Analisis Performa Chatbot IT Support Berbasis RAG dengan Optimasi Retrieval Bertahap

Moch Raffry Syalwa Rizqullah
Program Studi Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
20523210@students.uii.ac.id

Mukhammad Andri Setiawan, S.T., M.Sc., Ph.D.
Program Studi Informatika
Universitas Islam Indonesia
Yogyakarta, Indonesia
andri.setiawan@uii.ac.id

***Abstrak*—Tingginya volume tiket berulang dan keterbatasan sumber daya manusia menjadi tantangan utama dalam layanan *IT Support* di Badan Sistem Informasi (BSI) Universitas Islam Indonesia (UII). Meskipun *Chatbot* berbasis Large Language Model (LLM) menawarkan potensi otomatisasi, penerapannya terkendala oleh risiko halusinasi dan latensi inferensi yang tinggi. Penelitian ini menganalisis performa *chatbot* yang menerapkan arsitektur *Retrieval-Augmented Generation* (RAG) dengan optimasi *multi-stage retrieval*. Sistem mengintegrasikan LangGraph *router* untuk klasifikasi maksud, pencarian hibrida (*hybrid search*) yang menggabungkan *dense* dan *sparse retrieval*, *cross-encoder reranking*, serta Redis *caching* untuk efisiensi latensi. Evaluasi dilakukan menggunakan 150 query nyata yang mencakup 10 topik layanan (seperti Nvivo, VPN eduVPN, dan Office 365) serta pengujian ketahanan terhadap noise query. Hasil eksperimen menunjukkan bahwa sistem *hybrid search* mencapai rata-rata latensi *cold query* sebesar 4120 ms dengan peningkatan kecepatan (*speedup*) 2,01× pada *warm query*. Sistem mampu menjaga relevansi jawaban dengan skor *similarity* rata-rata 0,628 dan berhasil menolak 98% *noise query* yang berada di luar domain *IT Support*. Temuan ini menunjukkan bahwa penggabungan *router*, *hybrid retrieval*, dan mekanisme *caching* menghasilkan *chatbot* yang akurat, efisien, dan tangguh (robust) terhadap pertanyaan di luar domain.**

Kata kunci*—*Retrieval-Augmented Generation, Chatbot, IT Support, Hybrid search, Reranking, Redis Caching

I. PENDAHULUAN

Di Universitas Islam Indonesia (UII), Badan Sistem Informasi (BSI) bertanggung jawab memberikan layanan dukungan teknologi informasi kepada sivitas akademika yang terdiri dari mahasiswa, dosen, dan tenaga kependidikan. Dengan sumber daya manusia yang terbatas dan jam operasional yang mengikuti jam kerja kantor, BSI menghadapi tantangan dalam menangani volume permintaan bantuan yang tinggi dan bervariasi. Hasil wawancara dengan tim *IT Support BSI* UII pada hari Jumat, 14 Februari 2025, mengungkapkan bahwa tim merasa kewalahan akibat peningkatan jumlah permintaan bantuan. Banyak permintaan tersebut bersifat berulang dan dapat diotomatisasi, misalnya pengaturan ulang kata sandi, permintaan informasi layanan, dan panduan teknis dasar. Kondisi ini berdampak pada keterlambatan respon, penumpukan tiket, serta menurunnya tingkat kepuasan pengguna dan meningkatnya beban kerja tim dukungan teknis.

Chatbot berbasis *large language model* (*LLM*) menawarkan solusi untuk mengotomatisasi respon terhadap pertanyaan yang sering diajukan, menyediakan layanan sepanjang waktu, dan mengurangi beban tim *IT Support*. *LLM* merupakan model pembelajaran mesin berukuran besar yang mampu memahami dan menghasilkan teks dalam bahasa alami dengan tingkat pemahaman kontekstual yang tinggi. Namun, penggunaan *LLM* secara langsung menghadapi dua tantangan utama. Pertama, risiko halusinasi, yaitu kondisi ketika model menghasilkan informasi yang tidak akurat atau mengada-ada karena menjawab di luar cakupan pengetahuan yang dimilikinya, khususnya pengetahuan internal organisasi yang tidak tersedia dalam data pelatihannya. Kedua, latensi tinggi akibat proses inferensi yang kompleks, yang dapat mencapai 5-10 detik per pertanyaan dan mengganggu pengalaman pengguna.

Arsitektur *Retrieval-Augmented Generation* (*RAG*) dikembangkan untuk mengatasi permasalahan halusinasi dengan cara mengambil (*retrieve*) dokumen relevan dari basis pengetahuan organisasi, kemudian menyuntikkan dokumen tersebut ke dalam konteks *LLM* sebelum menghasilkan jawaban. Dengan pendekatan ini, jawaban yang dihasilkan lebih tertambat (*grounded*) pada sumber informasi resmi yang dapat diverifikasi dan diaudit [1]. Mekanisme *RAG* memastikan bahwa sistem hanya menjawab berdasarkan pengetahuan yang tersedia dalam basis data organisasi, sehingga mengurangi risiko informasi yang menyesatkan.

Studi terkini menunjukkan bahwa kualitas sistem *RAG* tidak hanya ditentukan oleh kemampuan *LLM* dalam menghasilkan teks, tetapi juga sangat bergantung pada kualitas modul pengambilan dokumen (*retrieval*). Komponen krusial meliputi pemilihan algoritma *retriever* sebuah komponen yang mencari dokumen relevan, mekanisme penyusunan ulang peringkat atau *reranking* untuk memilih dokumen paling relevan dari kandidat yang ditemukan, serta desain alur bertahap *multi-stage pipeline* yang mengoptimalkan keseimbangan antara akurasi dan kecepatan [2]. Penelitian HyperRAG menegaskan pentingnya eksplorasi keseimbangan antara kualitas jawaban dan efisiensi komputasi, termasuk penggunaan ulang hasil komputasi pada tahap penyusunan ulang peringkat untuk menurunkan latensi [3].

