Analisis Pengaruh Metode *Chunking* terhadap Kinerja *Retrieval-Augmented Generation* pada Dokumen Publikasi Badan Pusat Statistik

SKRIPSI

Disusun oleh:
Haikal Thoriq Athaya
NIM: 225150200111004

PROGRAM STUDI TEKNIK INFORMATIKA
DEPARTEMEMEN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS B*RAWI*JAYA
MALANG
2025

PENGESAHAN

Judul ini digunakan untuk dokumen final yang elah direvisi/disetujui setelah kelulusan ujian.

Untuk pendaftaran semhas dan ujian skripsi, judul halaman ini adalah **PERSETUJUAN.**

JUDUL SKRIPSI

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :
Nama Mahasiswa
NIM: 123456789

Bagian ini hanya digunakan jika halaman ini adalah halaman **PENGESAHAN.** Jika ini adalah halaman PERSETUJUAN, maka bagian ini tidak diperlukan.

Skripsi ini telah diuji dan dinyatakan lulus pada
2 Januari 2015
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Nama Dosen Pembimbing 1 NIK: 123456789 /*jika terdapat NIK saja*/	Nama Dosen Pembimbing 2 NIK: - /*jika tidak terdapat NIP, NIK, atau keduanya*/

Mengetahui
Ketua Jurusan **Nama Jurusan**

Contoh: Ketua Jurusan **Teknik Informatika**

Nama Ketua Jurusan
NIP: 123456789
/*jika terdapat NIP*/

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 1 Januari 2015

Nama Mahasiswa

NIM: 123456789

PRAKATA

Bagian ini memuat pernyataan resmi untuk menyampaikan rasa terima kasih penulis kepada berbagai pihak yang telah membantu penyelesaian skripsi ini. Nama-nama penerima ucapan terima kasih sebaiknya dituliskan lengkap, termasuk gelar akademik, dan pihak-pihak yang tidak terkait dihindari untuk dituliskan. Bahasa yang digunakan seharusnya mengikuti kaidah bahasa Indonesia yang baku. Prakata boleh diakhiri dengan paragraf yang menyatakan bahwa penulis menerima kritik dan saran untuk pengembangan penelitian selanjutnya. Terakhir, prakata ditutup dengan mencantumkan kota dan tanggal penulisan prakata, lalu diikuti dengan kata "Penulis".

Malang, 1 Januari 2015

Penulis

email@domain.com

ABSTRAK

Nama Mahasiswa, Judul Skripsi

Pembimbing: Nama Pembimbing 1 dan Nama Pembimbing 2

Bagian ini diisi dengan abstrak dalam Bahasa Indonesia. Abstrak adalah uraian singkat (umumnya 200-300 kata) yang merupakan intisari dari sebuah skripsi. Abstrak membantu pembaca untuk mendapatkan gambaran secara cepat dan akurat tentang isi dari sebuah skripsi. Melalui abstrak, pembaca juga dapat menentukan apakah akan membaca skripsi lebih lanjut. Oleh karena itu, abstrak sebaiknya memberikan gambaran yang padat tetapi tetap jelas dan akurat tentang (1) apa dan mengapa penelitian dikerjakan: sedikit latar belakang, pertanyaan atau masalah penelitan, dan/atau tujuan penelitian; (2) bagaimana penelitian dikerjakan: rancangan penelitian dan metodologi/metode dasar yang digunakan dalam penelitian; (3) hasil penting yang diperoleh: temuan utama, karakteristik artefak, atau hasil evaluasi artefak yang dibangun; (4) hasil pembahasan dan kesimpulan: hasil dari analisis dan pembahasan temuan atau evaluasi artefak yang dibangun, yang dikaitkan dengan pertanyaan/tujuan penelitian.

Yang harus dihindari dalam sebuah abstrak diantaranya (1) penjelasan latar belakang yang terlalu panjang; (2) sitasi ke pustaka lainnya; (3) kalimat yang tidak lengkap; (3) singkatan, jargon, atau istilah yang membingungkan pembaca, kecuali telah dijelaskan dengan baik; (4) gambar atau tabel; (5) angka-angka yang terlalu banyak.

Di akhir abstrak ditampilkan beberapa kata kunci (normalnya 5-7) untuk membantu pembaca memposisikan isi skripsi dengan area studi dan masalah penelitian. Kata kunci, beserta judul, nama penulis, dan abstrak biasanya dimasukkan dalam basis data perpustakaan. Kata kunci juga dapat diindeks dalam basis data sehingga dapat digunakan untuk proses pencarian tulisan ilmiah yang relevan. Oleh karena itu pemilihan kata kunci yang sesuai dengan area penelitian dan masalah penelitian cukup penting. Pemilihan kata kunci juga bisa didapatkan dari referensi yang dirujuk.

Kata kunci: abstrak, skripsi, intisari, kata kunci, artefak

ABSTRACT

Student Name, Skripsi Title

Supervisors: First Supervisor's Name and Second Supervisor's Name

The absract of your skripsi in English is written here.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Retrieval-Augmented Generation (RAG) telah menjadi paradigma penting dalam pengembangan sistem kecerdasan buatan modern karena kemampuannya mengintegrasikan pengetahuan eksternal dengan kemampuan generatif *Large Language Models* (LLMs) (Fan et al., 2024). Pendekatan ini terbukti efektif dalam mengatasi keterbatasan LLM konvensional seperti halusinasi dan pengetahuan yang usang, melalui penyediaan akses ke basis pengetahuan yang dapat diperbarui secara dinamis (Gargari and Habibi, 2025). Keunggulan tersebut menjadikan RAG salah satu solusi utama untuk meningkatkan akurasi dan kredibilitas sistem berbasis bahasa alami yang memerlukan pengetahuan terkini.

Di balik performa unggul RAG, terdapat tahap *preprocessing* fundamental yang memiliki pengaruh besar terhadap efektivitas sistem, yaitu *chunking*. Proses *chunking* melibatkan pemecahan dokumen besar menjadi segmen-segmen kecil berukuran tetap agar *retriever* dapat memproses informasi secara lebih fokus dan efisien. Menentukan strategi *chunking* yang ideal menjadi tantangan penting karena hasil penelitian menunjukkan bahwa variasi ukuran dan teknik *chunking* berpengaruh signifikan terhadap kinerja sistem RAG, baik dari segi *retrieval accuracy* maupun *generation quality* (Aadit Kshirsagar, 2024). Dengan demikian, analisis terhadap metode *chunking* menjadi aspek penting dalam upaya optimalisasi sistem RAG.

Kompleksitas dokumen statistik semakin memperkuat urgensi penelitian ini karena jenis dokumen tersebut memiliki karakteristik unik berupa struktur hierarkis, tabel numerik, serta terminologi yang sangat spesifik terhadap domain tertentu (Sarmah et al., 2024). Kondisi ini menimbulkan tantangan dalam proses pemahaman konteks dan ekstraksi informasi yang akurat. Secara khusus, dokumen publikasi Badan Pusat Statistik (BPS) sebagai sumber utama informasi statistik nasional menuntut pendekatan pemrosesan yang mampu menangani keterkaitan antarbagian serta kompleksitas semantik di dalamnya. Oleh karena itu, penerapan sistem RAG pada dokumen BPS memerlukan strategi *chunking* yang adaptif dan kontekstual agar informasi yang dihasilkan tetap relevan dan koheren.

