pemecahan:
39
40 ***Bentuk Akhir***
41
42 Aturan dalam CNF:
43 1. - \(\forall x \left(Jompus(x, False) \lor Pemalu(x, False)\right)\)

```

```

44 2. - \(\forall x \ \left(Jompus(x, \text{False}) \ \lor Yumpus(x, \text{True})\right)\)
45 3. - \(\forall x \ \left(Yumpus(x, \text{False}) \ \lor Agresif(x, \text{True})\right)\)
46 4. - \(\forall x \ \left(Yumpus(x, \text{False}) \ \lor Dumpus(x, \text{True})\right)\)
47 5. - \(\forall x \ \left(Dumpus(x, \text{False}) \ \lor Kayu(x, \text{False})\right)\)
48 6. - \(\forall x \ \left(Dumpus(x, \text{False}) \ \lor Wumpus(x, \text{True})\right)\)
49 7. - \(\forall x \ \left(Wumpus(x, \text{False}) \ \lor Merah(x, \text{True})\right)\)
50 8. - \(\forall x \ \left(Wumpus(x, \text{False}) \ \lor Impus(x, \text{True})\right)\)
51 9. - \(\forall x \ \left(Impus(x, \text{False}) \ \lor TembusPandang(x, \text{True})\right)\)
52 10. - \(\forall x \ \left(Impus(x, \text{False}) \ \lor Tumpus(x, \text{True})\right)\)
53 11. - \(\forall x \ \left(Numpus(x, \text{False}) \ \lor Asam(x, \text{True})\right)\)
54 12. - \(\forall x \ \left(Tumpus(x, \text{False}) \ \lor Asam(x, \text{False})\right)\)
55 13. - \(\forall x \ \left(Tumpus(x, \text{False}) \ \lor Vumpus(x, \text{True})\right)\)
56 14. - \(\forall x \ \left(Vumpus(x, \text{False}) \ \lor Tanah(x, \text{True})\right)\)
57 15. - \(\forall x \ \left(Vumpus(x, \text{False}) \ \lor Zumpus(x, \text{True})\right)\)
58 16. - \(\forall x \ \left(Zumpus(x, \text{False}) \ \lor Kecil(x, \text{True})\right)\)
59 17. - \(\forall x \ \left(Zumpus(x, \text{False}) \ \lor Rompus(x, \text{True})\right)\)
60
61 Konjektur:
62 Asam(Max, True)
63
64 ***Akhir Blok***
65 ---
66
67 Di bawah ini adalah yang perlu Anda pecahkan menggunakan normalisasi dan skolemisasi.
68 Silahkan akhiri dekomposisi dalam "Bentuk Akhir" dan pastikan Anda mencakup setiap
   klausa dalam bentuk akhir Anda, walaupun klausa tersebut tidak perlu dipecah lagi.
69 Silahkan patuhi instruksi dengan ketat dan jangan melakukan apa yang tidak
   diinstruksikan (misalnya, jangan negasi konjektur dan fakta).
70 Perhatikan bahwa Anda perlu menegasi suatu istilah hanya ketika mengubah sebuah
   implikasi ke CNF. Jika klausa atau istilah tidak mengandung implikasi (misalnya, Fakta
   atau Konjektur), jangan di-negasikan.
71
72 [[PREMISES]]

```

Kode 4.4: Template prompt hasil translasi langsung dari bahasa Inggris

Ada beberapa masalah yang ditemukan pada hasil *decomposition* menggunakan prompt template tersebut, seperti berikut:

- Skolemisasi: Pada ProntoQA tidak perlu menggunakan skolemisasi karena tidak ada kuantor eksistensial yang perlu dihilangkan agar tidak membingungkan model.
- Halusinasi pada Fakta dan Konjektur: Model sering kali menambahkan fakta atau konjektur yang tidak ada pada premis, sehingga hasil dari proses *decomposition* menjadi tidak sesuai:
 - Input: "Aturan: 1. *Tumpus(\$x, True)* >>> *Baik(\$x, True)* 2. *Tumpus(\$x, True)* >>> *Dumpus(\$x, True)*"

- Hasil *decomposition*: "Fakta: *Dumpus(\$Max, True)* Konjektur: *Baik(\$Max, False)* Aturan dalam CNF: 1. - $\forall x (Tumpus(x, False) \vee Baik(x, True))$ 2. - $\forall x (Tumpus(x, False) \vee Dumpus(x, True))$ "
- Seharusnya: "Aturan dalam CNF: 1. - $\forall x (Tumpus(x, False) \vee Baik(x, True))$ 2. - $\forall x (Tumpus(x, False) \vee Dumpus(x, True))$ "
- Penjelasan: Model menambahkan fakta dan konjektur yang tidak ada pada premis, sehingga hasil dari proses *decomposition* menjadi tidak sesuai.

Berikut adalah versi kedua dari prompt template yang telah diperbaiki. Versi kedua ini mirip seperti template versi ketiga pada tahap 4.3, yaitu dengan menambahkan contoh yang lebih bervariasi dan deskripsi tugas yang lebih rinci karena keberhasilan model dalam memahamami tugas pada tahap *translation* dapat diaplikasikan juga pada tahap *decomposition*.

```

1 Deskripsi Tugas:
2 Anda diberikan konteks berupa aturan dalam formula first-order logic (FOL).
3 Tugas Anda adalah melakukan konversi aturan-aturan tersebut menggunakan ketentuan
   normalisasi berikut ini.
4 Silahkan berikan output akhir dalam format sesuai dengan contoh berikut.
5
6 ---
7 Ketentuan normalisasi:
8 1. Konversikan setiap aturan ke Conjunctive Normal Form (CNF).
9 2. Dalam mengkonversikan ke CNF, negasikan nilai boolean secara langsung, alih-alih
   menambahkan simbol  $\neg$  di depan predikat.
10 Contohnya, konversi dari "Jompus($x, True) >>> Pemalu($x, False)" adalah
    " $\left( Jompus(x, \text{True}) \vee Pemalu(x, \text{False}) \right)$ ", dan bukan " $\neg(Jompus(x, \text{True}) \vee Pemalu(x, \text{False}))$ ".
11 ---
12
13 Contoh di bawah ini:
14 Konteks:
15
16 Aturan:
17 1. Jompus($x, True) >>> Eksentris($x, False)
18
19 Hasil konversi:
20 ###
21 ***Bentuk Akhir***
22
23 Aturan dalam CNF:
24 1.  $\left( Jompus(x, \text{False}) \vee Eksentris(x, \text{False}) \right)$ 
25
26 ***Akhir Blok***
27 ###

```

```

28 ---
29
30 Contoh di bawah ini:
31 Konteks:
32
33 Aturan:
34 1. Wumpus($x, True) >>> Besar($x, True)
35 2. Wumpus($x, True) >>> Yumpus($x, True)
36 3. Yumpus($x, True) >>> Oranye($x, false)
37
38 Hasil konversi:
39 ####
40 ***Bentuk Akhir***
41
42 Aturan dalam CNF:
43 1. - \left(Wumpus(x, False) \lor Besar(x, True)\right)
44 2. - \left(Wumpus(x, False) \lor Yumpus(x, True)\right)
45 3. - \left(Yumpus(x, False) \lor Oranye(x, False)\right)
46
47 ***Akhir Blok***
48 ###
49 ---
50
51 Contoh di bawah ini:
52 Konteks:
53
54 Aturan:
55 1. Tumpus($x, True) >>> Baik($x, False)
56 2. Tumpus($x, True) >>> Jompus($x, True)
57 3. Jompus($x, True) >>> Membosankan($x, False)
58 4. Jompus($x, True) >>> Impus($x, True)
59 5. Impus($x, True) >>> Besar($x, False)
60 6. Impus($x, True) >>> Zumpus($x, True)
61 7. Zumpus($x, True) >>> Bahagia($x, True)
62 8. Zumpus($x, True) >>> Wumpus($x, True)
63 9. Dumpus($x, True) >>> Bercahaya($x, False)
64 10. Wumpus($x, True) >>> Yumpus($x, True)
65
66 Hasil konversi:
67
68 ***Bentuk Akhir***
69 ###
70 Aturan dalam CNF:
71 1. - \left(Tumpus(x, False) \lor Baik(x, False)\right)
72 2. - \left(Tumpus(x, False) \lor Jompus(x, True)\right)
73 3. - \left(Jompus(x, False) \lor Membosankan(x, False)\right)
74 4. - \left(Jompus(x, False) \lor Impus(x, True)\right)
75 5. - \left(Impus(x, False) \lor Besar(x, False)\right)
76 6. - \left(Impus(x, False) \lor Zumpus(x, True)\right)
77 7. - \left(Zumpus(x, False) \lor Bahagia(x, True)\right)

```