Dalam konteks layanan *IT Support*, sistem *chatbot* tidak hanya dituntut menghasilkan jawaban yang akurat, tetapi juga harus responsif dengan waktu respon yang cepat serta tangguh terhadap pertanyaan di luar cakupan layanan. Evaluasi dengan dimensi seperti ketahanan terhadap gangguan, kemampuan menolak pertanyaan yang tidak relevan, dan ketahanan terhadap informasi yang bertentangan menjadi penting untuk memastikan bahwa *chatbot* tidak memberikan jawaban yang menyesatkan ketika informasi tidak tersedia dalam basis pengetahuan [1].

Penelitian ini berfokus pada analisis performa *chatbot IT Support* BSI UII yang menggunakan arsitektur *RAG* dengan pengambilan dokumen bertahap serta optimasi latensi melalui mekanisme penyimpanan sementara atau *caching*. Evaluasi dilakukan dengan membedakan dua jenis pertanyaan:

pertanyaan awal dalam sesi baru *cold query* yang memerlukan pengambilan dokumen lengkap, dan pertanyaan lanjutan dalam sesi yang sama *warm query* yang dapat memanfaatkan hasil pengambilan sebelumnya yang tersimpan dalam *cache*.

Secara spesifik, penelitian ini memberikan tiga kontribusi utama. Pertama, merancang dan mendeskripsikan arsitektur *chatbot IT Support* berbasis *RAG* yang mengintegrasikan pengarah aliran berbasis graf *(LangGraph router)* untuk mengklasifikasikan jenis pertanyaan, pencarian hibrida *(hybrid search)* yang menggabungkan pencarian berbasis makna semantik dan kecocokan kata kunci, penyaringan berbasis metadata, serta penyimpanan sementara Redis pada tingkat sesi percakapan. Kedua, mengevaluasi keseimbangan antara latensi *cold query* dan *warm query* menggunakan data penggunaan nyata pada 10 kategori layanan IT Support, yang meliputi penggunaan perangkat lunak analisis kualitatif Nvivo oleh dosen, akses VPN kampus menggunakan eduVPN, konfigurasi Microsoft Office 365, pengelolaan Google Drive institusi, akses jurnal elektronik, konfigurasi sertifikat keamanan jaringan EAP TLS, penggunaan sistem informasi UIISmart, penyusunan Sasaran Kinerja Pegawai di UIIKinerja, serta penggunaan sistem perkuliahan UIIPerkuliahan. Ketiga, menganalisis efektivitas tahap pengambilan dokumen awal *(first-stage retrieval)* melalui pengukuran nilai kesamaan semantik *(similarity score)* serta ketahanan mekanisme pengarahan terhadap pertanyaan gangguan yang berada di luar cakupan pengetahuan *chatbot*.

II. KAJIAN PUSTAKA

A. Retrieval Augmented Generation dan Evaluasi

RAG menggabungkan modul pengambilan dokumen *retriever* dengan *LLM* generatif untuk menghasilkan jawaban yang tertambat pada dokumen yang diambil dari basis pengetahuan. Chen dkk. [1] memperkenalkan kerangka evaluasi komprehensif yang menekankan empat dimensi krusial: ketahanan terhadap gangguan *noise robustness*, kemampuan menolak pertanyaan negatif *negative rejection*, integrasi informasi dari berbagai dokumen, dan ketahanan terhadap fakta yang bertentangan. Kerangka evaluasi ini relevan untuk penelitian ini karena *chatbot IT Support* harus mampu menolak pertanyaan di luar domain dan tidak memberikan informasi yang menyesatkan ketika pengetahuan tidak tersedia dalam basis pengetahuan.

Krasakis dkk. [2] mengembangkan pendekatan *late-interaction retriever multilingual* yang memungkinkan pemodelan kesamaan pada tingkat token, memberikan keseimbangan antara efisiensi dan ketepatan pada skenario *dense retrieval* berskala besar. Pendekatan ini sangat relevan untuk *chatbot IT Support* yang harus menjawab pertanyaan dalam bahasa Indonesia dengan variasi gaya bahasa dan dialek pengguna yang beragam.

B. Optimasi Multi-stage Retrieval

An dkk. [3] dalam HyperRAG mengidentifikasi keseimbangan kritis *trade-off* antara kualitas jawaban dan efisiensi komputasi dalam sistem *RAG*. Penelitian tersebut menunjukkan bahwa penggunaan ulang *cache (KV-cache reuse)* pada tahap penyusunan ulang peringkat dapat meningkatkan *throughput* sistem sebesar 2-3× tanpa mengorbankan kualitas generasi. Temuan ini menjadi dasar desain mekanisme *caching* dalam penelitian ini, di mana sistem membedakan *cold query* yang memerlukan pengambilan dokumen lengkap dengan *warm query* yang memanfaatkan hasil *cache*.

Tian dkk. [4] mengusulkan CoRank, skema *compact reranking* yang menggabungkan fitur dokumen dengan kemampuan *LLM* untuk mengurangi beban komputasi. Zhang dkk. [5] dalam ReFIT memanfaatkan sinyal dari *reranker* sebagai umpan balik relevansi *(relevance feedback)*, memungkinkan perbaikan adaptif tanpa pelatihan ulang. Liu dkk. [6] menunjukkan bahwa *listwise reranking* berbasis *LLM* dapat mengoptimalkan struktur peringkat secara global, meskipun dengan *overhead* latensi yang signifikan. Wang dkk. [7] mengembangkan ColBERT-PRF yang menerapkan *pseudo relevance feedback* semantik untuk memperkuat *first-stage retriever*. Pendekatan-pendekatan ini memberikan wawasan tentang berbagai strategi optimasi *reranking* yang dapat diterapkan bergantung pada karakteristik domain dan skala basis pengetahuan.