Meskipun berbagai penelitian telah mengeksplorasi optimalisasi RAG dalam beragam domain seperti kesehatan (Jeong et al., 2024), keuangan (Sarmah et al., 2024), dan hukum (Ajay Mukund and Easwarakumar, 2025), masih terdapat kesenjangan dalam memahami pengaruh spesifik metode *chunking* terhadap kinerja RAG pada dokumen statistik kompleks. Penelitian terdahulu menunjukkan bahwa "strategi *chunking* yang berbeda melayani kasus penggunaan yang berbeda" (Aadit Kshirsagar, 2024), namun belum ada kajian komprehensif yang secara khusus mengevaluasi dampak variasi teknik *chunking* pada dokumen publikasi statistik. Optimalisasi sistem RAG untuk dokumen BPS juga berpotensi memberikan kontribusi praktis dalam meningkatkan aksesibilitas dan

pemanfaatan informasi statistik nasional bagi peneliti, pembuat kebijakan, dan masyarakat luas.

Kesenjangan penelitian tersebut menjadi semakin penting untuk diatasi mengingat sistem RAG masih menghadapi tujuh titik kegagalan utama dalam implementasinya (Barnett et al., 2024), di mana salah satunya berkaitan dengan efisiensi dan ketepatan pengelolaan konteks. Oleh karena itu, penelitian ini bertujuan untuk menganalisis secara sistematis pengaruh metode *chunking* terhadap kinerja sistem RAG dalam konteks dokumen publikasi BPS. Hasil penelitian ini diharapkan memberikan kontribusi ilmiah dan praktis bagi pengembangan sistem RAG yang lebih efektif, efisien, dan kontekstual untuk mendukung pemanfaatan data statistik pemerintah di Indonesia.

1.2 Rumusan Masalah

1. Bagaimana pengaruh perbedaan metode *chunking* terhadap kinerja *retrieval* yang diukur menggunakan metrik *Precision*@k, *Recall*@k, dan *Mean Reciprocal Rank* (MRR)?
2. Bagaimana pengaruh perbedaan metode *chunking* terhadap kualitas hasil generasi yang diukur menggunakan metrik BLEU dan ROUGE-L?
3. Metode *chunking* mana yang memberikan performa terbaik berdasarkan gabungan metrik *retrieval* dan metrik generasi?

1.3 Tujuan

1. Menganalisis pengaruh perbedaan metode *chunking* terhadap kinerja *retrieval* dalam sistem RAG menggunakan metrik *Precision*@k, *Recall*@k, dan *Mean Reciprocal Rank* (MRR).Menganalisis pengaruh perbedaan metode *chunking* terhadap kinerja *retrieval* pada sistem RAG.
2. Menganalisis pengaruh setiap metode chunking terhadap kualitas generasi jawaban yang dievaluasi menggunakan metrik BLEU dan ROUGE-L.
3. Membandingkan performa keseluruhan ketiga metode *chunking* berdasarkan gabungan indikator kinerja *retrieval*, kualitas generasi, serta efisiensi waktu proses, guna menentukan metode *chunking* yang paling optimal untuk dokumen publikasi BPS.

1.4 Manfaat

1.4.1 Manfaat Akademis

Penelitian ini berkontribusi dalam pengembangan kajian ilmiah di bidang *Natural Language Processing* (NLP) dan *Retrieval-Augmented Generation (RAG)*, khususnya dalam memahami pengaruh metode *chunking* terhadap kinerja sistem berbasis *Large Language Models (LLMs)*. Hasil penelitian ini dapat menjadi referensi bagi peneliti selanjutnya untuk mengoptimalkan proses *document preprocessing* dan meningkatkan efektivitas sistem RAG dalam berbagai domain dokumen yang kompleks.

1.4.2 Manfaat Praktis

Hasil penelitian ini diharapkan dapat memberikan rekomendasi praktis bagi pengembang sistem kecerdasan buatan dan lembaga pemerintah, khususnya Badan Pusat Statistik (BPS), dalam penerapan teknologi RAG untuk meningkatkan aksesibilitas, relevansi, dan efisiensi pemanfaatan dokumen publikasi statistik nasional. Selain itu, penelitian ini juga dapat membantu pengambil kebijakan, peneliti, dan masyarakat umum dalam memperoleh informasi statistik yang lebih akurat dan kontekstual melalui sistem pencarian berbasis AI.

1.5 Batasan Masalah

Agar penelitian ini memiliki ruang lingkup yang terarah dan fokus pada permasalahan utama, maka batasan masalah dalam penelitian ini ditetapkan sebagai berikut:

1.5.1 Cakupan Penelitian

Penelitian ini berfokus pada tahap *preprocessing* dalam sistem *Retrieval-Augmented Generation* (RAG), khususnya pada analisis pengaruh metode *chunking* terhadap kinerja sistem. Komponen lain seperti *retriever, reranker*, dan *generator* digunakan sebagaimana standar *pipeline* tanpa dilakukan modifikasi mendalam.

1.5.2 Metode *Chunking* yang Digunakan

Penelitian ini membandingkan tiga metode *chunking* berikut:

1. *Element-Based Chunking*, yang membagi dokumen berdasarkan struktur dan elemen format (misalnya paragraf, subjudul, atau tabel) untuk mempertahankan keteraturan isi.
2. *Max-Min Semantic Chunking*, yang mengelompokkan teks berdasarkan kesamaan semantik dengan algoritma *Max-Min,* menghasilkan segmen dengan kesetimbangan konteks tinggi.
3. *Recursive Chunking*, yang memecah teks secara bertahap berdasarkan struktur hierarkis seperti paragraf, kalimat, dan spasi. Metode ini mempertahankan keseimbangan antara ukuran potongan teks dan kelengkapan konteks, sehingga sering digunakan sebagai *baseline* dalam sistem (RAG) untuk memastikan proses pemotongan teks yang efisien tanpa kehilangan makna utama.

1.5.3 Model Bahasa dan Lingkungan Eksperimen

Model yang digunakan adalah Qwen3-8B sebagai *Large Language Model* (LLM) utama dan Qwen3-*Embedding*-8B untuk proses *vectorization*. Pemilihan model ini mempertimbangkan dukungan *native* terhadap bahasa Indonesia dan performa dalam *semantic retrieval*. Seluruh eksperimen dijalankan pada lingkungan lokal dengan spesifikasi perangkat keras yang memadai.

1.5.4 *Dataset* Penelitian

Dataset terdiri atas dokumen publikasi resmi Badan Pusat Statistik (BPS) yang berbentuk teks atau hasil konversi dari PDF. Dokumen yang digunakan memiliki kombinasi teks naratif dan tabel statistik sederhana.

Elemen non-teks seperti grafik, peta, atau infografis tidak diproses secara langsung karena *pipeline* RAG yang digunakan beroperasi berbasis teks. Namun, keberadaan elemen visual tetap diperhitungkan sebagai konteks struktural dokumen.

1.5.5 Evaluasi Kinerja Sistem

Evaluasi difokuskan pada dua komponen utama:

1. *Retrieval performance* diukur dengan metrik *Precision*@k, *Recall*@k, dan *Mean Reciprocal Rank* (MRR);
2. *Generation performance* diukur dengan *BLEU* dan ROUGE-L.
3. Analisis hasil dilakukan secara kuantitatif untuk menilai perbedaan performa antar metode *chunking*.

1.5.6 Bahasa dan Konteks Pengujian

Seluruh eksperimen dilakukan dalam konteks bahasa Indonesia untuk menjaga kesesuaian dengan isi dokumen publikasi BPS dan kemampuan pemahaman model Qwen3.