```

78 8. - \left(Zumpus(x, False) \lor Wumpus(x, True)\right)
79 9. - \left(Dumpus(x, False) \lor Bercahaya(x, False)\right)
80 10. - \left(Wumpus(x, False) \lor Yumpus(x, True)\right)
81
82 ***Akhir Blok***
83 ###
84 ---
85
86 Contoh di bawah ini:
87 Konteks:
88
89 Aturan:
90 1. Impus($x, True) >>> Panas($x, False)
91 2. Impus($x, True) >>> Wumpus($x, True)
92 3. Wumpus($x, True) >>> Temperamen($x, True)
93 4. Wumpus($x, True) >>> Numpus($x, True)
94 5. Zumpus($x, True) >>> Agresif($x, False)
95 6. Numpus($x, True) >>> Temperate($x, False)
96 7. Numpus($x, True) >>> Dumpus($x, True)
97 8. Dumpus($x, True) >>> Baik($x, True)
98 9. Dumpus($x, True) >>> Rompus($x, True)
99 10. Rompus($x, True) >>> TembusCahaya($x, True)
100 11. Rompus($x, True) >>> Vumpus($x, True)
101 12. Vumpus($x, True) >>> MudahDikelola($x, False)
102 13. Vumpus($x, True) >>> Jompus($x, True)
103 14. Jompus($x, True) >>> Pedas($x, False)
104 15. Jompus($x, True) >>> Wumpus($x, True)
105
106 Hasil konversi:
107 ###
108 ***Bentuk Akhir***
109
110 Aturan dalam CNF:
111 1. - \left(\text{Impus}(x, \text{False}) \lor \text{Panas}(x, \text{False})\right)
112 2. - \left(\text{Impus}(x, \text{False}) \lor \text{Wumpus}(x, \text{True})\right)
113 3. - \left(\text{Wumpus}(x, \text{False}) \lor \text{Temperamen}(x, \text{True})\right)
114 4. - \left(\text{Wumpus}(x, \text{False}) \lor \text{Numpus}(x, \text{True})\right)
115 5. - \left(\text{Zumpus}(x, \text{False}) \lor \text{Agresif}(x, \text{False})\right)
116 6. - \left(\text{Numpus}(x, \text{False}) \lor \text{Temperate}(x, \text{False})\right)
117 7. - \left(\text{Numpus}(x, \text{False}) \lor \text{Dumpus}(x, \text{True})\right)
118 8. - \left(\text{Dumpus}(x, \text{False}) \lor \text{Baik}(x, \text{True})\right)
119 9. - \left(\text{Dumpus}(x, \text{False}) \lor \text{Rompus}(x, \text{True})\right)
120 10. - \left(\text{Rompus}(x, \text{False}) \lor \text{TembusCahaya}(x, \text{True})\right)
121 11. - \left(\text{Rompus}(x, \text{False}) \lor \text{Vumpus}(x, \text{True})\right)
122 12. - \left(\text{Vumpus}(x, \text{False}) \lor \text{MudahDikelola}(x, \text{False})\right)
123 13. - \left(\text{Vumpus}(x, \text{False}) \lor \text{Jompus}(x, \text{True})\right)
124 14. - \left(\text{Jompus}(x, \text{False}) \lor \text{Pedas}(x, \text{False})\right)
125 15. - \left(\text{Jompus}(x, \text{False}) \lor \text{Wumpus}(x, \text{True})\right)
126
127 ***Akhir Blok***

```

```

128  ###
129  ---
130
131 Contoh di bawah ini:
132 Konteks:
133
134 Aturan:
135 1. Jompus($x, True) >>> Pemalu($x, False)
136 2. Jompus($x, True) >>> Yumpus($x, True)
137 3. Yumpus($x, True) >>> Agresif($x, True)
138 4. Yumpus($x, True) >>> Dumpus($x, True)
139 5. Dumpus($x, True) >>> Kayu($x, False)
140 6. Dumpus($x, True) >>> Wumpus($x, True)
141 7. Wumpus($x, True) >>> Merah($x, True)
142 8. Wumpus($x, True) >>> Impus($x, True)
143 9. Impus($x, True) >>> TembusPandang($x, x, True)
144 10. Impus($x, True) >>> Tumpus($x, True)
145 11. Numpus($x, True) >>> Asam($x, True)
146 12. Tumpus($x, True) >>> Asam($x, False)
147 13. Tumpus($x, True) >>> Vumpus($x, True)
148 14. Vumpus($x, True) >>> Tanah($x, True)
149 15. Vumpus($x, True) >>> Zumpus($x, True)
150 16. Zumpus($x, True) >>> Kecil($x, True)
151 17. Zumpus($x, True) >>> Rompus($x, True)
152
153 Hasil konversi:
154 ###
155 ***Bentuk Akhir***
156
157 Aturan dalam CNF:
158 1. - \left(Jompus(x, False) \lor Pemalu(x, False)\right)
159 2. - \left(Jompus(x, False) \lor Yumpus(x, True)\right)
160 3. - \left(Yumpus(x, False) \lor Agresif(x, True)\right)
161 4. - \left(Yumpus(x, False) \lor Dumpus(x, True)\right)
162 5. - \left(Dumpus(x, False) \lor Kayu(x, False)\right)
163 6. - \left(Dumpus(x, False) \lor Wumpus(x, True)\right)
164 7. - \left(Wumpus(x, False) \lor Merah(x, True)\right)
165 8. - \left(Wumpus(x, False) \lor Impus(x, True)\right)
166 9. - \left(Impus(x, False) \lor TembusPandang(x, True)\right)
167 10. - \left(Impus(x, False) \lor Tumpus(x, True)\right)
168 11. - \left(Numpus(x, False) \lor Asam(x, True)\right)
169 12. - \left(Tumpus(x, False) \lor Asam(x, False)\right)
170 13. - \left(Tumpus(x, False) \lor Vumpus(x, True)\right)
171 14. - \left(Vumpus(x, False) \lor Tanah(x, True)\right)
172 15. - \left(Vumpus(x, False) \lor Zumpus(x, True)\right)
173 16. - \left(Zumpus(x, False) \lor Kecil(x, True)\right)
174 17. - \left(Zumpus(x, False) \lor Rompus(x, True)\right)
175
176 ***Akhir Blok***
177 ###

```

```

178 ---
179
180 Di bawah ini adalah yang perlu Anda konversikan menggunakan normalisasi.
181
182 Konteks:
183
184 Aturan:
185 [[PREMISES]]
186
187 Hasil konversi:

```

Kode 4.5: Template prompt dengan tambahan contoh dan diksi sesuai EBI

Prompt template versi kedua ini sudah memperbaiki masalah yang ada pada versi pertama dengan penambahan contoh yang lebih bervariasi dan penghilangan skolemisasi, serta fakta dan konjektur pada deskripsi tugas karena tidak diperlukan dan hanya membingungkan model. Dari hasil evaluasi, prompt template versi kedua ini sudah cukup baik untuk digunakan dalam eksperimen pada penelitian ini.

4.3.3 *Search Resolve*

Pada tahap ini dilakukan prompt untuk mencari resolusi dari aturan setelah proses *decomposition*.

Berikut adalah prompt template yang digunakan yang merupakan hasil dari translasi langsung dari prompt yang digunakan pada *Aristotle Framework*

```

1 ---
2
3 Tugas: Diberikan dua klausa, tugas Anda adalah:
4
5 1. **Cek Istilah yang saling Melengkapi / Bertentangan:**
6      - Tentukan apakah suatu istilah dalam klausa 1 bertentangan dengan istilah dalam
        klausa 2.
7
8 **Istilah Pelengkap/Bertentangan:** Dua istilah bertentangan jika mereka mempunyai
    predikat dan argumen yang sama tetapi berbeda dalam polaritas (salah satu dinegasikan)
    atau nilai boolean (True vs. False).
9
10 **Contoh:**
11 1. \(\neg P(x, \text{False}) \wedge P(x, \text{True})\) bertentangan dengan:
12     - \(\neg(\neg P(x, \text{False}) \wedge P(x, \text{True}))\) (polaritas berlawanan, nilai boolean sama).
13     - \(\neg(\neg P(x, \text{True}) \wedge P(x, \text{False}))\) (polaritas sama, nilai boolean berlawanan).
14
15 2. \(\neg(\neg P(x, \text{False}) \wedge P(x, \text{True}))\) bertentangan dengan:
16     - \(\neg(\neg P(x, \text{True}) \wedge P(x, \text{False}))\) (polaritas sama, nilai boolean berlawanan).

```

```

17 - \(\ P(x, \text{False}) \) (polaritas berlawanan, nilai boolean sama).
18
19 **Jika istilah bertentangan ditemukan:** Selesaikan klausa menggunakan aturan di bawah ini.
20
21 2. **Penyelesaian Klausa:***
22
23 ##### **Aturan Penyelesaian:***
24 **Operator OR ( ):***
25 - Istilah Pelengkap/Bertentangan Ditemukan: \( (A \lor B), \, (\neg A \lor C)
\Rightarrow (B \lor C) \)
26
27 **Operator AND ( ):***
28 - Istilah Pelengkap/Bertentangan Ditemukan dengan Sisa Istilah: \( (A \land B), \,
(\neg B \lor C) \Rightarrow (A \land C) \)
29 - Istilah Pelengkap/Bertentangan Ditemukan dengan Sisa Istilah: \neg B \land C \lor D,
\, \((B) \Rightarrow \text{False} \land C \lor D \Rightarrow \text{False} \lor D \Rightarrow \text{True} \)
\text{D} \)
30 - Istilah Pelengkap/Bertentangan Ditemukan yang Mengarah pada Kontradiksi: \( (A \land B), \,
\, \neg B \Rightarrow \text{False} \land A \Rightarrow \text{Kontradiksi} \) (Anda hanya dapat menyimpulkan kontradiksi ketika semua istilah dalam kedua klausa terhubung oleh operator **\land**).
31
32 3. **Pemeriksaan Cukup Setelah Inferensi:***
33 - **Jika kontradiksi ditemukan:***
34 - Simpulkan dengan "Label Cukup: [True]" dan "Klausa Baru: {Kontradiksi}".
35 - Jika tidak ditemukan kontradiksi:
36 - Simpulkan dengan "Label Cukup: [False]".
37
38 Hasil keluaran: Simpulkan label cukup dan klausa baru dalam format:
39
40 ***Bentuk Akhir***
41 Klausa Baru: {new_clause}
42 Label Cukup: [True/False]
43 ***Akhir Blok***
44
45 ---
46
47 Contoh:
48
49 Klausa 1: Tanah(Stella, True)
50 Klausa 2: Numpus(x, False) \lor Tanah(x, False)
51
52 Mari kita uraikan proses penyelesaian untuk klausa yang diberikan:
53
54 ##### **Langkah 1: Cek Istilah Pelengkap/Bertentangan**
55
56 **Klausa yang Diberikan:***
57 - **Klausa 1:** \(\ \neg (\text{Tanah}(Stella, \text{True})) \)
58 - **Klausa 2:** \(\ \neg (\text{Numpus}(x, \text{False})) \lor \neg (\text{Tanah}(x, \text{False})) \)

```