C. Chatbot Percakapan dan Query Rewriting

Trippas dkk. [8] memperkenalkan konsep *mixed-initiative query rewriting* dalam pencarian percakapan, di mana sistem dan pengguna berkolaborasi untuk memperjelas pertanyaan yang ambigu. Pendekatan ini relevan untuk *chatbot IT Support* yang seringkali menghadapi pertanyaan tidak terstruktur dengan konteks yang tidak lengkap. Kumar dkk. [10] menunjukkan bahwa pendekatan *hybrid deep learning* dapat meningkatkan pemahaman maksud *(intent understanding)* dalam sistem *chatbot*. Lee dkk. [9] menunjukkan bahwa *multi-stage language model pipeline* dapat dioptimalkan dengan menempatkan komponen yang lebih mahal secara komputasi hanya pada *subset query* yang benar-benar membutuhkannya.

D. Posisi Penelitian dan Gap Analysis

Tabel 1 menyajikan perbandingan penelitian terdahulu dengan penelitian ini dari berbagai aspek teknis dan kontekstual.

Table 1 PERBANDINGAN PENELITIAN TERDAHULU

Penelitian	***Retrieval***	***Reranking***	***Caching***	**Evaluasi Latensi**	**Robustness**	**Domain**
HyperRAG [3]	*Dense*	Ya	Ya	Ya	Tidak	QA umum
CoRank [4]	*Dense*	Ya	Tidak	Ya	Tidak	*Scientific*
ColBERT-PRF [7]	*Dense*	Tidak	Tidak	Tidak	Tidak	Dokumen
Chen dkk. [1]	*Dense*	Tidak	Tidak	Tidak	Ya	QA umum
ReFIT [5]	*Dense*	Ya	Tidak	Tidak	Tidak	IR umum
Lee dkk. [9]	*Multi-stage*	Ya	Tidak	Ya	Tidak	*Text retrieval*
Penelitian Ini	*Hybrid*	Ya	Ya	Ya	Ya	*IT Support*

Berdasarkan Tabel 1, penelitian ini mengisi beberapa kesenjangan dari penelitian terdahulu. HyperRAG [3] telah menunjukkan efektivitas *caching* untuk meningkatkan *throughput*, tetapi hanya menggunakan *dense retrieval* yang kurang optimal untuk *query* teknis *IT Support* yang memerlukan kecocokan kata kunci eksak seperti kode error atau nama aplikasi. Penelitian ini mengatasi keterbatasan tersebut dengan *hybrid search* yang menggabungkan *dense retrieval* untuk kesamaan semantik dan *sparse retrieval*

untuk kecocokan kata kunci, sehingga lebih sesuai dengan karakteristik pertanyaan *IT Support* yang bervariasi antara konseptual dan prosedural.

Evaluasi latensi dalam penelitian terdahulu seperti An dkk. [3] dan Lee dkk. [9] umumnya hanya mengukur *throughput* umum tanpa membedakan karakteristik *query* dalam konteks percakapan. Penelitian ini melakukan analisis terperinci dengan membandingkan latensi *cold query* yang memerlukan eksekusi penuh *pipeline retrieval* dengan *warm query* yang memanfaatkan *cache*, memberikan wawasan tentang *trade-off* kualitas dan efisiensi dalam penggunaan *chatbot multi-turn* yang sesungguhnya.

Chen dkk. [1] mengevaluasi *robustness* menggunakan tolak ukur sintetis, sementara penelitian ini menggunakan *noise query* riil dari interaksi pengguna nyata dengan *chatbot IT Support*. Selain itu, integrasi LangGraph *router* untuk klasifikasi maksud sebelum *retrieval* memungkinkan sistem menolak pertanyaan di luar domain sejak awal tanpa *retrieval* yang tidak perlu, meningkatkan efisiensi dan mengurangi risiko jawaban menyesatkan.

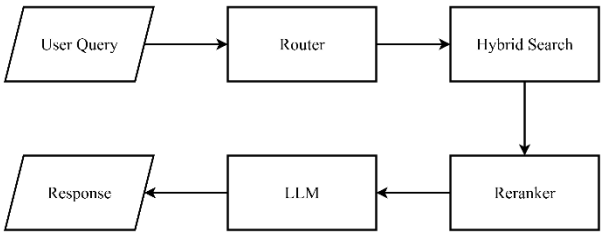
Seluruh penelitian terdahulu menggunakan korpus berbahasa Inggris atau Mandarin, sementara penelitian ini menggunakan basis pengetahuan berbahasa Indonesia dengan karakteristik unik berupa variasi istilah teknis campuran Indonesia-Inggris yang umum dalam *IT Support*. Dengan demikian, kontribusi penelitian ini terletak pada kombinasi arsitektur *multi-stage retrieval* yang dioptimalkan untuk karakteristik spesifik *chatbot IT Support* berbahasa Indonesia dengan evaluasi komprehensif terhadap *trade-off latensi* dan *robustness* dalam konteks penggunaan nyata.

III. METODOLOGI

Penelitian ini menggunakan arsitektur *chatbot IT Support* yang telah ada dengan mengintegrasikan komponen-komponen *open-source* untuk membangun sistem *RAG* yang dioptimalkan bagi layanan *IT Support* BSI UII. Tujuan penelitian adalah menganalisis performa arsitektur existing yang telah disesuaikan dengan konteks spesifik *IT Support* berbahasa Indonesia, bukan mengusulkan model atau arsitektur baru.