1.6 Sistematika Pembahasan

Sistematika pembahasan dalam skripsi ini terdiri dari enam bab yang tersusun secara sistematis agar memudahkan pembaca dalam memahami isi dan alur penelitian. Adapun susunan bab dalam skripsi ini adalah sebagai berikut:

1.6.1 Bab I Pendahuluan

Bab ini menjelaskan latar belakang penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika pembahasan. Bagian ini memberikan gambaran umum mengenai dasar pemikiran dan arah penelitian yang dilakukan.

1.6.2 Bab II Landasan Kepustakaan

Bab ini memuat teori dan konsep yang menjadi dasar dalam penelitian, meliputi pembahasan mengenai *Large Language Models* (LLMs), *Retrieval-Augmented Generation* (RAG), metode *chunking*, serta metrik evaluasi kinerja RAG. Selain itu, bab ini juga mencakup kajian penelitian terdahulu yang relevan serta kesenjangan penelitian yang melatarbelakangi studi ini.

1.6.3 Bab III Metodologi Penelitian

Bab ini menjelaskan pendekatan dan tahapan penelitian, mulai dari rancangan penelitian, sumber dan karakteristik data, hingga penjelasan sistem *Retrieval-*

Augmented Generation (RAG) yang digunakan. Bab ini juga menguraikan penerapan tiga metode *chunking* yang dibandingkan, yaitu *Element-Based Chunking*, *Max-Min Semantic Chunking*, dan *Recursive Chunking*. Model bahasa yang digunakan, alat bantu, serta metrik evaluasi juga dijelaskan secara rinci di bab ini.

1.6.4 Bab IV Perancangan

Bab ini berisi rancangan sistem yang dikembangkan, mencakup arsitektur sistem, alur proses *pipeline*, rancangan struktur *chunking*, dan rancangan penyimpanan *vector database*. Bab ini juga menampilkan diagram alur sistem, diagram proses, serta penjelasan komponen utama dalam *pipeline* RAG yang akan diimplementasikan.

1.6.5 Bab V Implementasi

Bab ini menjelaskan proses implementasi rancangan sistem ke dalam bentuk nyata, termasuk penerapan *pipeline* RAG menggunakan ketiga metode *chunking*, pembuatan *embedding*, serta integrasi model generator dan *vector database*. Bab ini juga membahas konfigurasi perangkat lunak, lingkungan pengujian, serta hasil uji coba awal sistem.

1.6.6 Bab VI Hasil dan Pembahasan

Bab ini menyajikan hasil pengujian sistem RAG serta analisis performa dari setiap metode *chunking* berdasarkan metrik evaluasi yang telah ditentukan. Pembahasan difokuskan pada perbandingan hasil *retrieval* dan *generation* menggunakan metrik *Precision*@k, *Recall*@k, *Mean Reciprocal Rank* (MRR), BLEU, dan ROUGE-L. Bab ini juga menyoroti temuan utama, keterbatasan penelitian, serta interpretasi hasil terhadap efektivitas metode *chunking* yang diuji.

1.6.7 Bab VII Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil penelitian yang dilakukan serta saran untuk penelitian selanjutnya. Kesimpulan disusun berdasarkan hasil analisis terhadap pengaruh metode *chunking* terhadap kinerja sistem RAG pada dokumen publikasi Badan Pusat Statistik (BPS). Saran yang diberikan diharapkan dapat menjadi masukan bagi pengembangan penelitian dan penerapan RAG di masa mendatang.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Penelitian Terdahulu

Penelitian terdahulu berperan penting dalam memberikan landasan teoretis dan arah pengembangan penelitian ini, khususnya terkait *Retrieval-Augmented Generation* (RAG), metode *chunking*, dan penerapan *Large Language Models* (LLMs) dalam pemrosesan dokumen kompleks. Kajian literatur sebelumnya membantu memperkuat pemahaman terhadap konsep RAG sekaligus mengidentifikasi kesenjangan riset yang masih terbuka, terutama dalam penerapan RAG pada dokumen semi-terstruktur seperti publikasi Badan Pusat Statistik (BPS). Ringkasan hasil penelitian yang relevan disajikan pada Tabel 2.1 untuk memberikan gambaran menyeluruh mengenai perkembangan dan kontribusi penelitian sebelumnya terhadap studi ini.

Tabel 2.1 Penelitian Terdahulu

No	Peneliti dan Tahun	Judul	Metode	Temuan
1	(Gao et al., 2024)	*Retrieval-Augmented Generation for Large Language Models: A Survey*	Studi survei literatur terhadap perkembangan arsitektur dan paradigma RAG (Naive, Advanced, Modular)	Menjelaskan bahwa RAG efektif mengurangi halusinasi dan meningkatkan relevansi dengan menggabungkan *retrieval* eksternal dan LLM generatif
2	(Fan et al., 2024)	*A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models*	Survei sistematis terhadap integrasi RAG dan LLM (RA-LLMs) mencakup arsitektur, strategi pelatihan, dan aplikasi	RAG terbukti meningkatkan kualitas keluaran LLM melalui integrasi *retrieval*, augmentation, dan *generation* yang adaptif
3	(Aadit Kshirsagar, 2024)	*Evaluating Chunking Strategies for Retrieval-Augmented*	Eksperimen perbandingan metode *chunking*: *fixed-size*, *overlap*ping,	Ukuran dan strategi *chunking* sangat memengaruhi efisiensi *retrieval* dan kualitas

		Generation Systems	*semantic*, *recursive*	*generation* dalam sistem RAG
4	(Sarmah et al., 2024)	*HybridRAG: Integrating Knowledge Graphs and Vector Retrieval-Augmented Generation for Efficient Information Extraction*	Eksperimen integrasi *knowledge graph* dengan vektor *retrieval*	Pendekatan HybridRAG meningkatkan efisiensi pencarian dan akurasi konteks dibanding RAG konvensional
5	(Jeong et al., 2024)	*Improving Medical Reasoning through Retrieval and Self-Reflection with RAG-LLMs*	Eksperimen penerapan RAG dalam domain medis	RAG dengan mekanisme *self-reflection* meningkatkan kemampuan *reasoning* dan akurasi diagnosis berbasis teks medis
6	(Ajay Mukund and Easwarakumar, 2025)	*Legal Document Reasoning with Retrieval-Augmented Generation*	Studi penerapan RAG dalam dokumen hukum berbasis *knowledge graph*	RAG memperkuat *reasoning* hukum dan koherensi antar teks yuridis
7	(Murtiyoso, Tahyudin and Berlilana, 2025)	*Comparative Analysis of RAG Implementation Models in Multilingual Contexts*	Eksperimen komparatif RAG multibahasa menggunakan berbagai corpus	RAG efektif memperkuat pemahaman lintas bahasa dan menjaga akurasi semantik
8	(Anassai and Josaphat, 2024)	*Pembangunan Chatbot Sistem Informasi KBLI dan KBJI Berbasis LLM*	Metode prototyping dengan integrasi RAG dan FAISS untuk sistem klasifikasi KBLI/KBJI BPS	Chatbot berbasis LLM dan RAG mencapai 97% akurasi dengan *hallucination rate* 0% pada dokumen semi-terstruktur BPS

Berdasarkan Tabel 2.1, terlihat bahwa mayoritas penelitian terdahulu berfokus pada pengembangan arsitektur RAG, optimisasi mekanisme *retrieval*, dan penerapan metode *chunking* di berbagai domain seperti medis, hukum, dan

keuangan. Namun, belum ditemukan penelitian yang secara spesifik menganalisis pengaruh metode *chunking* terhadap kinerja RAG pada dokumen publikasi BPS yang bersifat semi-terstruktur. Celah inilah yang menjadi dasar utama penelitian ini, yaitu untuk mengkaji secara empiris bagaimana variasi metode *chunking* dapat memengaruhi efektivitas *retrieval* dan kualitas hasil generasi dalam konteks dokumen publikasi statistik nasional.