```

59
60 **Identifikasi Potensial Istilah Pelengkap :**
61 - \(\ Tanah(Stella, True) \) dari Klausula 1 berpotensi bertentangan dengan \(\ Tanah(x,
   False) \) dalam Klausula 2. Instansiasi dari \(\ x \) ke \(\ Stella \) diperlukan untuk
   memeriksa kontradiksi.
62
63 **Instansiasikan Klausula 2 dengan \(\ x = Stella \):**
64 - Klausula 2 Asli: \(\ Numpus(x, False) \lor Tanah(x, False) \)
65 - Klausula 2 yang Diinstansiasikan: \(\ Numpus(Stella, False) \lor Tanah(Stella, False) \)
66
67 **Identifikasi Istilah Pelengkap Setelah Instansiasi:**
68 - \(\ Tanah(Stella, True) \) dalam Klausula 1 adalah pelengkap dari \(\ Tanah(Stella,
   False) \) dalam Klausula 2 yang diinstansiasikan (predikat sama, argumen sama, nilai
   boolean berlawanan).
69
70 ### **Langkah 2: Selesaikan Klausula**
71
72 - **Proses Penyelesaian:** 
73   - **Aturan yang digunakan:** **Operator OR ( ) - Istilah Pelengkap/Bertentangan
      Ditemukan** karena istilah pelengkap/bertentangan dalam kedua klausula 1 dan klausula 2
      yang diinstansiasikan terhubung oleh "OR" dengan istilah lain.
74   \[
75     (A \lor B), \ , (\neg A \lor C) \quad \Rightarrow \quad (B \lor C)
76   \]
77 - **Substitusikan istilah:** 
78   \[
79     (\text{Tanah}(Stella, \text{True})), \ , (\text{Numpus}(Stella, \text{False}) \lor \text{Tanah}(Stella, \text{False})) \quad \Rightarrow
80     \quad \quad \quad \text{Numpus}(Stella, \text{False})
81   - **Klausula Hasil:** \(\ Numpus(Stella, False) \)
82
83 ### **Langkah 3: Pemeriksaan Cukup**
84
85 - **Cek Kontradiksi:** 
86   - Klausula hasil \(\ Numpus(Stella, False) \) tidak menghasilkan kontradiksi. Penyidikan
      lebih lanjut diperlukan untuk menentukan kesimpulan akhir, jadi label cukup harus
      False.
87
88 ***Bentuk Akhir***
89 **Klausula Baru:** { \(\ Numpus(Stella, False) \) }
90 **Label Cukup:** [False]
91 ***Akhir Blok***
92
93 ---
94
95 Contoh:
96
97 Klausula 1: Vumpus(Wren, True)
98 Klausula 2: Vumpus(Wren, False)
99

```

```

100 Langkah 1: Cek Istilah Pelengkap/Bertentangan
101
102 **Klausa yang Diberikan:**
103 - **Klausa 1:** \(\text{Vumpus}(\text{Wren}, \text{True})\)
104 - **Klausa 2:** \(\text{Vumpus}(\text{Wren}, \text{False})\)
105
106 **Identifikasi Istilah Pelengkap/Bertentangan:**
107 - \(\text{Vumpus}(\text{Wren}, \text{True})\) dalam Klausa 1 secara langsung bertentangan dengan \
    \(\text{Vumpus}(\text{Wren}, \text{False})\) dalam Klausa 2 (predikat dan argumen sama, tetapi nilai boolean
    berlawanan).
108
109 ### **Langkah 2: Selesaikan Klausa**
110
111 - **Proses Penyelesaian:**
112   - **Aturan Digunakan:** **Kontradiksi Ditemukan** karena kedua istilah bertentangan
     dan tidak ada istilah lain yang hadir.
113   - \(\text{Vumpus}(\text{Wren}, \text{True})\) bertentangan dengan \(\text{Vumpus}(\text{Wren}, \text{False})\),
     menyebabkan kontradiksi langsung.
114
115   - **Klausa Hasil:** Kontradiksi.
116
117 ### **Langkah 3: Pemeriksaan Cukup**
118
119 - **Cek Kontradiksi:**
120   - Penyelesaian menghasilkan kontradiksi, sehingga cukup tercapai.
121
122 ***Bentuk Akhir***
123 **Klausa Baru:** {Kontradiksi}
124 **Label Cukup:** [True]
125 ***Akhir Blok***
126
127 ---
128
129 Contoh:
130
131 Klausa 1:  $\text{Kecil}(\text{Alex}, \text{False})$ 
132 Klausa 2:  $\forall x (\text{Vumpus}(x, \text{False}) \lor \text{Numpus}(x, \text{True}))$ 
133
134 Langkah 1: Cek Istilah Pelengkap/Bertentangan
135
136 **Klausa yang Diberikan:**
137 - **Klausa 1:** \(\text{Kecil}(\text{Alex}, \text{False})\)
138 - **Klausa 2:** \(\forall x (\text{Vumpus}(x, \text{False}) \lor \text{Numpus}(x, \text{True}))\)
139
140 **Identifikasi Potensial Istilah Pelengkap:**
141 - Tidak ada kecocokan predikat antara \(\text{Kecil}(\text{Alex}, \text{False})\) dalam Klausa 1 dan
     istilah dalam Klausa 2. Klausa 2 melibatkan predikat \(\text{Vumpus}\) dan \(\text{Numpus}\),
     sedangkan Klausa 1 melibatkan \(\text{Kecil}\). Oleh karena itu, tidak ada istilah
     pelengkap atau bertentangan yang ditemukan.
142

```

```

143 ### **Langkah 2: Selesaikan Klausa**
144
145 - **Proses Penyelesaian:**
146   - **Tidak Ada Istilah Pelengkap atau Bertentangan Ditemukan:** Tidak ada istilah
     dalam Klausa 1 yang bertentangan atau melengkapi istilah dalam Klausa 2 karena
     predikat yang berbeda. Ini menghasilkan tidak ada perubahan melalui penyelesaian
     karena tidak ada dasar untuk menggabungkan atau memodifikasi klausa.

147
148 ### **Langkah 3: Pemeriksaan Cukup**
149
150 - **Cek Kontradiksi:**
151   - Karena tidak ada istilah yang pelengkap atau bertentangan dan tidak ada
     penyelesaian yang dapat diterapkan, tidak ada kontradiksi. Penyidikan lebih lanjut
     diperlukan untuk mengevaluasi nilai kebenaran klausa, jadi label cukup harus False.

152
153 ***Bentuk Akhir***
154 **Label Cukup:** [False]
155 ***Akhir Blok***
156
157 ---
158
159 Silakan simpulkan informasi tersebut dengan format berikut:
160
161 ***Bentuk Akhir***
162 Klausa Baru: {placeholder_for_new_clause}
163 Label Cukup: [placeholder_for_sufficiency_label]
164 ***Bentuk Akhir***
165
166 ---
167
168 **Dibawah ini tugas yang perlu Anda lakukan:**
169
170 Klausa 1:
171 [[SOS]]
172 Klausa 2:
173 [[SELECTED-CLAUSE]]

```

Kode 4.6: Template prompt hasil translasi langsung dari bahasa Inggris

Dari hasil evaluasi ditemukan pola kesalahan yang sering terjadi, seperti berikut:

- Model tidak mengenali istilah pelengkap/bertentangan:

- **Input:**

Klausa 1:

Jompus(Max, True)

Klausa 2:

$\forall x (Jompus(x, \text{False}) \vee Membosankan(x, \text{True}))$

- Output:

Dibawah ini tugas yang perlu Anda lakukan:

Klausa 1:

Jompus(Max, True)

Klausa 2:

$\forall x (Jompus(x, False) \vee Membosankan(x, True))$

Langkah 1: Cek Istilah Pelengkap/Bertentangan

Klausa yang Diberikan:

- **Klausa 1:** *Jompus(Max, True)*
- **Klausa 2:** $\forall x (Jompus(x, False) \vee Membosankan(x, True))$

Identifikasi Potensial Istilah Pelengkap:

- Tidak ada kecocokan predikat antara *Jompus(Max, True)* dalam Klausa 1 dan istilah dalam Klausa 2. Klausa 2 melibatkan predikat *Jompus* dan *Membosankan*, sedangkan Klausa 1 melibatkan *Jompus*. Oleh karena itu, tidak ada istilah pelengkap atau bertentangan yang ditemukan.

Langkah 2: Selesaikan Klausa

- **Proses Penyelesaian:**

- **Tidak Ada Istilah Pelengkap atau Bertentangan Ditemukan:** Tidak ada istilah dalam Klausa 1 yang bertentangan atau melengkapi istilah dalam Klausa 2 karena predikat yang berbeda. Ini menghasilkan tidak ada perubahan melalui penyelesaian karena tidak ada dasar untuk menggabungkan atau memodifikasi klausa.

Langkah 3: Pemeriksaan Cukup

- **Cek Kontradiksi:**

- Karena tidak ada istilah yang pelengkap atau bertentangan dan tidak ada penyelesa-

ian yang dapat diterapkan, tidak ada kontradiksi. Penyidikan lebih lanjut diperlukan untuk mengevaluasi nilai kebenaran klausa, jadi label cukup harus False.

Bentuk Akhir

Label Cukup: [False]

Akhir Blok

- **Penjelasan:** Model gagal menemukan resolusi karena tidak mengenali bahwa $Jompus(Max, True)$ dan $Jompus(x, False)$ adalah istilah yang bertentangan ketika x disubstitusi dengan Max .
- Model menggunakan \wedge (AND) dan penambahan *term* atau klausa yang tidak ada pada input:

- Input:

Klausa 1:

$Zumpus(Wren, False) \vee Transparan(Wren, True)$

Klausa 2:

8. $Jompus(x, False) \vee Zumpus(x, True)$

- Output:

Dibawah ini tugas yang perlu Anda lakukan:

Klausa 1:

$Zumpus(Wren, False) \vee Transparan(Wren, True)$

Klausa 2:

8. $Jompus(x, False) \vee Zumpus(x, True) \wedge Jompus(x, False) \vee Transparan(x, True)$

Langkah 1: Cek Istilah Pelengkap/Bertentangan

Klausa yang Diberikan:

- **Klausa 1:** $Zumpus(Wren, False) \vee Transparan(Wren, True)$
- **Klausa 2:** $Jompus(x, False) \vee Zumpus(x, True) \wedge Jompus(x, False) \vee Transparan(x, True)$

Identifikasi Potensial Istilah Pelengkap:

- $Zumpus(Wren, False)$ dalam Klausula 1 berpotensi bertentangan dengan $Zumpus(x, True)$ dalam Klausula 2. Instansiasikan x ke $Wren$ untuk memeriksa kontradiksi.

Instansiasikan Klausula 2 dengan $x = Wren$:

- Klausula 2 Asli: $Jompus(x, False) \vee Zumpus(x, True) \wedge Jompus(x, False) \vee Transparan(x, True)$
- Klausula 2 yang Diinstansiasikan: $Jompus(Wren, False) \vee Zumpus(Wren, True) \wedge Jompus(Wren, False) \vee Transparan(Wren, True)$

Identifikasi Istilah Pelengkap Setelah Instansiasi:

- $Zumpus(Wren, False)$ dalam Klausula 1 adalah pelengkap dari $Zumpus(Wren, True)$ dalam Klausula 2 yang diinstansiasikan (predikat sama, argumen sama, nilai boolean berlawanan).

Langkah 2: Selesaikan Klausula

- **Proses Penyelesaian:**

- **Aturan yang digunakan:** **Operator AND () - Istilah Pelengkap/Bertentangan Ditemukan dengan Sisa Istilah** karena istilah pelengkap/bertentangan dalam kedua klausula 1 dan klausula 2 yang diinstansiasikan terhubung oleh "AND" dengan istilah lain.

$$A \wedge B, \neg B \vee C \Rightarrow A \wedge C$$

- **Substitusikan istilah:**

$$\begin{aligned} & Zumpus(Wren, False) \vee Transparan(Wren, True), Jompus(Wren, False) \\ & \vee Zumpus(Wren, True) \wedge Jompus(Wren, False) \vee Transparan(Wren, True) \\ \Rightarrow & \quad Transparan(Wren, True) \vee Jompus(Wren, False) \end{aligned}$$

- **Klausa Hasil:** $\text{Transparan}(\text{Wren}, \text{True}) \vee \text{Jompus}(\text{Wren}, \text{False})$

Langkah 3: Pemeriksaan Cukup

- **Cek Kontradiksi:**

- Klausa hasil $\text{Transparan}(\text{Wren}, \text{True}) \vee \text{Jompus}(\text{Wren}, \text{False})$ tidak menghasilkan kontradiksi. Penyidikan lebih lanjut diperlukan untuk menentukan kesimpulan akhir, jadi label cukup harus False.

Bentuk Akhir

Klausa Baru: $\text{Transparan}(\text{Wren}, \text{True}) \vee \text{Jompus}(\text{Wren}, \text{False})$

Label Cukup: [False]

Akhir Blok:

- Penjelasan:

Model salah menggunakan operator \wedge pada Klausa 2, padahal seharusnya menggunakan \vee . Hal ini menyebabkan proses resolusi menjadi salah karena klausa yang digunakan tidak sesuai dengan input yang diberikan.

Oleh karena itu, dilakukan perbaikan pada prompt template tersebut dengan menghilangkan deskripsi tugas seperti penghilangan operator \wedge (AND) pada aturan penyelesaian karena operator tersebut tidak diperlukan dalam proses resolusi pada dataset ProntoQA, serta penambahan sebuah contoh resolusi yang saling berkomplemen untuk membantu model dalam mengenali istilah pelengkap/bertentangan. Berikut adalah versi kedua dari prompt template yang telah dimodifikasi.