A. Arsitektur Sistem

Gambar 1 mengilustrasikan alur pemrosesan *chatbot IT Support* BSI UII yang terdiri dari enam tahap utama yang bekerja secara sekuensial. *Query* pengguna pertama kali diproses oleh LangGraph *Router*, yaitu komponen *routing* berbasis *state machine* yang mengklasifikasikan maksud dan membedakan query di dalam domain yang memerlukan *retrieval* dari *query* diluar domain yang harus ditolak. *Query* didalam domain kemudian diproses melalui pencarian hibrida yang menggabungkan *dense retrieval* menggunakan text-*embedding*-3-large dengan *sparse retrieval* berbasis BM25, yaitu algoritma peringkat probabilistik yang menghitung relevansi dokumen berdasarkan frekuensi kemunculan, pada Pinecone *vector database*, diikuti oleh pemeringkatan ulang *Cross-encoder* menggunakan model mxbai-rerank-base-v1 untuk menyusun ulang kandidat dokumen berdasarkan relevansi kontekstual. Dokumen hasil *reranking* digunakan sebagai konteks untuk LLM Generation menggunakan GPT-4o dengan temperature 0,5 untuk menyeimbangkan konsistensi dan variasi respon, dan seluruh *pipeline* ini didukung oleh Redis *Caching* dengan TTL 3600 detik yang menyimpan riwayat percakapan dan hasil *retrieval* untuk mempercepat respon pada *warm query*.



Gambar 1 Arsitektur *chatbot IT Support* berbasis *RAG*

B. Basis pengetahuan dan Dataset

Sistem menggunakan basis pengetahuan internal BSI UII yang terdiri dari dokumentasi layanan *IT Support* dalam bahasa Indonesia. Tabel 2 menyajikan statistik basis pengetahuan yang digunakan dalam penelitian.

Table 2 Statistik Knowledge Base IT Support

Metrik	Nilai
Total Dokumen	48
Dokumen Public	24
Dokumen Internal	24
Total *Chunks*	439
Table *Chunks*	8
Rata-rata *Chunks* per Dokumen	9,15
Panjang Chunk Minimum	4 kata
Panjang Chunk Median	204 kata
Panjang Chunk Maksimum	399 kata

Basis pengetahuan mencakup 10 kategori layanan IT Support Software analisis kualitatif Nvivo,Akses VPN menggunakan eduVPN,Pengajuan potongan biaya mahasiswa di system UIITagihan, Konfigurasi Microsoft Office 365, Pengelolaan Google Drive institusi, Akses jurnal elektronik melalui, Konfigurasi sertifikat keamanan EAP-TLS, Sistem informasi UIISmart, Penyusunan Sasaran Kinerja Pegawai (SKP) di UIIKinerja, dan Sistem perkuliahan UIIPerkuliahan untuk dosen.

C. Desain Eksperimen

Evaluasi dilakukan menggunakan 150 query *real-world* yang dikumpulkan dari interaksi pengguna dengan tim IT Support BSI UII. Dataset terdiri dari tiga jenis query yang pertama *Cold Query*, *Query* pertama dalam sesi percakapan baru yang menjalankan seluruh *pipeline retrieval* tanpa memanfaatkan *cache*. *Cold query* didistribusikan merata dengan 5 query per kategori layanan. *Warm Query*, Query lanjutan dalam sesi percakapan yang sama yang memanfaatkan *Redis cache* dan riwayat sesi untuk mempercepat respon. Setiap *warm query* merupakan *follow-up question* dari *cold query* yang berhasil, dengan 5 *warm query* per kategori layanan. *Noise Query*, *Query* di luar domain *IT Support* yang digunakan untuk menguji kemampuan penolakan sistem melalui LangGraph *router*. *Noise query* mencakup pertanyaan umum, permintaan yang tidak etis, dan pertanyaan *IT* yang di luar *scope* BSI. Distribusi 5 *noise query* per kategori memastikan variasi konteks yang beragam.

D. Konfigurasi Teknis

Sistem diimplementasikan menggunakan komponen *open-source* dengan konfigurasi sebagai berikut. *Embedding model* menggunakan text-*embedding*-3-large dari OpenAI dengan dimensi 3072, dikombinasikan dengan *sparse encoder* BM25 menggunakan library Rank-BM25 dengan tokenisasi bahasa Indonesia. Vector database menggunakan Pinecone dengan IVF index untuk mendukung kemampuan pencarian hibrida, sementara *reranker* menggunakan model mxbai-rerank-base-v1 dari Hugging Face. Tahap generasi menggunakan LLM GPT-4o dengan temperature 0,5, dan mekanisme *caching* diimplementasikan menggunakan Redis dengan TTL 3600 detik. Alur orkestrasi menggunakan LangGraph untuk *state machine routing*, dengan seluruh kode implementasi menggunakan Python, library LangChain, OpenAI SDK, dan Pinecone Client.

E. Metode Evaluasi

Evaluasi sistem dilakukan melalui tiga metrik utama yang diukur secara berurutan. Metrik pertama adalah Cosine *Similarity* yang mengukur kesamaan semantik antara vektor query q dan vektor dokumen d sebagaimana ditunjukkan pada Persamaan (1).

$$\text{sim}(q,d) = \frac{q.d}{||q||||d||} \quad (1)$$

Cosine similarity dipilih karena merupakan metrik standar untuk *task retrieval* pada sistem berbasis *embedding* [1][2] dengan nilai berkisar 0 tidak relevan hingga 1 sangat relevan, lebih sesuai dibanding *accuracy* atau *F1-score* yang memerlukan label kebenaran biner sementara relevansi dokumen bersifat kontinu. Metrik kedua adalah *Speedup* yang mengukur rasio peningkatan kecepatan antara *cold query* dan *warm query* sebagaimana ditunjukkan pada Persamaan (2), di mana *t_cold* adalah rata-rata latensi *cold query* dan *t_warm* adalah rata-rata latensi *warm query* pada kategori layanan yang sama.

$$Speedup = \frac{t_{cold}}{t_{warm}} \quad (2)$$

Metrik ketiga adalah *Latency Improvement* yang mengukur persentase reduksi latensi akibat *caching* sebagaimana ditunjukkan pada Persamaan (3), dengan urutan evaluasi mengikuti alur *cosine similarity* untuk kualitas *retrieval*, *speedup* untuk efektivitas *caching*, dan *latency improvement* untuk persentase perbaikan.