2.2 *Large Language Models* (LLMs)

LLM dibangun berdasarkan rancangan transformer yang memungkinkan model untuk memproses urutan teks secara efisien melalui mekanisme *self-attention* (Matarazzo and Torlone, 2025). Mekanisme ini membuat model mampu mengenali hubungan antarkata dan memahami konteks dalam rentang teks yang panjang. Dengan pendekatan tersebut, LLM dapat menangkap makna semantik yang lebih kaya dibandingkan model-model sebelumnya yang berbasis recurrent atau convolutional. Dalam konteks ini, kemampuan untuk memahami konteks (*context* understanding) dan menghasilkan teks baru (*text generation*) menjadikan LLM sangat relevan untuk sistem *Retrieval-Augmented Generation* (RAG), di mana model berperan sebagai *generator* yang memanfaatkan informasi dari proses *retrieval* eksternal (Mikulic et al., 2025).

2.2.1 Qwen3-8B

Qwen3-8B merupakan *Large Language Model* (LLM) yang dikembangkan oleh Alibaba Cloud sebagai bagian dari seri Qwen3 dengan fokus pada peningkatan performa, efisiensi komputasi, dan kemampuan multibahasa (Yang et al., 2025). Model ini menggunakan arsitektur transformer modern dengan dua mode operasi, yaitu *thinking* mode untuk penalaran kompleks multi-langkah dan non-*thinking* mode untuk respons cepat berbasis konteks.

Secara teknis, Qwen3-8B memiliki delapan miliar parameter dengan arsitektur *dense* yang mengintegrasikan Grouped *Query Attention* (GQA), SwiGLU, Rotary Positional *Embedding*s (RoPE), dan RMSNorm dengan pre-normalization, serta QK-Norm untuk menjaga stabilitas pelatihan (Yang et al., 2025). Kombinasi ini memastikan model efisien secara komputasi sekaligus konsisten dalam menghasilkan keluaran pada berbagai tugas bahasa alami. Keunggulan utama Qwen3-8B terletak pada kemampuannya menangani hingga 119 bahasa dan dialek, peningkatan besar dibandingkan seri sebelumnya yang hanya mendukung 29 bahasa, dengan performa unggul pada tugas-tugas STEM dan pemrograman.

Berdasarkan dokumentasi teknis, performa bahasa Indonesia pada Qwen3-8B juga menunjukkan peningkatan signifikan dibandingkan versi sebelumnya (Yang et al., 2025). Model ini dilatih menggunakan *dataset* multibahasa yang mencakup 119 bahasa dan dialek, termasuk bahasa Indonesia, dengan total 36 triliun *token*s yang memberikan cakupan linguistik luas. Secara khusus, Qwen3-8B menunjukkan kemampuan kuat dalam tugas multibahasa seperti instruction following, knowledge assessment, mathematics, dan logical *reasoning*.

2.3 *Retrieval-Augmented Generation* (RAG)

2.3.1 Definisi dan Konsep Dasar RAG

Retrieval-Augmented Generation (RAG) adalah arsitektur sistem kecerdasan buatan yang menggabungkan proses *retrieval* dengan *generation*. Dalam konteks tersebut, RAG hadir sebagai solusi yang memungkinkan integrasi pengetahuan eksternal ke dalam proses generatif, sehingga sistem dapat menghasilkan keluaran yang lebih faktual, relevan, dan kontekstual. Komponen *retriever* bertugas menemukan serta mengambil informasi yang relevan dari basis data eksternal berdasarkan masukan pengguna, sedangkan *generator* memanfaatkan hasil pencarian tersebut untuk menghasilkan respons yang koheren dan sesuai konteks (Oche et al., 2025).

2.3.2 *Retriever*

Komponen *retriever* berfungsi mencari dan mengambil informasi relevan dari sumber pengetahuan berdasarkan *query* yang diberikan oleh pengguna (Gao et al., 2024). Proses ini dilakukan melalui mekanisme pencarian kesamaan (*similarity search)* yang dapat menggunakan pendekatan berbasis kata kunci maupun semantik.

2.3.3 *Embedding* Model

Embedding model berperan dalam mengubah teks menjadi representasi vektor numerik yang merepresentasikan makna semantik dari teks tersebut (Gao et al., 2024). Model ini mengonversi baik *query* pengguna maupun dokumen ke dalam ruang vektor berdimensi tinggi, sehingga sistem dapat melakukan pencarian berdasarkan kesamaan makna, bukan hanya kesamaan kata. Dengan kualitas *embedding* yang baik, sistem RAG dapat mengidentifikasi konteks yang relevan secara lebih akurat dan efisien.

2.3.3.1 Qwen3-8B Embedding Model

Selain *model* generatif, seri Qwen3 juga mencakup Qwen3 *Embedding Series* yang dirancang untuk menghasilkan representasi teks berkualitas tinggi pada berbagai bahasa dan domain. Seri ini memegang peranan penting dalam sistem RAG karena berfungsi sebagai komponen *retriever* yang mengonversi teks menjadi vektor semantik (*embedding*) yang dapat dibandingkan secara efisien. Menurut (Zhang et al., 2025), Qwen3-8B *Embedding* mencapai kinerja *state-of-the-art* pada *Massive Text Embedding* Benchmark (MTEB), menunjukkan kemampuannya dalam memahami makna lintas bahasa dengan presisi tinggi.

2.3.4 *Vector Database*

Vector database (VDB) merupakan komponen yang dirancang untuk menyimpan dan mengelola representasi vektor hasil proses *embedding*. Basis data ini memungkinkan sistem melakukan pencarian kemiripan secara cepat pada skala besar menggunakan algoritma seperti *Approximate Nearest Neighbor* (ANN) (Gao et al., 2024). Selain menyimpan vektor, VDB juga mencatat *metadata* penting

seperti sumber dokumen dan atribut tambahan yang mendukung proses *filtering* serta *rank*ing hasil pencarian. (Cheng et al., 2018) menjelaskan bahwa penggunaan VDB seperti FAISS, Milvus, Pinecone, atau Chroma memberikan fleksibilitas tinggi dalam mengelola data vektor untuk berbagai kebutuhan aplikasi.

2.3.4.1 ChromaDB

ChromaDB merupakan *vector database* yang dirancang untuk menyimpan dan mengambil *embedding* berdimensi tinggi dalam sistem *Retrieval-Augmented Generation* (RAG). Basis data ini memungkinkan pencarian kemiripan secara efisien, sehingga mendukung proses *retrieval* informasi pada *Large Language Models* (Shaik, Chitralingappa and Harichandana, 2024). Keunggulan utama ChromaDB terletak pada kemampuannya menangani *dataset* berskala besar melalui dukungan indeks penyimpanan di *disk*. Pendekatan ini memberikan performa optimal seiring meningkatnya ukuran data, berbeda dengan in-*memory databases* seperti Faiss atau Qdrant yang cenderung lebih efisien hanya pada *dataset* berukuran kecil (Öztürk and Mesut, 2024). Selain efisiensi penyimpanan, ChromaDB juga memiliki kemampuan pengelolaan koleksi vektor dalam jumlah besar dengan waktu respon yang cepat, menjadikannya solusi praktis bagi sistem RAG yang menuntut kecepatan dan ketepatan dalam proses pencarian konteks (Singh, Talasila and Banakar, 2023).