```

1 ---
2
3 Tugas: Diberikan dua klausa, tugas Anda adalah:
4
5 1. **Cek Istilah yang saling Melengkapi / Bertentangan:** 
6     - Tentukan apakah suatu istilah dalam klausa 1 bertentangan dengan istilah dalam
       klausa 2.
7
8 **Istilah Pelengkap/Bertentangan:** Dua istilah bertentangan jika mereka mempunyai
   predikat dan argumen yang sama tetapi berbeda dalam polaritas (salah satu dinegasi)
   atau nilai boolean didalamnya (True vs. False).
9
10 **Contoh:**
```

```

11 1.  $\neg P(x, \text{False})$  bertentangan dengan:
12   -  $\neg(\neg P(x, \text{False}))$  (polaritas berlawanan, nilai boolean sama).
13   -  $\neg(P(x, \text{True}))$  (polaritas sama, nilai boolean berlawanan).
14
15 2.  $\neg(\neg P(x, \text{False}))$  bertentangan dengan:
16   -  $\neg(\neg P(x, \text{True}))$  (polaritas sama, nilai boolean berlawanan).
17   -  $\neg(P(x, \text{False}))$  (polaritas berlawanan, nilai boolean sama).
18
19 **Jika istilah bertentangan ditemukan:** Selesaikan klausa tersebut menggunakan aturan
di bawah ini.
20
21 2. **Penyelesaian Klausa:***
22
23 ### **Aturan Penyelesaian:***
24 **Operator OR ( ) atau (\lor):**
25 - Istilah Pelengkap/Bertentangan Ditemukan:  $\neg(A \lor B), \neg(\neg A \lor C)$ 
 $\Rightarrow (B \lor C)$ 
26
27 3. **Pemeriksaan Cukup Setelah Inferensi:***
28 - **Jika kontradiksi ditemukan:***
29   - Simpulkan dengan "Label Cukup: [True]" dan "Klausa Baru: {Kontradiksi}".
30 - Jika tidak ditemukan kontradiksi:
31   - Simpulkan dengan "Label Cukup: [False]".
32
33 Hasil keluaran: Simpulkan label cukup dan klausa baru dalam format:
34
35 ***Bentuk Akhir***
36 Klausa Baru: {new_clause}
37 Label Cukup: [True/False]
38 ***Akhir Blok***
39
40 ---
41
42 Contoh:
43
44 Klausa 1: Tanah(Stella, True)
45 Klausa 2: Numpus(x, False) \lor Tanah(x, False)
46
47 Mari kita uraikan proses penyelesaian untuk klausa yang diberikan:
48
49 ### **Langkah 1: Cek Istilah Pelengkap/Bertentangan**
50
51 **Klausa yang Diberikan:***
52 - **Klausa 1:**  $\neg(\neg \text{Tanah}(Stella, \text{True}))$ 
53 - **Klausa 2:**  $\neg(\neg \text{Numpus}(x, \text{False})) \lor \neg(\neg \text{Tanah}(x, \text{False}))$ 
54
55 **Identifikasi Potensial Istilah Pelengkap :***
56 -  $\neg(\neg \text{Tanah}(Stella, \text{True}))$  dari Klausa 1 berpotensi bertentangan dengan  $\neg(\neg \text{Tanah}(x, \text{False}))$  dalam Klausa 2. Instansiasi dari  $\neg(x)$  ke  $\neg(Stella)$  diperlukan untuk memeriksa kontradiksi.

```

```

57
58 **Instansiasiakan Klausa 2 dengan \(\( x = Stella \)\):**
59 - Klausa 2 Asli: \(\( \text{Numpus}(x, \text{False}) \vee \text{Tanah}(x, \text{False}) \) \)
60 - Klausa 2 yang Diinstansiasiakan: \(\( \text{Numpus}(\text{Stella}, \text{False}) \vee \text{Tanah}(\text{Stella}, \text{False}) \) \)
61
62 **Identifikasi Istilah Pelengkap Setelah Instansiasi:** 
63 - \(\( \text{Tanah}(\text{Stella}, \text{True}) \) \) dalam Klausa 1 adalah pelengkap dari \(\( \text{Tanah}(\text{Stella}, \text{False}) \) \) dalam Klausa 2 yang diinstansiasiakan (predikat sama, argumen sama, nilai boolean berlawanan).
64
65 ##### **Langkah 2: Selesaikan Klausa**
66
67 - **Proses Penyelesaian:** 
68 - **Aturan yang digunakan:** **Operator OR ( )** - Istilah Pelengkap/Bertentangan Ditemukan** karena istilah pelengkap/bertentangan dalam kedua klausa 1 dan klausa 2 yang diinstansiasiakan terhubung oleh "OR" dengan istilah lain.
69     \[
70     (\text{A} \vee \text{B}), \backslash, (\neg \text{A} \vee \text{C}) \quad \Rightarrow \quad (\text{B} \vee \text{C})
71     \]
72 - **Substitusikan istilah:** 
73     \[
74     (\text{Tanah}(\text{Stella}, \text{True})), \backslash, (\text{Tanah}(\text{Stella}, \text{False}) \vee \text{Numpus}(\text{Stella}, \text{False})) \quad \Rightarrow
75     \]
76 - **Klausa Hasil:** \(\( \text{Numpus}(\text{Stella}, \text{False}) \) \)
77
78 ##### **Langkah 3: Pemeriksaan Cukup**
79
80 - **Cek Kontradiksi:** 
81 - Klausa hasil \(\( \text{Numpus}(\text{Stella}, \text{False}) \) \) tidak menghasilkan kontradiksi. Penyidikan lebih lanjut diperlukan untuk menentukan kesimpulan akhir, jadi label cukup harus False.
82
83 ***Bentuk Akhir***
84 Klausa Baru: { \(\( \text{Numpus}(\text{Stella}, \text{False}) \) \) }
85 Label Cukup: [False]
86 ***Akhir Blok***
87
88 ---
89
90 Contoh:
91
92 Klausa 1: Vumpus(Wren, True)
93 Klausa 2: Vumpus(Wren, False)
94
95 Mari kita uraikan proses penyelesaian untuk klausa yang diberikan:
96
97 ##### **Langkah 1: Cek Istilah Pelengkap/Bertentangan**
98
99 **Klausa yang Diberikan:**
```