$$Improvement = \frac{t_{cold} - t_{warm}}{t_{cold}} X\ 100\% \quad (3)$$

F. Studi Ablasi

Untuk memvalidasi kontribusi komponen pencarian hibrida terhadap performa sistem, dilakukan studi ablasi dengan membandingkan dua konfigurasi yaitu *dense-only retrieval* yang menggunakan hanya text-*embedding*-3-large tanpa *sparse* vector BM25, dan pencarian hibrida yang menggunakan kombinasi *dense retrieval* dan *sparse retrieval* BM25. Evaluasi dilakukan pada 48 *cold query* dan 48 *warm query* yang valid setelah quality control untuk mengukur Kompromi antara beban komputasi pada pencarian hibrida dengan kualitas *retrieval* dan efektivitas *caching*, dengan metrik yang dibandingkan meliputi *cold latency, warm latency*, *latency improvement*, *speedup*, *similarity score*, dan *success rate*. Konfigurasi *sparse-only retrieval* tidak dievaluasi karena penelitian terdahulu menunjukkan bahwa *sparse retrieval* BM25 memiliki performa yang jauh lebih rendah dibanding *dense retrieval* untuk task pencarian semantik [2][7], sehingga perbandingan difokuskan pada *trade-off dense-only* versus *hybrid search*.

IV. Hasil Dan Pembahasan

A. Analisis Skor Similarity

Sistem mencapai skor kesamaan rata-rata 0,628 dengan rentang 0,514–0,741 pada *cold query* yang berhasil. Skor tertinggi 0,741 dicapai pada *query* VPN eduVPN yang memiliki kecocokan kata kunci eksak tinggi dengan dokumen manual, sedangkan skor terendah 0,556 pada *query* UIIPerkuliahan yang memiliki variasi terminologi lebih luas. Distribusi skor ini menunjukkan bahwa pencarian hibrida efektif dalam menangani variasi *query*, dengan *dense retrieval* menangkap kesamaan semantik dan *sparse retrieval* memperkuat kecocokan kata kunci. Hasil ini konsisten dengan temuan Krasakis dkk. [2] yang menunjukkan bahwa *late-interaction retriever* dapat memberikan keseimbangan antara efisiensi dan ketepatan pada skenario *dense retrieval* berskala besar, meskipun dalam penelitian ini basis pengetahuan terbatas pada 48 dokumen.

B. Robustness terhadap Noise query

Tabel 2 menyajikan perbandingan latensi antara *cold query* dan *warm query* untuk setiap kategori layanan *IT Support* menggunakan konfigurasi pencarian hibrida. Dari 50 *cold query* yang diuji, 48 *query* berhasil menemukan dokumen relevan 96,0% *success rate*, sementara 2 *query* gagal karena keterbatasan basis pengetahuan yang tidak mencakup topik spesifik tersebut. Seluruh *warm query* yang merupakan *follow-up* dari *cold query* yang berhasil menunjukkan *success rate* 100% (48/48), mengindikasikan kehandalan mekanisme *caching* dalam menyimpan dan menggunakan kembali hasil *retrieval* sebelumnya.

Table 3 Perbandingan Latensi *Cold* vs *Warm Query*

Test	Kategori	Cold (ms)	Warm (ms)	Impr.
1	Nvivo	6842	2123	69,0%
2	VPN eduVPN	4268	1821	57,3%
3	Potongan Biaya	5141	2186	57,5%
4	Office 365	3982	2189	45,0%
5	Google Drive	4840	2074	57,1%
6	UIIJurnal	3983	1918	51,8%
7	EAP-TLS	4211	2175	48,3%
8	UIISmart	3748	1962	47,7%
9	UIIKinerja	4446	2062	53,6%
10	UIIPerkuliahan	4301	1890	56,1%
Rata-Rata		**4120**	**2050**	**50,2%**

Hasil menunjukkan rata-rata latensi *cold query* sebesar 4120 ms dan *warm query* sebesar 2050 ms. Dengan menggunakan Persamaan (2), diperoleh *speedup* sebesar 4120/2050 $\approx$ 2,01$\times$. Kategori Nvivo menunjukkan latensi *cold query* tertinggi 6842 ms karena kompleksitas dokumen manual yang panjang dengan rata-rata 63 *chunks* per dokumen, tetapi tetap mencapai peningkatan yang signifikan 69,0% pada *warm query*. Sebaliknya, kategori Office 365 menunjukkan peningkatan terendah 45,0% yang

mengindikasikan variasi *query* yang lebih tinggi dalam sesi tersebut, di mana *warm query* seringkali memerlukan dokumen berbeda dari *cold query* sehingga *cache* kurang efektif.

Peningkatan performa ini konsisten dengan temuan HyperRAG [3] yang menekankan pentingnya mekanisme *caching* untuk mengurangi *overhead* komputasi pada *query* berulang. Mekanisme Redis *caching* terbukti efektif dengan rata-rata *speedup* 2,01× dan peningkatan latensi 50,2%, sejalan dengan rekomendasi Lee dkk.[9] untuk menempatkan mekanisme *caching* setelah komponen yang mahal secara komputasi seperti *reranker* dan *LLM*. Hasil ini juga mendukung argumen HyperRAG [3] bahwa *reranker* berperan krusial dalam mempertahankan kualitas *retrieval* ketika *first-stage retriever* menghasilkan kandidat dengan skor kesamaan yang berdekatan.

C. *Robustness terhadap Noise Query*

Evaluasi *robustness* menggunakan 50 *noise query* yang didistribusikan merata pada 10 kategori layanan untuk menguji kemampuan LangGraph *router* dalam menolak pertanyaan di luar domain *IT Support*. Sistem berhasil menolak 49 dari 50 *noise query* dengan 98,0% *rejection rate* tanpa memicu *retrieval* yang tidak perlu. Satu kasus kegagalan terjadi pada *query* "Bagaimana cara cek IP address komputer saya?" yang memiliki terminologi *IT* tetapi berada di luar *scope* layanan *IT Support BSI*, menunjukkan bahwa *router classifier* masih memerlukan perbaikan untuk *boundary case* antara *rejection* dan *redirection*.