2.3.5 *Generator* (LLM)

Komponen *generator* berperan sebagai inti dari sistem *Retrieval-Augmented Generation* (RAG), yang bertugas menghasilkan respons akhir berdasarkan konteks yang telah diperoleh dari tahap *retrieval*.

2.4 *Chunking*

2.4.1 Pengertian dan Konsep *Chunking*

Chunking dalam konteks *Retrieval-Augmented Generation* (RAG) merupakan proses penting dalam tahap *preprocessing* yang berfungsi memecah dokumen panjang menjadi segmen-segmen teks berukuran lebih kecil agar sistem dapat memprosesnya secara efisien (Aadit Kshirsagar, 2024). Proses ini memungkinkan sistem *retriever* beroperasi lebih fokus karena informasi dibagi ke dalam unit-unit yang terukur, sehingga setiap segmen dapat dianalisis dan dicocokkan secara lebih efektif dengan *query* pengguna. Dalam arsitektur RAG, *chunking* berperan langsung dalam menentukan keberhasilan proses *dense retrieval* serta kualitas respons yang dihasilkan, karena segmentasi teks yang baik akan menjaga keseimbangan antara kelengkapan konteks dan efisiensi komputasi (Zhao et al., 2025).

Secara konseptual, tujuan utama dari *chunking* adalah mengatasi keterbatasan kapasitas konteks pada *Large Language Models* (LLMs) sekaligus meningkatkan efisiensi pemrosesan dokumen berskala besar (Merola and Singh, 2025). LLM memiliki batas panjang input yang dapat diolah, sehingga memproses dokumen utuh sering kali tidak memungkinkan secara praktis. Dengan membagi teks

menjadi segmen yang lebih kecil, sistem dapat mengurangi konsumsi sumber daya komputasi serta menekan risiko kehilangan konteks akibat panjang sekuens yang berlebihan (Aadit Kshirsagar, 2024). Selain itu, segmentasi yang tepat membantu model untuk lebih fokus pada isi setiap bagian teks, meminimalkan gangguan dari informasi yang tidak relevan, dan meningkatkan presisi dalam memahami makna spesifik yang terkandung dalam setiap *chunk* (Zhao et al., 2025).

Dampak dari *chunking* mencakup tiga aspek utama yang menentukan performa sistem RAG, yaitu *embedding quality*, *retrieval accuracy*, dan *generation quality*. Dalam tahap *embedding*, ukuran segmen yang tepat membantu model merepresentasikan makna semantik dengan lebih akurat karena teks tidak terkompresi secara berlebihan ke dalam ruang vektor (Günther et al., 2025). Selanjutnya, dalam proses *retrieval*, strategi *chunking* yang efektif meningkatkan kemampuan sistem dalam menemukan segmen paling relevan dengan *query*, sehingga hasil pencarian menjadi lebih tepat sasaran (Zhao et al., 2025). Terakhir, pada tahap *generation*, kualitas *chunk* yang baik menyediakan konteks yang koheren bagi LLM untuk menghasilkan respons yang logis dan relevan terhadap pertanyaan pengguna (Singh et al., 2025).

Sebaliknya, *chunking* yang tidak dirancang dengan baik dapat menurunkan efektivitas sistem secara signifikan. Salah satu permasalahan umum adalah fragmentasi konteks, yaitu ketika proses pemecahan teks memutus kontinuitas semantik antarbagian, sehingga makna keseluruhan menjadi terdistorsi (Merola and Singh, 2025). Kondisi ini dapat menyebabkan hilangnya informasi penting yang seharusnya terhubung antara satu *chunk* dengan yang lain. Selain itu, strategi yang kurang tepat dapat menimbulkan *semantic noise*, di mana segmen teks berisi informasi yang tidak lengkap atau terlalu padat, mengakibatkan penurunan akurasi hasil *retrieval* dan kualitas jawaban (Zhao et al., 2025).

2.4.2 Pengaruh Strategi *Chunking* dalam *Pipeline* Sistem RAG

Strategi *chunking* yang diterapkan secara langsung memengaruhi kinerja dua komponen utama dalam sistem RAG, yaitu *retriever* dan *generator*. Pada tahap *retrieval*, pemilihan ukuran dan metode *chunking* yang tepat dapat meningkatkan kemampuan sistem dalam mengidentifikasi konteks yang relevan dengan pertanyaan pengguna. Sebaliknya, segmentasi yang tidak proporsional dapat mengakibatkan pengambilan informasi yang tidak akurat atau bahkan kehilangan konteks penting yang diperlukan dalam proses pencarian (Bhat et al., 2025). Selanjutnya, hasil dari *retrieval* akan menjadi dasar bagi *generator* dalam menyusun respons. Oleh karena itu, kualitas *chunk* yang dihasilkan menentukan sejauh mana sistem dapat menghasilkan jawaban yang koheren, faktual, dan sesuai dengan konteks (Nguyen, Nguyen and Nguyen, 2025).

2.4.3 *Element-Based Chunking*

Element-Based Chunking merupakan pendekatan segmentasi dokumen yang berfokus pada struktur dan elemen penyusun teks seperti *heading*, *subheading*, tabel, daftar, serta blok paragraf. Tidak seperti metode *chunking* konvensional

yang memecah dokumen berdasarkan panjang *token* tetap, pendekatan ini menganalisis organisasi hierarkis dan tata letak dokumen untuk menentukan batas segmen yang logis dan bermakna (Yepes et al., 2024). Prosesnya melibatkan identifikasi elemen struktural dengan bantuan teknik natural *language processing* dan *computer vision*, kemudian menggabungkan elemen-elemen kecil menjadi satuan teks yang utuh tanpa mengorbankan keutuhan struktur aslinya. Dengan cara ini, sistem dapat mempertahankan keselarasan antara isi dokumen dan struktur yang mendasarinya.

Pendekatan berbasis elemen ini menjadi penting, terutama bagi dokumen yang memiliki struktur kompleks dan terformat, seperti laporan statistik atau publikasi resmi Badan Pusat Statistik (BPS). Dokumen semacam ini umumnya disusun dengan organisasi informasi yang sistematis, di mana setiap elemen memiliki fungsi semantik dan tujuan penyajian tertentu. Dengan memanfaatkan struktur tersebut, *Element-Based Chunking* memungkinkan sistem *Retrieval-Augmented Generation* (RAG) memahami keterkaitan antarbagian teks berdasarkan konteks alami dokumen.

2.4.4 *Max-Min Semantic Chunking*

Max-Min Semantic Chunking merupakan pendekatan segmentasi dokumen yang dikembangkan untuk mempertahankan koherensi semantik antarbagian teks melalui penerapan algoritma *Max-Min.* Metode ini berupaya mengatasi keterbatasan strategi *chunking* tradisional yang umumnya didasarkan pada ukuran atau jumlah *token* tertentu. Dengan menitikberatkan pada hubungan makna antar kalimat atau paragraf, pendekatan ini memastikan setiap segmen teks tidak hanya seragam secara panjang, tetapi juga konsisten secara semantik (Kiss, Nagy and Szilágyi, 2025). Pendekatan ini mencerminkan arah baru dalam pengembangan sistem *Retrieval-Augmented Generation* (RAG), di mana kualitas segmentasi menjadi faktor penting dalam menjaga relevansi dan ketepatan hasil *retrieval*.