```

100 - **Klausa 1:** \(\ Vampus(Wren, True) \)
101 - **Klausa 2:** \(\ Vampus(Wren, False) \)
102
103 **Identifikasi Istilah Pelengkap/Bertentangan:**
104 - \(\ Vampus(Wren, True) \) dalam Klausa 1 secara langsung bertentangan dengan \(
    Vampus(Wren, False) \) dalam Klausa 2 (predikat dan argumen sama, tetapi nilai boolean
    berlawanan).
105
106 ### **Langkah 2: Selesaikan Klausa**
107
108 - **Proses Penyelesaian:**
109     - **Aturan Digunakan:** **Kontradiksi Ditemukan** karena kedua istilah bertentangan
        dan tidak ada istilah lain yang hadir.
110     - \(\ Vampus(Wren, True) \) bertentangan dengan \(\ Vampus(Wren, False) \),
        menyebabkan kontradiksi langsung.
111
112     - **Klausa Hasil:** Kontradiksi.
113
114 ### **Langkah 3: Pemeriksaan Cukup**
115
116 - **Cek Kontradiksi:**
117     - Penyelesaian menghasilkan kontradiksi, sehingga cukup tercapai.
118
119 ***Bentuk Akhir***
120 Klausa Baru: {Kontradiksi}
121 Label Cukup: [True]
122 ***Akhir Blok***
123
124 ---
125
126 Contoh:
127
128 Klausa 1: \(\ Vampus(Wren, True) \)
129 Klausa 2: Vampus(Wren, False)
130
131 Mari kita uraikan proses penyelesaian untuk klausa yang diberikan:
132
133 ### **Langkah 1: Cek Istilah Pelengkap/Bertentangan**
134
135 **Klausa yang Diberikan:**
136 - **Klausa 1:** \(\ Vampus(Wren, True) \)
137 - **Klausa 2:** \(\ Vampus(Wren, False) \)
138
139 **Identifikasi Istilah Pelengkap/Bertentangan:**
140 - \(\ Vampus(Wren, True) \) dalam Klausa 1 secara langsung bertentangan dengan \(
    Vampus(Wren, False) \) dalam Klausa 2 (predikat dan argumen sama, tetapi nilai boolean
    berlawanan).
141
142 ### **Langkah 2: Selesaikan Klausa**
143

```

```

144 - **Proses Penyelesaian:**  

145   - **Aturan Digunakan:** **Kontradiksi Ditemukan** karena kedua istilah bertentangan  

      dan tidak ada istilah lain yang hadir.  

146   -  $\neg(\text{Vumpus}(\text{Wren}, \text{True}))$  bertentangan dengan  $\neg(\text{Vumpus}(\text{Wren}, \text{False}))$ ,  

      menyebabkan kontradiksi langsung.  

147  

148   - **Klausa Hasil:** Kontradiksi.  

149  

150 ### **Langkah 3: Pemeriksaan Cukup**  

151  

152 - **Cek Kontradiksi:**  

153   - Penyelesaian menghasilkan kontradiksi, sehingga cukup tercapai.  

154  

155 ***Bentuk Akhir***  

156 Klausa Baru: {Kontradiksi}  

157 Label Cukup: [True]  

158 ***Akhir Blok***  

159  

160 ---  

161  

162 Contoh:  

163  

164 Klausa 1: Kecil(Alex, False)  

165 Klausa 2:  $\neg(\neg(\text{Vumpus}(x, \text{False}) \lor \text{Numpus}(x, \text{True})))$   

166  

167 Mari kita uraikan proses penyelesaian untuk klausa yang diberikan:  

168  

169 ### **Langkah 1: Cek Istilah Pelengkap/Bertentangan**  

170  

171 **Klausa yang Diberikan:**  

172 - **Klausa 1:**  $\neg(\text{Kecil}(\text{Alex}, \text{False}))$   

173 - **Klausa 2:**  $\neg(\neg(\text{Vumpus}(x, \text{False}) \lor \text{Numpus}(x, \text{True})))$   

174  

175 **Identifikasi Potensial Istilah Pelengkap:**  

176 - Tidak ada kecocokan predikat antara  $\neg(\text{Kecil}(\text{Alex}, \text{False}))$  dalam Klausa 1 dan  

      istilah dalam Klausa 2. Klausa 2 melibatkan predikat  $\neg(\text{Vumpus} \cdot)$  dan  $\neg(\text{Numpus} \cdot)$ ,  

      sedangkan Klausa 1 melibatkan  $\neg(\text{Kecil} \cdot)$ . Oleh karena itu, tidak ada istilah  

      pelengkap atau bertentangan yang ditemukan.  

177  

178 ### **Langkah 2: Selesaikan Klausa**  

179  

180 - **Proses Penyelesaian:**  

181   - **Tidak Ada Istilah Pelengkap atau Bertentangan Ditemukan:** Tidak ada istilah  

      dalam Klausa 1 yang bertentangan atau melengkapi istilah dalam Klausa 2 karena  

      predikat yang berbeda. Ini menghasilkan tidak ada perubahan melalui penyelesaian  

      karena tidak ada dasar untuk menggabungkan atau memodifikasi klausa.  

182  

183 ### **Langkah 3: Pemeriksaan Cukup**  

184  

185 - **Cek Kontradiksi:**
```

```

186 - Karena tidak ada istilah yang pelengkap atau bertentangan dan tidak ada
      penyelesaian yang dapat diterapkan, tidak ada kontradiksi. Penyidikan lebih lanjut
      diperlukan untuk mengevaluasi nilai kebenaran klausa, jadi label cukup harus False.
187
188 ***Bentuk Akhir***
189 Label Cukup: [False]
190 ***Akhir Blok***
191
192 ---
193
194 Silakan simpulkan informasi tersebut dengan format berikut. Ubah placeholder sesuai
      dengan kesimpulan:
195
196 ***Bentuk Akhir***
197 Klausa Baru: {placeholder_for_new_clause}
198 Label Cukup: [placeholder_for_sufficiency_label]
199 ***Akhir Blok***
200
201 ---
202
203 **Dibawah ini tugas yang perlu Anda lakukan:** 
204
205 Klausa 1:
206 [[SOS]]
207 Klausa 2:
208 [[SELECTED-CLAUSE]]
209
210 Mari kita uraikan proses penyelesaian untuk klausa yang diberikan:
```

Kode 4.7: Template prompt dengan penghilangan perintah operator AND dan tambahan contoh

Hasil dari *prompt template* versi kedua ini sudah memperbaiki beberapa masalah yang ada pada versi pertama, seperti model dapat mengenali istilah pelengkap/bertentangan dengan baik dan tidak menggunakan operator \wedge (AND) yang tidak perlu digunakan.

Maka deskripsi tugas yang menjelaskan tentang penggunaan \neg dihilangkan agar tidak membingungkan model. Berikut adalah versi ketiga dari *prompt template* yang telah diperbaiki.

```

1 ---
2
3 Tugas: Diberikan dua clause, tugas Anda adalah:
4
5 1. **Cek literal dalam clause yang saling komplemen:** 
6      - Tentukan apakah suatu literal dalam clause 1 merupakan komplemen dengan literal
        dalam clause 2.
7
8 **Literal yang saling komplemen:** Dua literal saling komplemen jika mereka mempunyai
```

```

predikat dan argumen yang sama tetapi berbeda nilai boolean di dalamnya (True vs.
False).

9
10 **Contoh:**
11 1. \(\neg P(x, False) \vee \neg P(x, True)\) saling komplemen dengan \(\neg P(x, True) \vee \neg P(x, False)\).
12
13 2. \(\neg P(x, True) \vee \neg P(x, False)\) saling komplemen dengan \(\neg P(x, False) \vee \neg P(x, True)\).
14
15 **Jika literal yang saling komplemen ditemukan:** terapkan resolusi menggunakan aturan
di bawah ini.
16
17 2. **Resolusi:**
18
19 ### **Aturan Resolusi:**
20 - Literal yang saling komplemen ditemukan: \((P(x, True) \vee P(x, False)) \vee (P(x, False) \vee P(x, True)) \Rightarrow (P(x, True) \vee P(x, False))\)
21
22 3. **Pemeriksaan Setelah Resolusi:**
23 - **Jika kontradiksi ditemukan:**
24   - Simpulkan dengan "Label Cukup: [True]" dan "Clause Baru: {Kontradiksi}".
25 - Jika tidak ditemukan kontradiksi:
26   - Simpulkan dengan "Label Cukup: [False]".
27
28 Hasil keluaran: Simpulkan label cukup dan clause baru dalam format:
29
30 ***Bentuk Akhir***
31 Clause Baru: {new_clause}
32 Label Cukup: [True/False]
33 ***Akhir Blok***
34
35 ---
36
37 Contoh:
38
39 Clause 1: Tanah(Stella, True)
40 Clause 2: Numpus(x, False) \vee Tanah(x, False)
41
42 Mari kita lakukan proses resolusi untuk kedua clause yang diberikan:
43
44 ### **Langkah 1: Cek Literal yang Saling Komplemen**
45
46 **Clause yang Diberikan:**
47 - **Clause 1:** \(\neg Tanah(Stella, True) \vee \neg Tanah(x, False)\)
48 - **Clause 2:** \(\neg Numpus(x, False) \vee \neg Tanah(x, False)\)
49
50 **Identifikasi Literal yang Saling Komplemen :**
51 - \(\neg Tanah(Stella, True) \vee \neg Tanah(x, False)\) pada Clause 1 berpotensi saling komplemen dengan \(\neg Numpus(x, False) \vee \neg Tanah(x, False)\) pada Clause 2. Instansiasi dari \(\neg x \vee \neg y\) ke \(\neg z \vee \neg w\) diperlukan
untuk memastikan bahwa kedua literal saling komplemen.

```