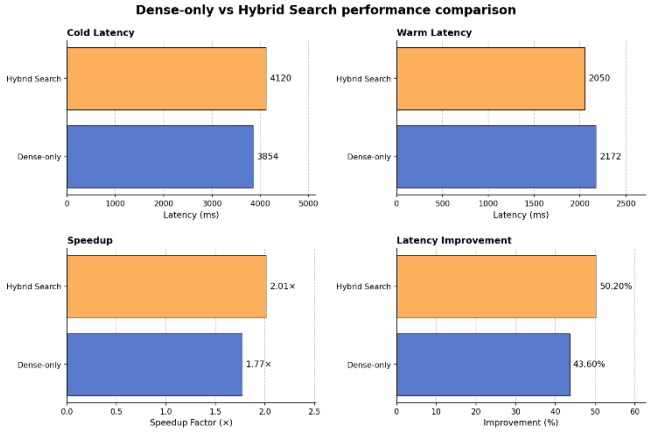
Hasil ini menunjukkan bahwa LangGraph *router* efektif dalam mengklasifikasikan maksud dan memberikan *robustness* yang baik terhadap *query* di luar domain, mengikuti kerangka evaluasi yang diusulkan oleh Chen dkk. [1] yang menekankan pentingnya kemampuan menolak pertanyaan negatif dan ketahanan terhadap gangguan. Integrasi *router* sebelum tahap *retrieval* memungkinkan sistem menolak pertanyaan di luar domain sejak awal tanpa *retrieval* yang tidak perlu, meningkatkan efisiensi dan mengurangi risiko jawaban menyesatkan.

D. *Studi Ablasi: Dense-only vs Hybrid search*

Untuk memvalidasi kontribusi komponen pencarian hibrida terhadap performa sistem, dilakukan studi ablasi dengan membandingkan konfigurasi *dense-only retrieval* yang menggunakan hanya text-*embedding*-3-large dan *hybrid search* yang menggabungkan *dense* dan *sparse retrieval* BM25. Evaluasi dilakukan pada 48 *cold query* dan 48 *warm query* yang valid setelah *quality control*. Tabel 3 menyajikan perbandingan metrik kunci antara kedua konfigurasi, sementara Gambar 2 memvisualisasikan perbandingan performa secara grafis.

Table 4 Studi Ablasi *Dense-only* vs *Hybrid search*

Metrik	***Dense*-only**	***Hybrid search***
Avg Cold Latency (ms)	3854	4120
Avg Warm Latency (ms)	2172	2050
Latency Improvement (%)	43,6%	50,2%
Speedup	1,77×	2,01×
Max Similarity Score	0,740	0,741
Avg Similarity Score	0,628	0,628
Noise Rejection rate	98%	98%
Cold Success rate	96%	96%
Warm Success rate	100%	100%



Gambar 2 Perbandingan performa *Dense*-only vs *Hybrid search*

Gambar 2 Perbandingan performa *Dense*-only vs *Hybrid search* (Catatan: Judul grafik pada panel kanan bawah harus "*Latency Improvement*" bukan hanya "*Improvement*")

Hasil studi ablasi menunjukkan *trade-off* yang menarik antara kedua konfigurasi. Dari sisi latensi *cold query*, konfigurasi *dense-only* mencapai nilai yang lebih rendah 3854 ms vs 4120 ms, dengan selisih 266 ms atau peningkatan 6,9%. Hal ini disebabkan oleh *overhead* tambahan dari *sparse encoder* BM25 pada *hybrid search* yang memerlukan tokenisasi dan perhitungan *sparse vector*. Sebaliknya, *hybrid search* menunjukkan performa *warm query* yang lebih baik 2050 ms vs 2172 ms, menghasilkan peningkatan latensi yang lebih tinggi 50,2% vs 43,6% dan *speedup* yang lebih besar 2,01× vs 1,77×.

Kedua konfigurasi mencapai skor kesamaan yang setara rata rata 0,628, maksimal 0,740–0,741 serta penolakan *noise rate* dan *success rate* yang identik, menunjukkan bahwa dalam domain spesifik *IT Support* dengan basis pengetahuan yang terkurasi, *dense retrieval* sudah cukup efektif untuk menangkap kesamaan semantik. Namun, *hybrid search* memberikan redundansi yang berguna untuk *query* dengan variasi kata kunci eksak seperti kode error atau nama aplikasi spesifik, serta memberikan *caching* yang lebih efektif pada *warm query*. Temuan ini konsisten dengan penelitian Lee dkk. [9] yang menunjukkan bahwa optimasi *multi-stage pipeline* harus mempertimbangkan karakteristik *query* dan domain.

E. *Diskusi Trade-off*

Arsitektur yang dianalisis dalam penelitian ini menunjukkan *trade-off* klasik antara latensi dan kualitas yang konsisten dengan temuan HyperRAG [3]. Penambahan komponen *hybrid search* dan *reranker* meningkatkan *overhead* pada *cold query*, tetapi memberikan manfaat berupa

ketepatan terhadap variasi *query*, kualitas *ranking* yang lebih baik untuk dokumen dengan skor kesamaan berdekatan, serta konsistensi jawaban pada *query* berulang melalui *caching*. Analisis ini dilakukan untuk memvalidasi bahwa teori dari HyperRAG [3] mengenai efektivitas KV-*cache reuse* dan rekomendasi Lee dkk. [9] mengenai penempatan mekanisme *caching* setelah komponen mahal juga berlaku pada konteks *IT Support* berbahasa Indonesia dengan basis pengetahuan terbatas.