Secara prinsip, metode ini bekerja dengan menghitung kesamaan semantik antar representasi teks yang telah dikonversi ke dalam bentuk *embedding*. Hasil perhitungan tersebut kemudian dianalisis menggunakan algoritma *Max-Min* untuk menentukan titik pemisahan optimal antara bagian teks dengan koherensi semantik tinggi dan bagian yang mulai menyimpang secara makna (Kiss, Nagy and Szilágyi, 2025). Dengan mekanisme ini, sistem dapat membentuk batas-batas *chunk* yang secara alami mengikuti perubahan topik atau konteks dalam dokumen, bukan sekadar berdasarkan jumlah *token*. Pendekatan ini menjadikan setiap *chunk* lebih representatif terhadap struktur makna keseluruhan, sehingga mendukung pemrosesan teks yang lebih kontekstual pada tahap *retrieval* maupun *generation*.

Max-Min Semantic Chunking mampu menghasilkan segmentasi yang stabil dan relevan tanpa mengorbankan keragaman informasi. Hasil penelitian menunjukkan bahwa metode ini mencapai skor *Adjusted Mutual Information* (AMI) rata-rata 0.85-0.90 serta akurasi 0.56, mengungguli metode pembanding seperti Llama *Semantic Splitter* (Kiss, Nagy and Szilágyi, 2025).

2.4.5 *Recursive Chunking*

Recursive Chunking merupakan metode segmentasi teks yang beroperasi secara bertahap dengan membagi dokumen ke dalam unit-unit yang lebih kecil secara hierarkis hingga mencapai tingkat granularitas yang diinginkan (Danter, Mühle and Stöckl, 2024). Pendekatan ini memanfaatkan struktur linguistik alami dengan menggunakan hierarki pemisah seperti baris baru, spasi, dan tanda baca untuk memecah teks secara rekursif. Melalui proses tersebut, sistem tetap mempertahankan kesadaran terhadap konteks sintaksis dan semantik, sehingga hasil segmentasi tetap bermakna meskipun dilakukan dalam beberapa lapisan (Amiri and Bocklitz, 2025).

Recursive Chunking mampu menciptakan segmen teks yang koheren secara semantik dengan kebutuhan komputasi yang relatif rendah. Hasil penelitian menunjukkan bahwa konfigurasi *recursive token-based chunking*, khususnya R100-0 (100 *token* tanpa *overlap*), secara konsisten memberikan peningkatan *precision* hingga 45% dibandingkan metode *fixed-span chunking* terbaik (Amiri and Bocklitz, 2025).

Dibandingkan metode segmentasi lainnya, *Recursive Chunking* menawarkan keseimbangan optimal antara kualitas hasil *retrieval* dan efisiensi komputasi. Metode berbasis semantik seperti *semantic chunking* memang mampu menghasilkan keselarasan topikal yang tinggi, namun sering kali disertai beban komputasi yang besar dan peningkatan performa yang tidak selalu sebanding (Qu, Tu and Bao, 2024). Sementara itu, *fixed-size chunking* cenderung mengabaikan struktur semantik dan sintaksis, sehingga menghasilkan potongan teks yang terfragmentasi dan kurang kontekstual (Amiri and Bocklitz, 2025).

2.4.6 Analisis Konseptual dan Perbandingan Metode *Chunking*

Secara umum, penelitian ini membandingkan tiga pendekatan utama yang merepresentasikan evolusi teknik *chunking*: *Element-Based Chunking*, *Max-Min Chunking*, dan *Recursive Chunking*. Secara konseptual, ketiga metode tersebut menunjukkan arah perkembangan strategi *chunking* dari berbasis struktur menuju berbasis makna dan adaptasi logis. Setiap metode memiliki ruang penerapan yang berbeda, tergantung pada karakteristik data dan tujuan sistem.

Tabel berikut menyajikan perbandingan prinsip dasar, keunggulan, keterbatasan, serta konteks penerapan yang relevan untuk masing-masing metode *chunking*.

Tabel 2.2 Perbandingan Metode *Chunking*

No	Metode *Chunking*	Prinsip Dasar	Kelebihan	Kelemahan	Relevansi
1	***Element-Based Chunking***	Membagi teks berdasarkan struktur dokumen seperti *heading*,	Menjaga keteraturan isi dan hierarki dokumen	Potensi kehilangan konteks antar elemen	Dokumen terstruktur (laporan, publikasi BPS)

		subjudul, tabel, atau paragraf			
2	***Max-Min Chunking***	Menentukan batas segmen berdasarkan jarak semantik antar *embedding* (*max-min* distance)	Mengurangi redundansi dan mempertahankan keseimbangan makna	Kompleksitas komputasi tinggi; sensitif terhadap kualitas *embedding*	Dokumen dengan topik padat dan bervariasi
3	***Recursive Chunking***	Memecah teks secara bertahap berdasarkan struktur hierarkis seperti paragraf, kalimat, dan spasi untuk menjaga keseimbangan ukuran potongan dan konteks.	Sederhana, efisien, dan menjaga kesinambungan konteks antar segmen.	Tidak mempertimbangkan hubungan semantik atau struktur mendalam dokumen.	Digunakan sebagai *baseline* untuk evaluasi kinerja *chunking* pada dokumen teks panjang seperti publikasi BPS.

Ketiga metode pada Tabel 2.2 merepresentasikan pendekatan yang berbeda dalam membagi dokumen menjadi segmen yang siap digunakan pada sistem *Retrieval-Augmented Generation* (RAG). *Recursive Chunking* berperan sebagai metode *baseline* karena bersifat generik dan efisien tanpa memanfaatkan struktur atau makna semantik secara eksplisit. Sementara itu, *Element-Based Chunking* dan *Max-Min Semantic Chunking* mewakili pendekatan yang lebih adaptif, di mana keduanya mempertimbangkan konteks struktural dan semantik untuk meningkatkan relevansi hasil *retrieval*. Dengan membandingkan ketiga metode ini, penelitian ini bertujuan untuk mengidentifikasi strategi *chunking* yang paling efektif dalam menangani dokumen kompleks dan semi-struktural seperti publikasi BPS.

2.5 Evaluasi Kinerja Sistem

2.5.1 Konsep Evaluasi Kinerja Sistem RAG

Evaluasi kinerja sistem *Retrieval-Augmented Generation* (RAG) merupakan langkah penting untuk menilai sejauh mana sistem mampu menggabungkan proses pencarian informasi dan generasi teks secara efektif. Mengingat arsitektur RAG bersifat hibrida, dengan keterkaitan erat antara komponen *retrieval* dan *generation*, diperlukan pendekatan evaluasi yang mampu menilai performa secara menyeluruh dan terintegrasi (Yu et al., 2024). Penilaian semacam ini tidak hanya bertujuan untuk mengukur akurasi hasil, tetapi juga untuk memahami bagaimana setiap tahap dalam *pipeline*, termasuk *chunking*, *retrieval*, *reranking*, dan *generation*, berkontribusi terhadap kualitas keluaran secara keseluruhan (Es et al., 2024).

Tahap *retrieval* dan *generation* dalam evaluasi RAG memiliki fokus yang berbeda namun saling melengkapi. Evaluasi pada tahap *retrieval* berorientasi pada kemampuan sistem dalam mengidentifikasi dan mengambil informasi yang relevan dari knowledge *base*, sedangkan tahap *generation* menilai sejauh mana teks yang dihasilkan bersifat akurat, koheren, dan tetap setia terhadap konteks sumber (Ru et al., 2024).