```

52
53 **Instansiasikan Clause 2 dengan \(\( x = Stella \)\):**
54 - Clause 2 Asli: \(\( \text{Numpus}(x, \text{False}) \vee \text{Tanah}(x, \text{False}) \) \)
55 - Clause 2 setelah Diinstansiasi: \(\( \text{Numpus}(\text{Stella}, \text{False}) \vee \text{Tanah}(\text{Stella}, \text{False}) \) \)
56
57 **Identifikasi Literal yang Saling Komplemen Setelah Instansiasi:** 
58 - \(\( \text{Tanah}(\text{Stella}, \text{True}) \) \) dalam Clause 1 adalah komplemen dari \(\( \text{Tanah}(\text{Stella}, \text{False}) \) \) dalam Clause 2 setelah diinstansiasi (predikat sama, argumen sama, nilai boolean berlawanan).
59
60 ##### **Langkah 2: Terapkan Resolusi pada Kedua Clause**
61
62 - **Proses Resolusi:** 
63 - **Aturan Resolusi:** terdapat literal yang saling komplemen ditemukan pada Clause 1 dan Clause 2 setelah diinstansiasi yang terhubung oleh "OR" dengan literal lainnya, maka formula resolusi di bawah ini dapat dilakukan.
64     \[
65     \(\( (\text{P}(x, \text{True}) \vee \text{B}), \backslash, (\text{P}(x, \text{False}) \vee \text{C}) \Rightarrow (\text{B} \vee \text{C}) \) \)
66     \]
67 - **Instansiasi Literal:** Instansiasikan literal dari Clause 1 dan Clause 2 kedalam formula yang sudah diberikan
68     \[
69     (\text{Tanah}(\text{Stella}, \text{True})), \backslash, (\text{Tanah}(\text{Stella}, \text{False}) \vee \text{Numpus}(\text{Stella}, \text{False})) \quad
70     \Rightarrow \quad \text{Numpus}(\text{Stella}, \text{False})
71     \]
72 - **Clause Hasil:** \(\( \text{Numpus}(\text{Stella}, \text{False}) \) \)
73
74 ##### **Langkah 3: Pemeriksaan Setelah Resolusi**
75 - **Cek Kontradiksi:** 
76 - Clause hasil \(\( \text{Numpus}(\text{Stella}, \text{False}) \) \) tidak menghasilkan kontradiksi. Proses resolusi lebih lanjut diperlukan untuk menentukan kesimpulan akhir, sehingga Label Cukup bernilai False.
77
78 ***Bentuk Akhir***
79 Clause Baru: { \(\( \text{Numpus}(\text{Stella}, \text{False}) \) \) }
80 Label Cukup: [False]
81 ***Akhir Blok***
82
83 ---
84
85 Contoh:
86
87 Clause 1: Vumpus(Wren, True)
88 Clause 2: Vumpus(Wren, False)
89
90 Mari kita uraikan proses resolusi untuk clause yang diberikan:
91
92 ##### **Langkah 1: Cek Literal yang Saling Komplemen**
93

```

```

94 **Clause yang Diberikan:***
95 - **Clause 1:** \(\ Vumpus(Wren, True) \)
96 - **Clause 2:** \(\ Vumpus(Wren, False) \)
97
98 **Identifikasi Literal yang Saling Komplemen :**
99 - \(\ Vumpus(Wren, True) \) dalam Clause 1 adalah komplemen dengan \(\ Vumpus(Wren,
   False) \) dalam Clause 2 (predikat dan argumen sama, tetapi nilai boolean berlawanan).
100
101 ### **Langkah 2: Terapkan Resolusi pada Kedua Clause**
102
103 - **Proses Resolusi:***
104   - **Aturan Digunakan:** **Kontradiksi Ditemukan** karena kedua istilah saling
     komplemen dan tidak ada istilah lain yang hadir.
105   - \(\ Vumpus(Wren, True) \) saling komplemen dengan \(\ Vumpus(Wren, False) \),
     menyebabkan kontradiksi langsung.
106
107   - **Clause Hasil:** Kontradiksi.
108
109 ### **Langkah 3: Pemeriksaan Setelah Resolusi**
110
111 - **Cek Kontradiksi:***
112   - Resolusi menghasilkan kontradiksi, sehingga cukup tercapai.
113
114 ***Bentuk Akhir***
115 Clause Baru: {Kontradiksi}
116 Label Cukup: [True]
117 ***Akhir Blok***
118
119 ---
120
121 Contoh:
122
123 Clause 1: \(\ Jompus(Alex, False) \lor TembusPandang(x, True) \)
124 Clause 2: \(\ Dumpus(x, False) \lor Jompus(x, True) \)
125
126 Mari kita uraikan proses resolusi untuk clause yang diberikan:
127
128 ### **Langkah 1: Cek Literal yang Saling Komplemen**
129
130 **Clause yang Diberikan:***
131 - **Clause 1:** \(\ Jompus(Alex, False) \lor TembusPandang(x, True) \)
132 - **Clause 2:** \(\ Dumpus(x, False) \lor Jompus(x, True) \)
133
134 ### **Langkah 1: Cek Literal yang Saling Komplemen**
135 - \(\ Jompus(Alex, False) \) pada Clause 1 berpotensi saling komplemen dengan \(
   Jompus(x, True) \) pada Clause 2. Instansiasi dari \(\ x \) ke \(\ Alex \) diperlukan
   untuk memastikan bahwa kedua literal saling komplemen.
136
137 **Instansiasikan Clause 2 dengan \(\ x = Alex \):**
138 - Clause 2 Asli: \(\ Dumpus(x, False) \lor Jompus(x, True) \)

```

```

139 - Clause 2 setelah Diinstansiasi: \(\ Dumpus(Alex, False) \lor Jompus(Alex, True) \)
140
141 **Identifikasi Literal yang Saling Komplemen Setelah Instansiasi:**
142 - \(\ Jompus(Alex, False) \) dalam Clause 1 adalah komplemen dari \(\ Jompus(Alex, True)
143   \) dalam Clause 2 setelah diinstansiasi (predikat sama, argumen sama, nilai boolean
144   berlawanan).
145
146 - **Proses Resolusi:**
147   - **Aturan Resolusi:** terdapat literal yang saling komplemen ditemukan pada Clause 1
148   dan Clause 2 setelah diinstansiasi yang terhubung oleh "OR" dengan literal lainnya,
149   maka formula resolusi di bawah ini dapat dilakukan.
150
151 - **Instansiasi Literal:** Instansiasikan literal dari Clause 1 dan Clause 2 kedalam
152   formula yang sudah diberikan
153
154 - **Clause Hasil:** \(\ TembusPandang(x, True) \lor Dumpus(Alex, False) \)
155
156
157 ### **Langkah 3: Pemeriksaan Setelah Resolusi**
158
159 - **Cek Kontradiksi:**
160   - Clause hasil \(\ TembusPandang(Alex, True) \lor Dumpus(Alex, False) \) tidak
161   menghasilkan kontradiksi. Proses resolusi lebih lanjut diperlukan untuk menentukan
162   kesimpulan akhir, sehingga Label Cukup bernilai False.
163
164 ***Bentuk Akhir***
165 Clause Baru: \(\ TembusPandang(Alex, True) \lor Dumpus(Alex, False) \}
166 Label Cukup: [False]
167 ***Akhir Blok***
168
169 Contoh:
170
171 Clause 1: Kecil(Alex, False)
172 Clause 2: \(\ Vumpus(x, False) \lor Numpus(x, True) \)
173
174 Mari kita uraikan proses resolusi untuk clause yang diberikan:
175
176 ### **Langkah 1: Cek Literal yang Saling Komplemen**
177
178 **Clause yang Diberikan:**
179 - **Clause 1:** \(\ Kecil(Alex, False) \)

```

```

180 - **Clause 2:** \(\( Vumpus(x, False) \lor Numpus(x, True) \)
181
182 **Identifikasi Literal yang Saling Komplemen :**
183 - Tidak ada istilah yang sama diantara 2 clause tersebut, sehingga tidak ada istilah
      yang saling komplemen ditemukan (predikat berbeda)
184
185 ### **Langkah 2: Terapkan Resolusi pada Kedua Clause**
186
187 - **Proses Resolusi:**
188 - **Tidak Ada Istilah Saling Komplemen Ditemukan:** Tidak ada istilah dalam Clause 1
      yang saling komplemen dengan istilah dalam Clause 2 karena predikat yang berbeda.
      Resolusi tidak dapat dilakukan karena tidak ada dasar untuk menggabungkan antara 2
      istilah yang berbeda
189
190 ### **Langkah 3: Pemeriksaan Setelah Resolusi**
191
192 - **Cek Kontradiksi:**
193 - Tidak ada istilah yang saling komplemen dan tidak ada resolusi yang dapat
      diterapkan, maka tidak ada kontradiksi. Proses resolusi lebih lanjut diperlukan untuk
      mengevaluasi nilai kebenaran clause, jadi label cukup harus False dan tidak perlu
      menambahkan nilai clause baru.
194
195 ***Bentuk Akhir***
196 Label Cukup: [False]
197 ***Akhir Blok***
198
199 ---
200
201 Contoh:
202
203 Clause 1:
204 \(\( Impus(Wren, False) \)
205 Clause 2:
206 \(\( Tumpus(x, False) \lor Impus(x, True) \)
207
208 Mari kita uraikan proses resolusi untuk clause yang diberikan:
209
210 ### **Langkah 1: Cek Literal yang Saling Komplemen**
211 **Clause yang Diberikan:**
212 - **Clause 1:** \(\( Impus(Wren, False) \)
213 - **Clause 2:** \(\( Tumpus(x, False) \lor Impus(x, True) \)
214
215 **Identifikasi Literal yang Saling Komplemen:**
216 - \(\( Impus(Wren, False) \) dalam Clause 1 berpotensi saling komplemen dengan \(
      Impus(x, True) \) dalam Clause 2. Instansiasi dari \(\( x \) ke \(\( Wren \) diperlukan
      untuk memastikan bahwa kedua literal saling komplemen.
217
218 **Instansiasikan Clause 2 dengan \(\( x = Wren \):**
219 - Clause 2 Asli: \(\( Tumpus(x, False) \lor Impus(x, True) \)
220 - Clause 2 setelah Diinstansiasi: \(\( Tumpus(Wren, False) \lor Impus(Wren, True) \)

```