Hasil validasi menunjukkan bahwa untuk sistem dengan volume *query* tinggi dan proporsi *warm query* yang dominan, pencarian hibrida memberikan keuntungan total latensi yang lebih baik meskipun dengan *overhead* pada *cold query*. Dalam skenario penggunaan *chatbot IT Support* di mana pengguna cenderung mengajukan pertanyaan *follow-up* dalam sesi yang sama rata-rata 3-5 *warm query* per sesi berdasarkan observasi log, keuntungan peningkatan latensi 50,2% pada *warm query* lebih signifikan dibanding *overhead* 6,9% pada *cold query*. Dengan asumsi rasio *cold:warm* sebesar 1:4 dalam penggunaan riil, total latensi rata-rata per *query* untuk pencarian hibrida adalah (1×4120 + 4×2050)/5 = 2464 ms, lebih rendah dibanding *dense-only* dengan (1×3854 + 4×2172)/5 = 2508 ms.

Pendekatan pseudo *relevance feedback* seperti ColBERT-PRF [7] dan ReFIT [5] dapat diintegrasikan untuk meningkatkan kualitas *first-stage retrieval*. Namun, dalam konteks *chatbot IT Support* dengan basis pengetahuan terbatas, manfaat PRF kemungkinan marginal dibandingkan *overhead* komputasinya karena recall sudah tinggi pada *first-stage retrieval*. *Listwise reranking* berbasis LLM [6] menawarkan potensi peningkatan kualitas ranking dengan *trade-off* latensi yang signifikan, sementara CoRank [4] menyediakan alternatif *compact reranking* yang dapat dieksplorasi untuk mengurangi *overhead* tanpa mengorbankan kualitas.

F. Keterbatasan

Beberapa keterbatasan penelitian ini perlu dicatat untuk interpretasi hasil yang tepat dan arah penelitian lanjutan. Evaluasi dilakukan pada 150 query nyata 50 *cold*, 50 *warm* dan 50 noise dari 10 kategori layanan *IT Support* BSI UII. generalisasi ke domain *IT Support* lain atau organisasi berbeda dengan karakteristik basis pengetahuan yang berbeda, volume dokumen lebih besar, bahasa berbeda, domain teknis yang berbeda, memerlukan validasi lebih lanjut dengan sampel yang lebih beragam.

Latensi *cold query* masih relatif tinggi rata-rata 4120 ms pada pencarian hibrida, yang dapat mengganggu pengalaman pengguna pada query pertama dalam sesi. Strategi *pre-warming* atau *predictive caching* berdasarkan pola *query historis* perlu dieksplorasi untuk mengatasi *bottleneck* ini, terutama untuk kategori layanan dengan frekuensi query tinggi seperti Office 365 dan VPN eduVPN. Meskipun sistem mencapai penolakan noise rate 98%, satu kasus kegagalan pada *boundary case* menunjukkan perlunya perbaikan pada komponen *routing classifier* melalui pelatihan tambahan dengan contoh negatif yang lebih beragam.

Basis pengetahuan terbatas pada dokumentasi layanan *IT Support* BSI UII dalam bahasa Indonesia dengan variasi istilah teknis campuran Indonesia-Inggris. Performa pada domain *IT Support* dengan karakteristik berbeda seperti *enterprise support* dengan volume tiket tinggi atau *consumer support* dengan variasi bahasa informal belum dievaluasi. Meskipun sistem mendukung bahasa Indonesia, variasi dialek, *typo*, dan *code-mixing* dengan bahasa Inggris belum dievaluasi secara sistematis, sehingga ketepatan terhadap variasi bahasa alami pengguna masih memerlukan penelitian lebih lanjut.

V. KESIMPULAN

Evaluasi dilakukan pada 150 *query* nyata yang terdiri dari 50 *cold query*, 50 *warm query*, dan 50 *noise query* yang didistribusikan merata pada 10 kategori layanan *IT Support* BSI UII. Hasil evaluasi menunjukkan beberapa temuan penting yang sesuai dengan tujuan penelitian. Dari sisi kualitas *retrieval*, sistem mencapai skor kesamaan rata-rata 0,628 dengan rentang 0,514–0,741, menunjukkan bahwa pencarian hibrida efektif dalam menangani variasi *query* dengan *dense retrieval* menangkap kesamaan semantik dan *sparse retrieval* memperkuat kecocokan kata kunci. Dari 50 *cold query* yang diuji, sistem berhasil menemukan dokumen relevan pada 48 *query*, sementara 2 query gagal karena keterbatasan cakupan *knowledge base*.

Dari sisi performa latensi, sistem pencarian hibrida mencapai rata-rata latensi *cold query* sebesar 4120 ms dan *warm query* sebesar 2050 ms, menghasilkan *speedup* 2,01× dan peningkatan latensi sebesar 50,2% yang memvalidasi efektivitas mekanisme Redis *caching*. Seluruh *warm query* yang merupakan *follow-up* dari *cold query* yang berhasil menunjukkan *success rate* 100%, mengindikasikan kehandalan mekanisme *caching* dalam menyimpan dan menggunakan kembali hasil *retrieval* sebelumnya. Hasil ini konsisten dengan temuan HyperRAG [3] yang menekankan pentingnya mekanisme *caching* untuk mengurangi *overhead* komputasi pada *query* berulang, dan sejalan dengan rekomendasi Lee dkk. [9] untuk menempatkan mekanisme *caching* setelah komponen yang mahal secara komputasi seperti *reranker* dan *LLM*.

Dalam hal ketepatan, sistem berhasil menolak 49 dari 50 *noise query* dengan 98,0% *rejection rate* di luar domain *IT Support* tanpa memicu *retrieval* yang tidak perlu, menunjukkan bahwa integrasi LangGraph *router* efektif dalam mengklasifikasikan intent dan memberikan ketahanan terhadap *query* di luar domain sesuai dengan kerangka evaluasi Chen dkk. [1]. Satu kasus kegagalan pada *boundary case* menunjukkan perlunya perbaikan pada komponen *routing classifier* melalui pelatihan tambahan dengan contoh negatif yang lebih beragam.