2.5.2 Metrik Evaluasi *Retrieval*

Evaluasi tahap *retrieval* pada sistem *Retrieval-Augmented Generation* (RAG) bertujuan untuk menilai sejauh mana sistem mampu menemukan dan menyajikan konteks yang relevan dengan permintaan pengguna. Dalam arsitektur RAG, kualitas *retrieval* berperan krusial karena menjadi dasar bagi model generatif untuk menghasilkan jawaban yang akurat dan kontekstual. Oleh karena itu, diperlukan serangkaian metrik yang tidak hanya mengukur ketepatan hasil pencarian, tetapi juga keseimbangan antara cakupan informasi dan efisiensi sistem dalam menemukan dokumen yang relevan (Sadeli and Lawanda, 2023).

2.5.2.1 Recall@k

Recall@k merupakan metrik yang mengukur sejauh mana sistem berhasil menemukan dokumen relevan di antara top-k hasil pencarian terhadap seluruh dokumen relevan dalam koleksi (Sadeli and Lawanda, 2023). Nilai *recall* yang tinggi menunjukkan kemampuan sistem dalam menjangkau cakupan informasi yang luas, memastikan tidak ada konteks penting yang terlewat dalam proses pencarian. Dengan demikian, meskipun *recall* menggambarkan kemampuan sistem dalam menjangkau informasi secara menyeluruh, nilai tersebut perlu diimbangi dengan metrik lain untuk memastikan efisiensi dan ketepatan hasil *retrieval*. Perhitungan *recall* diberikan pada Persamaan (2.1).

$$Recall = \frac{TP}{TP + FN} \tag{2.1}$$

2.5.2.2 Precision@k

Berbeda dengan *recall*, metrik *precision*@k berfokus pada ketepatan hasil pencarian. Metrik ini mengukur rasio antara jumlah hasil relevan dengan jumlah keseluruhan hasil yang dikembalikan pada top-k, yang mencerminkan kemampuan sistem dalam menyeleksi hanya dokumen atau *chunk* yang benar-benar relevan terhadap *query* (Sadeli and Lawanda, 2023). Formula *precision* ditunjukkan pada Persamaan (2.2).

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

2.5.2.3 Mean Reciprocal Rank (MRR)

Mean Reciprocal Rank (MRR) digunakan untuk mengukur efisiensi sistem dalam menempatkan hasil relevan pada peringkat teratas. Metrik ini menghitung rata-rata kebalikan dari posisi peringkat hasil relevan pertama dalam daftar pencarian (Ferrante, Ferro and Fuhr, 2021). Nilai MRR yang tinggi menunjukkan bahwa sistem dapat menemukan dokumen relevan pada urutan awal, sehingga

mempercepat proses akses informasi dan meningkatkan efektivitas tahap *generation*. Perhitungan MRR ditampilkan pada Persamaan (2.3).

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i} \tag{2.3}$$

2.5.3 Metrik Evaluasi *Generation*

Tahap *generation* dalam sistem *Retrieval-Augmented Generation* (RAG) berperan penting dalam menghasilkan teks akhir yang koheren, relevan, dan kontekstual berdasarkan hasil *retrieval*. Kualitas pada tahap ini sangat menentukan efektivitas keseluruhan sistem, karena hasil generatif mencerminkan seberapa baik model bahasa mampu memanfaatkan informasi eksternal yang telah diambil sebelumnya. Oleh karena itu, diperlukan metrik evaluasi yang dapat menilai tidak hanya kesamaan linguistik antara teks hasil generasi dan referensi, tetapi juga kesetiaan makna serta urutan logis antar kata dalam konteks semantik yang utuh (Evtikhiev et al., 2023).

2.5.3.1 ROUGE-L

ROUGE-L (*Recall-Oriented Understudy for Gisting Evaluation - Longest Common Subsequence)* merupakan metrik evaluasi yang mengukur kesamaan urutan kata antara hasil generasi dan teks referensi berdasarkan *subsequence* terpanjang yang sama (Evtikhiev et al., 2023). Nilai *ROUGE-L* yang tinggi menunjukkan bahwa hasil generasi memiliki kesamaan semantik yang kuat dan mengikuti struktur logis dokumen acuan (Mastropaolo et al., 2024). Perhitungan skor ROUGE-L ditunjukkan pada Persamaan (2.4).

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs}+\beta^2 P_{lcs}} \tag{2.4}$$

2.5.3.2 BLEU

BLEU (*Bilingual Evaluation Understudy)* merupakan metrik yang menilai kesamaan antara hasil generasi dan teks referensi berdasarkan presisi n-gram, yaitu perbandingan antara kata atau frasa yang dihasilkan dengan yang terdapat pada teks acuan (Evtikhiev et al., 2023). Metrik ini mengukur sejauh mana sistem memilih kata, frasa, dan struktur kalimat yang sesuai secara leksikal maupun sintaksis Nilai *BLEU* yang tinggi menunjukkan tingkat ketepatan linguistik yang baik, di mana sistem tidak hanya memahami konteks tetapi juga mampu menghasilkan kalimat yang secara gramatikal benar dan sesuai dengan referensi. Formula utama BLEU disajikan pada Persamaan (2.5), sedangkan perhitungan *brevity penalty* diperlihatkan pada Persamaan (2.6).

$$BLEU = BP \cdot exp(\sum_{n=1}^{N} w_n \log p_n) \tag{2.5}$$

$$BP = \begin{cases} 1 \\ e^{(1-\frac{r}{c})} \end{cases} \tag{2.6}$$

2.6 Preprocess Dokumen PDF

Text preprocessing merupakan tahapan krusial dalam menyiapkan data teks mentah sebelum dianalisis lebih lanjut, terutama dalam sistem *Retrieval-Augmented Generation* (RAG). Tahap ini berfungsi untuk memastikan bahwa teks yang diperoleh dari berbagai sumber memiliki kualitas dan format yang seragam sehingga dapat diproses secara efektif oleh model bahasa. Berdasarkan penelitian sebelumnya, *preprocessing* mencakup serangkaian langkah sistematis untuk mengubah teks yang tidak terstruktur menjadi data yang bersih, terstandarisasi, dan siap diolah untuk analisis lanjutan (Pradana and Fitrianah, 2024).

2.6.1 PyMuPDF

PyMuPDF, yang juga dikenal sebagai fitz, adalah *library* Python berkinerja tinggi yang digunakan untuk mengakses, memodifikasi, dan mengekstrak data dari dokumen PDF. Dalam *pipeline Retrieval-Augmented Generation* (RAG), PyMuPDF memegang peranan krusial pada tahap praproses awal, khususnya untuk metode *chunking* yang "buta-*layout*" (*layout-blind)* seperti *Recursive* dan *Max-Min Semantic*. Fungsi utamanya adalah melakukan ekstraksi teks mentah (*raw text extraction)* secara efisien dari setiap halaman dokumen publikasi BPS. Hasil dari ekstraksi ini adalah sebuah *string* teks panjang yang kemudian dapat dibersihkan dan diolah lebih lanjut, namun umumnya masih mengandung elemen *noise* seperti *header*, *footer,* dan nomor halaman yang ikut terekstrak.