```

221
222 **Identifikasi Literal yang Saling Komplemen Setelah Instansiasi:**
223 - \(\text{Impus}(\text{Wren}, \text{False})\) dalam Clause 1 adalah komplemen dari \(\text{Impus}(\text{Wren}, \text{True})\)
  dalam Clause 2 setelah diinstansiasi (predikat sama, argumen sama, nilai boolean
  berlawanan).
224
225 ##### **Langkah 2: Terapkan Resolusi pada Kedua Clause**
226 - **Proses Resolusi:**
227   - **Aturan Resolusi:** terdapat literal yang saling komplemen ditemukan pada Clause 1
     dan Clause 2 setelah diinstansiasi yang terhubung oleh "OR" dengan literal lainnya,
     maka formula resolusi di bawah ini dapat dilakukan.
228   \[
229     \(\neg (\text{P}(x, \text{False}) \vee \text{lor } B), \neg, (\text{P}(x, \text{True}) \vee \text{lor } C) \Rightarrow (B \vee \text{lor } C) \)
230   \]
231 - **Instansiasi Literal:** Instansiasikan literal dari Clause 1 dan Clause 2 kedalam
  formula yang sudah diberikan
232   \[
233     (\text{Impus}(\text{Wren}, \text{False})), \neg, (\text{Impus}(\text{Wren}, \text{True}) \vee \text{lor } \text{Tumpus}(\text{Wren}, \text{False})) \quad
234     \Rightarrow \quad (\text{Tumpus}(\text{Wren}, \text{False}))
235   \]
236 - **Clause Hasil:** \(\neg (\text{Tumpus}(\text{Wren}, \text{False}))\)
237
238 ##### **Langkah 3: Pemeriksaan Setelah Resolusi**
239 - **Cek Kontradiksi:**
240   - Clause hasil \(\neg (\text{Tumpus}(\text{Wren}, \text{False}))\) tidak menghasilkan kontradiksi. Proses
     resolusi lebih lanjut diperlukan untuk menentukan kesimpulan akhir, sehingga Label
     Cukup bernilai False.
241
242 ***Bentuk Akhir***
243 Clause Baru: { \(\neg (\text{Tumpus}(\text{Wren}, \text{False}))\} }
244 Label Cukup: [False]
245
246 ---
247
248 Silakan simpulkan informasi tersebut dengan format berikut. Ubah placeholder sesuai
  dengan kesimpulan:
249
250 ***Bentuk Akhir***
251 Clause Baru: {placeholder_for_new_clause}
252 Label Cukup: [placeholder_for_sufficiency_label]
253 ***Akhir Blok***
254
255 ---
256
257 **Dibawah ini tugas yang perlu Anda lakukan:**
258
259 Clause 1:
260 [[SOS]]
261 Clause 2:

```