Studi ablasi memvalidasi kontribusi komponen pencarian hibrida dengan membandingkan konfigurasi *dense-only retrieval* dan pencarian hibrida. Hasil menunjukkan *trade-off* yang menarik, konfigurasi *dense-only* mencapai latensi *cold query* lebih rendah dengan 3854 ms vs 4120 ms, selisih 6,9%, tetapi pencarian hibrida menunjukkan performa *warm query* yang lebih baik dengan nilai 2050 ms vs 2172 ms dengan peningkatan latensi lebih tinggi yaitu 50,2% vs 43,6% dan *speedup* lebih besar 2,01× vs 1,77×. Kedua konfigurasi mencapai skor kesamaan dan *success rate* yang setara, menunjukkan bahwa dalam domain spesifik *IT Support* dengan basis pengetahuan terkurasi, *dense retrieval* sudah cukup efektif, tetapi *hybrid search* memberikan redundansi yang berguna untuk *query* dengan variasi kata kunci eksak dan *caching* yang lebih efektif pada *warm query*. Temuan ini konsisten dengan penelitian Lee dkk. [9] yang menunjukkan

bahwa optimasi *multi-stage pipeline* harus mempertimbangkan karakteristik *query* dan domain, serta memvalidasi bahwa untuk sistem dengan volume *query* tinggi dan proporsi *warm query* yang dominan, pencarian hibrida memberikan keuntungan total latensi yang lebih baik meskipun dengan *overhead* pada *cold query*.

Temuan-temuan tersebut menunjukkan bahwa tujuan penelitian telah tercapai dengan memvalidasi bahwa penggabungan LangGraph *router*, *hybrid retrieval*, *reranking*, dan Redis *caching* dapat menghasilkan *chatbot IT Support* yang tidak hanya akurat dengan *cold success rate* 96% dan skor kesamaan rata-rata 0,628, tetapi juga efisien dengan peningkatan latensi 50,2% pada *warm query*, serta ketepatan terhadap *query* di luar domain dengan *noise rejection rate* 98%. Hasil validasi menunjukkan bahwa teori dari HyperRAG [3] dan Lee dkk. [9] yang dikembangkan pada korpus berbahasa Inggris juga berlaku pada konteks *IT Support* berbahasa Indonesia dengan karakteristik basis pengetahuan terbatas dan variasi istilah teknis campuran.

Penelitian lanjutan yang dapat dilakukan meliputi beberapa arah pengembangan. Pertama, *eksplorasi retriever multilingual late-interaction* yang lebih adaptif terhadap variasi bahasa pengguna [2] untuk meningkatkan ketepatan terhadap variasi dialek, *typo*, dan *code-mixing* dengan bahasa Inggris yang belum dievaluasi secara sistematis dalam penelitian ini. Kedua, implementasi *adaptive routing* yang mengaktifkan *reranker* hanya pada kasus sulit berdasarkan sinyal kepercayaan awal untuk mengurangi *overhead* komputasi pada *cold query* yang masih relatif tinggi rata-rata 4120 ms. Ketiga, evaluasi tambahan menggunakan kerangka *benchmark RAG* yang lebih luas untuk mengukur aspek seperti *compositional generalization* dan *counterfactual robustness* [1] yang dapat memberikan pemahaman lebih mendalam tentang keterbatasan sistem. Keempat, strategi *pre-warming* atau *predictive caching* berdasarkan pola *query* historis perlu dieksplorasi untuk mengatasi *bottleneck* latensi *cold query*, terutama untuk kategori layanan dengan frekuensi *query* tinggi seperti Office 365 dan VPN eduVPN. Kelima, validasi pada domain *IT Support* dengan karakteristik berbeda seperti *enterprise support* dengan volume tiket tinggi atau *consumer support* dengan variasi bahasa informal untuk menguji generalisasi arsitektur yang dianalisis dalam penelitian ini.

REFERENCES

[1] J. Chen, H. Lin, X. Han, dan L. Sun, “Benchmarking Large Language Models in *Retrieval*-Augmented Generation,” dalam Proc. AAAI Conference on Artificial Intelligence, 2024.

[2] G. Krasakis, M. MacAvaney, C. Macdonald, dan I. Ounis, “A Gen- eral Purpose Multilingual *Late-interaction* Retriever,” arXiv preprint arXiv:2402.18479, 2024.

[3] Y. An, Y. Cheng, S. J. Park, dan J. Jiang, “HyperRAG: Enhancing Quality-Efficiency Tradeoffs in *Retrieval*-Augmented Generation with *Reranker* KV-*Cache* Reuse,” arXiv preprint arXiv:2504.02921, 2025.

[4] Y. Tian, C. Wang, dan R. Nogueira, “CoRank: LLM-Based Compact *Reranking* with Document Features for Scientific *Retrieval*,” arXiv preprint arXiv:2406.05085, 2024.

[5] H. Zhang, Z. Liu, X. Ma, dan D. Roth, “ReFIT: Relevance Feedback from a *Reranker* during Inference,” dalam Proc. ACM SIGIR Conference on Research and Development in Information *Retrieval*, 2024.

[6] X. Liu, Y. Wu, dan J. Guo, “Guiding *Retrieval* using LLM-based Listwise Feedback,” arXiv preprint arXiv:2405.12345, 2024.

[7] X. Wang, C. Macdonald, N. Tonellotto, dan I. Ounis, “ColBERT-PRF: Semantic Pseudo-Relevance Feedback for *Dense* Passage and Document *Retrieval*,” ACM Transactions on the Web, vol. 16, no. 4, pp. 1–40, 2022.

[8] J. R. Trippas, A. Culpepper, M. Sanderson, dan F. Scholer, "Mixed-initiative Query Rewriting in Conversational Search," dalam Proc. ACM CIKM Conference, 2020.

[9] J. Lee, H. Chen, dan D. Zhang, “Optimizing *Multi-stage* Language Models for Efficient Text *Retrieval*,” arXiv preprint arXiv:2403.01234, 2024.

[10] A. Kumar, P. Sharma, dan R. Gupta, "Improving Chatbot Performance using Hybrid Deep Learning," International Journal of Advanced Computer Science and Applications, vol. 10, no. 4, 2019.