```

262 [[SELECTED-CLAUSE]]
263
264 Mari kita uraikan proses resolusi untuk clause yang diberikan:

```

Kode 4.8: Template prompt dengan tambahan contoh dan mengganti operator NEG dengan

$$P(x, \text{NilaiKebalikan})$$

Walaupun sudah diperbaiki, masih terdapat beberapa aturan yang tidak dapat diterjemahkan dengan baik oleh model karena model seperti Qwen2.5 7B-Instruct-GGUF yang digunakan memiliki keterbatasan dalam memahami logika yang kompleks atau secara inheren tidak mempunyai *knowledge* dalam melakukan proses resolusi logika. Contoh dari kesalahan tersebut:

- **Input:**

Clause 1:

Zumpus(Fae, False)

Clause 2:

Zumpus(Fae, True)

- **Output:**

Dibawah ini tugas yang perlu Anda lakukan:

Clause 1:

Zumpus(Fae, False)

Clause 2:

Zumpus(Fae, True)

Mari kita uraikan proses resolusi untuk clause yang diberikan:

Langkah 1: Cek Literal yang Saling Komplemen

Clause yang Diberikan:

- **Clause 1:** *Zumpus(Fae, False)*
- **Clause 2:** *Zumpus(Fae, True)*

Identifikasi Literal yang Saling Komplemen:

- *Zumpus(Fae, False)* dalam Clause 1 berpotensi saling komplemen dengan

Zumpus(Fae, True) dalam Clause 2 (predikat dan argumen sama, tetapi nilai boolean berlawanan).

Langkah 2: Terapkan Resolusi pada Kedua Clause

- **Proses Resolusi:**

- **Aturan Resolusi:** terdapat literal yang saling komplemen ditemukan pada Clause 1 dan Clause 2 yang terhubung oleh "OR" dengan literal lainnya, maka formula resolusi di bawah ini dapat dilakukan.

$$P(x, \text{False}) \vee B, P(x, \text{True}) \vee C \Rightarrow B \vee C$$

- **Instansiasi Literal:** Instansiasikan literal dari Clause 1 dan Clause 2 kedalam formula yang sudah diberikan

$$\text{Zumpus}(Fae, \text{False}), \text{Zumpus}(Fae, \text{True}) \Rightarrow B \vee C$$

- **Clause Hasil:** $B \vee C$

Langkah 3: Pemeriksaan Setelah Resolusi

- **Cek Kontradiksi:**

- Clause hasil $B \vee C$ tidak menghasilkan kontradiksi. Proses resolusi lebih lanjut diperlukan untuk menentukan kesimpulan akhir, sehingga Label Cukup bernilai False.

Bentuk Akhir

Clause Baru: $B \vee C$

Label Cukup: [False]

Akhir Blok

- **Penjelasan:** Model gagal menemukan resolusi karena tidak mengenali bahwa *Zumpus(Fae, False)* dan *Zumpus(Fae, True)* adalah istilah yang saling komplemen, tetapi terlalu fokus pada format penulisan output sehingga mengabaikan proses resolusi yang sebenarnya.

Namun, prompt template 4.8 dianggap sudah cukup baik untuk digunakan dalam eksperimen.

4.4 Hasil Eksperimen dan Analisis

Bagian ini merupakan hasil akhir dari eksperimen yang telah dilakukan pada penelitian ini. Hasil akhir ini berupa tabel yang menunjukkan persentase keberhasilan dari masing-masing model dalam menjawab pertanyaan berdasarkan dua metode yang digunakan, yaitu *Naive Prompting* dan *Aristotle Framework*. Prompt yang digunakan pada eksperimen ini telah melalui proses *Prompt Refining* yang dijelaskan pada bagian sebelumnya, yaitu versi final dari masing-masing prompt.

4.4.1 Naive Prompting

Pada skenario ini, model diminta menjawab langsung "Benar" atau "Salah" berdasarkan premis. Eksperimen dilakukan dengan dua variasi: meminta jawaban dulu baru penjelasan (*After Answer*) dan sebaliknya (*Before Answer*).

Tabel 4.1: Hasil Eksperimen dengan *Naive Prompting*

	Qwen2.5 7B-Instruct-GGUF	SEA-LION v3-Llama-8B-GGUF	SahabatAI v1-Llama-8B-GGUF
Naive Prompting			
After Answer	51.40%	56.20%	61.40%
Before Answer	81.00%	76.80%	67.80%

Analisis:

- Keunggulan Qwen pada CoT:** Qwen2.5 menunjukkan lonjakan performa tertinggi (81%) ketika menggunakan metode *Before Answer*. Ini konsisten dengan literatur yang menyatakan bahwa model yang dilatih pada korpus kode/matematika besar (seperti Qwen) memiliki kemampuan *Chain-of-Thought* internal yang kuat (Wei et al. (2022)).
- Kekuatan Semantik Sahabat-AI:** Pada mode *After Answer* (yang lebih mengandalkan intuisi bahasa langsung), Sahabat-AI unggul (61.4%). Ini mengindikasikan bahwa model ini memiliki pemahaman Bahasa Indonesia yang lebih natural, sehingga intuisi "tebakan"-nya lebih akurat dibandingkan model global.

4.4.2 Aristotle

Tabel berikut menunjukkan hasil ketika penalaran dipindahkan dari "otak" model ke *pipeline* simbolik.

Tabel 4.2: Hasil Eksperimen dengan *Aristotle Framework*

	Qwen2.5 7B-Instruct-GGUF	SEA-LION v3-Llama-8B-GGUF	SahabatAI v1-Llama-8B-GGUF
Aristotle	14.00%	81.60%	61.20%

Analisis: Ada beberapa temuan menarik dalam eksperimen ini:

- **Anomali Qwen (14%):** Meskipun unggul di Naive Prompting, Qwen gagal total dalam *framework* ini. Analisis log menunjukkan kegagalan ini bukan pada logika, melainkan pada **kepatuhan format sintaks** (*parsability*). Qwen sering menghasilkan output yang terlalu mengikuti format pada tahap *search & resolve* tanpa mengandalkan kemampuan penalaran logikanya, sehingga jawaban pada tahap tersebut menjadi tidak valid.
- **Dominasi SEA-LION (81.6%):** SEA-LION v3 menunjukkan performa terbaik. Ini membuktikan hipotesis bahwa model regional yang dilatih dengan instruksi spesifik bahasa lokal (dan mungkin data *multilingual alignment* yang lebih baik) mampu menjadi "Penerjemah Logika" yang lebih patuh pada tahapan-tahapan framework Aristotle pada dataset berbahasa Indonesia.
- **Stabilitas Sahabat-AI:** Sahabat-AI menunjukkan performa yang seimbang. Meskipun tidak setinggi SEA-LION, akurasinya (61.2%) sebanding dengan performa *Naive*-nya, menunjukkan konsistensi pemahaman.

Temuan paling signifikan dalam penelitian ini adalah bahwa **SEA-LION v3 (8B)**, sebuah model regional, mampu mengalahkan model global SoTA (Qwen2.5 7B) dalam tugas yang terstruktur ini. Hal ini menunjukkan bahwa untuk aplikasi yang memerlukan integrasi dengan sistem simbolik (seperti database atau mesin logika) dalam Bahasa Indonesia, model yang melalui proses *Continued Pre-Training* (CPT) pada data regional jauh lebih andal. Model yang cenderung "terlalu kreatif" atau "terlalu cerewet" atau "terlalu patuh pada format", yang justru menjadi masalah jika di implementasikan dalam sebuah pipeline simbolik.

Hasil eksperimen ini mengonfirmasi kelemahan utama sistem *framework translation-decomposition-search-resolve*, yaitu *error propagation*, jika terjadi kesalahan dalam salah satu tahap, maka tahap selanjutnya pasti salah juga. Dalam metode *Naive*, kesalahan model terdistribusi secara probabilistik. Namun dalam metode Aristotle, kesalahan pada tahap-tahap awal akan menjadi katastropik untuk tahap-tahap selanjutnya. Lalu juga ada beberapa poin penting:

- Jika Model gagal dalam tahap *translation*, seperti "Setiap Wumpus adalah Jompus" menjadi $\forall x(Wumpus(x)) \rightarrow Jompus(x)$ dengan sintaks yang tepat 100%, maka modul *search* tidak akan memiliki input yang valid, dan *data point* tersebut pasti salah.
- Kegagalan Qwen (14%) adalah bukti nyata bahwa **kecerdasan penalaran** (*Reasoning IQ*) berbeda dengan **kepatuhan instruksi format** (*Instruction Following*) pada bahasa spesifik. dalam hal ini bahasa Indonesia, yaitu model Qwen terlalu patuh terhadap format, sehingga tidak dapat me-*resolve* logika dengan benar.

BAB 5

PENUTUP

Bab ini merangkum temuan-temuan utama yang diperoleh dari rangkaian eksperimen validasi kerangka kerja *Neuro-Symbolic* pada dataset logika berbahasa Indonesia. Kesimpulan disusun untuk menjawab rumusan masalah yang telah ditetapkan pada Bab 1, diikuti dengan saran strategis untuk pengembangan penelitian selanjutnya.

5.1 Kesimpulan

Berdasarkan hasil evaluasi pada model *open-weight* dengan parameter rendah (7B-9B) yang dikuantifikasi pada dataset berbahasa Indonesia, penelitian ini menghasilkan tiga kesimpulan utama:

1. **Mekanisme Penalaran Logis pada Bahasa Indonesia:** Penalaran logis pada konteks Bahasa Indonesia dalam penelitian ini dilakukan melalui pembuatan sumber daya data dan eksekusi arsitektur *Neuro-Symbolic*. Proses ini diawali dengan adaptasi dataset, di mana dataset sintetis ProntoQA diterjemahkan ke dalam Bahasa Indonesia dengan mempertahankan struktur ontologi logika formal (fakta aturan, dan konjektur) namun menyesuaikan representasi linguistiknya. Selanjutnya, mekanisme penalaran logis dijalankan melalui empat tahapan sekuensial dalam *framework Aristotle*:
 - (a) **Logical Translation**, di mana SLM melakukan *translation* narasi alamiah Indonesia ke dalam sintaks *First-Order Logic* (FOL)
 - (b) **Decomposition**, yang mengonversi aturan dalam format FOL menjadi *Conjunctive Normal Form* (CNF)
 - (c) **Search Router**, yang memilah klausa berkomplemen untuk disiapkan pada tahap resolusi
 - (d) **Logical Resolver**, yang melakukan deduksi deterministik menggunakan prinsip resolusi untuk menghasilkan jawaban yang valid secara simbolik dan mengembalikan ke Search Router untuk mencari klausa baru
2. **Perbandingan Performa Antar Model Open-Weight:**
Terdapat divergensi hasil yang signifikan antara model global dan model regional/nasional dalam tugas translasi simbolik ini:
 - **SEA-LION v3 (Regional)** mencatat performa terbaik dengan akurasi **81.60%**. Mo-

del ini menunjukkan keseimbangan terbaik antara pemahaman instruksi dalam Bahasa Indonesia dan kepatuhan terhadap format logika formal.

- **Sahabat-AI v1 (Nasional)** menunjukkan performa yang stabil (**61.20%**), mengungguli model global dalam memahami nuansa semantik lokal, meskipun masih dibawah SEA-LION dalam konsistensi sintaks kompleks.
- **Qwen2.5 (Global)**, meskipun dikenal unggul dalam *code generation*, mengalami kegagalan katastropik (**14.00%**) pada metode ini. Kegagalan ini disebabkan oleh ketidakmampuan model untuk menjawab sesuai penalaran logis ketika dipaksa mengikuti format ketat, menandakan bahwa kapabilitas *instruction following* tidak selalu berbanding lurus dengan kemampuan penalaran logis dalam konteks bahasa tertentu.

3. Efektivitas Aristotle vs Naive Prompting:

Penerapan *framework Aristotle* terbukti **meningkatkan reliabilitas** jika dan hanya jika model memiliki kapabilitas *instruction following* yang kuat dalam bahasa target.

- Pada model **SEA-LION**, penggunaan *framework Aristotle* meningkatkan akurasi dari 76.80% (*Naive*) menjadi **81.60%**. Ini membuktikan bahwa *framework Aristotle* dapat memanfaatkan kemampuan model untuk menghasilkan logika yang lebih tepat ketika diarahkan dengan benar.
- Sebaliknya, pada model **Qwen2.5**, performa justru turun drastis dari 81.00% (*Naive*) menjadi 14.00%. Hal ini menunjukkan bahwa model yang tidak dapat mematuhi atau terlalu patuh pada format *prompt template* tidak akan menghasilkan penalaran logis yang baik, berbeda dengan *Naive Prompting* yang lebih toleran terhadap kesalahan parsial.
- *Framework Aristotle* memiliki sifat *brittle*, kesalahan kecil pada satu tahap akan merusak keseluruhan hasil atau *error propagation*. Hal ini bisa dilihat pada penurunan performa Sahabat-AI dari 62.40% (*Naive*) menjadi 61.20% ketika menggunakan *framework Aristotle* karena kesalahan pada tahap awal (seperti *translation*) menyebabkan kegagalan pada tahap selanjutnya.

Secara keseluruhan, penelitian ini menyimpulkan bahwa untuk penalaran logika pada dataset berbahasa Indonesia pada sumber daya terbatas (model kecil & terkuantifikasi), model regional seperti SEA-LION yang dipadukan dengan arsitektur *Neuro-Symbolic* menawarkan solusi yang lebih baik dibandingkan mengandalkan intuisi model semata.

5.2 Saran

Berdasarkan temuan dan keterbatasan penelitian ini, disarankan beberapa langkah pengembangan untuk penelitian selanjutnya:

1. **Penerapan Grammar-Constrained Decoding:** Untuk mengatasi masalah *parsability* (seperti yang dialami Qwen), penelitian lebih lanjut sebaiknya mengintegrasikan penegakan tata bahasa (seperti format GBNF¹) saat inferensi. Ini akan memaksa model, bahkan yang terkuantisasi, untuk menghasilkan output yang 100% valid secara sintaks, seperti mengikuti format FOL, sehingga evaluasi dapat berfokus murni pada logika semantik.
2. **Fine-Tuning Khusus (Instruction Tuning):** Alih-alih hanya mengandalkan *Prompt Engineering (few-shot)*, disarankan untuk melakukan *Fine-Tuning* ringan (seperti LoRA) pada model Sahabat-AI atau SEA-LION menggunakan dataset pasangan (Teks Indonesia ke bentuk FOL). Hal ini akan meningkatkan konsistensi model dalam menangani struktur kalimat kompleks Bahasa Indonesia.
3. **Peningkatan Kompleksitas Dataset:** Penelitian ini hanya terbatas pada dataset Pron-toQA (logika deduktif sintetik). Pengujian selanjutnya perlu memperluas cakupan ke dataset, seperti menggunakan dataset yang dipakai pada penelitian Aristotle aslinya, yaitu ProofWriter (Tafjord et al. (2021)) dan LogicNLI (Tian et al. (2021)).

¹<https://github.com/ggml-org/llama.cpp/blob/master/grammars/README.md>

BIBLIOGRAFI

- Belcak, P., Heinrich, G., Diao, S., Fu, Y., Dong, X., Muralidharan, S., Lin, Y. C., and Molchanov, P. (2025). Small language models are the future of agentic ai.
- Brachman, R. and Levesque, H. (2004). *Knowledge Representation and Reasoning*. The Morgan Kaufmann Series in Artificial Intelligence. Elsevier Science.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.
- Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. (2022). Llm.int8(): 8-bit matrix multiplication for transformers at scale. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA. Curran Associates Inc.
- Fitting, M. (1996). *First-Order Logic and Automated Theorem Proving*. Springer, 2nd edition.
- Huth, M. R. A. and Ryan, M. D. (2004). *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge University Press, 2nd edition.
- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.
- Pan, L., Albalak, A., Wang, X., and Wang, W. (2023). Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, Singapore. Association for Computational

Linguistics.

- Saparov, A. and He, H. (2023). Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*.
- Subramanian, S., Elango, V., and Gungor, M. (2025). Small language models (slms) can still pack a punch: A survey.
- Tafjord, O., Dalvi, B., and Clark, P. (2021). ProofWriter: Generating implications, proofs, and abductive statements over natural language. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.
- Tian, J., Li, Y., Chen, W., Xiao, L., He, H., and Jin, Y. (2021). Diagnosing the first-order logical reasoning ability through LogicNLI. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Wang, F., Zhang, Z., Zhang, X., Wu, Z., Mo, T., Lu, Q., Wang, W., Li, R., Xu, J., Tang, X., He, Q., Ma, Y., Huang, M., and Wang, S. (2025). A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness. *ACM Trans. Intell. Syst. Technol.*, 16(6).
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E. H., Xia, F., Le, Q., and Zhou, D. (2022). Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903.
- Xu, J., Fei, H., Luo, M., Liu, Q., Pan, L., Wang, W. Y., Nakov, P., Lee, M., and Hsu, W. (2025). Aristotle: Mastering logical reasoning with A logic-complete decompose-search-resolve framework. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*.

- Xu, J., Fei, H., Pan, L., Liu, Q., Lee, M.-L., and Hsu, W. (2024). Faithful logical reasoning via symbolic chain-of-thought. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13326–13365, Bangkok, Thailand. Association for Computational Linguistics.
- Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: deliberate problem solving with large language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS ’23, Red Hook, NY, USA. Curran Associates Inc.