2.6.2 *Regex Clean*

Regular Expressions (*Regex*) adalah sebuah bahasa kueri khusus yang digunakan untuk pencocokan pola (pattern matching) di dalam data teks. Dalam konteks praproses data untuk RAG, *Regex* adalah alat utama untuk pembersihan teks (*text cleaning*) yang dilakukan setelah tahap ekstraksi PyMuPDF. Teknik ini sangat efisien untuk mengidentifikasi dan menghapus elemen-elemen *noise* yang berulang dan tidak relevan, seperti *header* dan *footer* yang muncul di setiap halaman, nomor halaman, atau artefak pemisah kolom yang sering ditemukan dalam PDF BPS. Tujuan dari langkah ini adalah untuk menghasilkan dokumen teks bersih (*clean text)* yang hanya berisi konten naratif dan tabel yang substantif, sehingga siap untuk diproses oleh algoritma *chunking*.

2.6.3 *Partition* PDF

Partition PDF (*partition_pdf*) adalah fungsi inti dari *library unstructured*.io yang dirancang spesifik untuk *pipeline* RAG yang bersifat *layout-aware* . Berbeda dengan PyMuPDF yang hanya mengekstrak seluruh teks mentah, *partition_pdf* menganalisis struktur visual dan *metadata* dokumen (terutama saat menggunakan strategi *hi_res*) untuk mengidentifikasi dan mempartisi konten secara cerdas ke dalam elemen-elemen logis seperti *Title* (Judul), *NarrativeText* (Paragraf), dan *Table* (Tabel). Dalam penelitian ini, fungsi ini merupakan implementasi utama dari *Element-Based Chunking*, di mana proses ekstraksi,

pembersihan *noise* (seperti deteksi *header/footer)*, dan segmentasi awal (pemecahan per elemen) terjadi dalam satu langkah terintegrasi.

BAB 3 METODOLOGI

Bab ini menjelaskan rancangan dan tahapan metodologis yang digunakan untuk menganalisis pengaruh metode *chunking* terhadap kinerja sistem *Retrieval-Augmented Generation* (RAG) pada dokumen publikasi Badan Pusat Statistik (BPS). Metodologi penelitian disusun secara sistematis untuk menggambarkan alur penelitian mulai dari pengumpulan dan pemrosesan data, penerapan berbagai strategi *chunking*, pembangunan sistem RAG berbasis model Qwen3-8B, hingga evaluasi performa menggunakan metrik *retrieval* dan *generation*. Pendekatan ini dirancang agar hasil penelitian dapat menjawab rumusan masalah secara terukur dan dapat direplikasi, sekaligus memberikan dasar yang kuat dalam menilai efektivitas setiap metode *chunking* dalam meningkatkan relevansi serta kualitas hasil generasi teks pada konteks dokumen statistik nasional.

3.1 Desain Penelitian

3.1.1 Tipe Penelitian

Penelitian ini termasuk dalam kategori nonimplementatif-analitik dengan pendekatan kuantitatif. Fokus penelitian terletak pada analisis hubungan antara variasi metode *chunking* dengan perubahan kinerja sistem *Retrieval-Augmented Generation* (RAG) secara terukur melalui data numerik hasil evaluasi. Pendekatan ini dipilih karena penelitian tidak berorientasi pada pengembangan sistem baru, melainkan pada pengujian dan analisis pengaruh variabel tertentu terhadap performa sistem yang telah ada. Dengan demikian, hasil penelitian diharapkan mampu memberikan pemahaman empiris dan objektif mengenai sejauh mana perbedaan strategi *chunking* memengaruhi efektivitas RAG dalam memproses dokumen publikasi Badan Pusat Statistik (BPS).

3.1.2 Strategi Penelitian

Strategi penelitian yang digunakan adalah eksperimen untuk menilai pengaruh variasi metode *chunking* terhadap kinerja sistem RAG pada dokumen publikasi BPS. Variabel bebas ialah metode *chunking* (*Element-Based*, *Max-Min Semantic*, dan *Recursive Chunking*), sedangkan variabel terikat ialah performa sistem yang diukur dengan metrik *retrieval* (*Precision*@k, *Recall*@k, MRR) dan *generation* (*BLEU* dan *ROUGE-L*). Faktor lain dijaga konstan seperti model *generator*, skema *embedding*, *retriever*, korpus BPS, serta prosedur praproses, agar perbandingan antarperlakuan valid dan dapat direplikasi.

3.1.3 Lokasi dan Lingkungan Penelitian

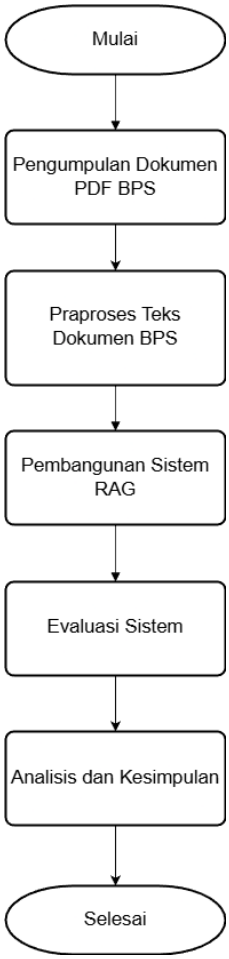
Penelitian dilaksanakan di lingkungan komputasi lokal dan server GPU AI Center Universitas B*rawi*jaya untuk pemrosesan intensif. Seluruh data diperoleh dari situs resmi BPS dan diolah *offline*. Lingkungan perangkat lunak yang digunakan meliputi Python, LangChain, FAISS, Qwen3-8B, Qwen3-8B *Embedding*, PyMuPDF (ekstraksi

PDF & tabel); sedangkan spesifikasi minimum perangkat keras Adalah RAM ≥16 GB, dan GPU Nvidia RTX dengan VRAM ≥6GB.

3.2 Prosedur Penelitian

Prosedur penelitian ini menjelaskan tahapan-tahapan sistematis yang dilakukan untuk menganalisis pengaruh metode *chunking* terhadap kinerja sistem *Retrieval-Augmented Generation* (RAG) pada dokumen publikasi Badan Pusat Statistik (BPS). Setiap tahap dirancang secara berurutan mulai dari pengumpulan dan praproses data hingga implementasi sistem RAG dan evaluasi hasilnya.

Alur lengkap tahapan penelitian yang dilakukan dalam studi ini ditunjukkan pada Gambar 3.1. Setiap langkah dalam diagram tersebut merepresentasikan proses utama yang dilakukan secara berurutan, mulai dari pengumpulan dan praproses dokumen BPS, pembangunan sistem *Retrieval-Augmented Generation* (RAG), hingga tahap evaluasi dan analisis hasil. *Flowchart* ini dirancang untuk memberikan gambaran menyeluruh mengenai urutan kegiatan penelitian, sehingga hubungan antara setiap tahap dapat dipahami secara sistematis dan konsisten dengan tujuan penelitian.



Gambar 3.1 *Flowchart* Prosedur Penelitian

3.2.1 Data Penelitian

Data yang digunakan dalam penelitian ini merupakan data sekunder yang diperoleh secara manual dari situs resmi Badan Pusat Statistik (BPS) melalui laman https://www.bps.go.id/id/publication. Publikasi BPS dipilih karena memiliki kredibilitas tinggi, format penyajian yang terstandar, dan cakupan informasi yang luas mengenai kondisi sosial, ekonomi, serta demografi Indonesia.

Jenis data yang digunakan adalah dokumen publikasi statistik tahunan dalam format PDF, yang terdiri dari teks naratif, tabel numerik, dan deskripsi grafik. Dokumen ini dipilih karena merepresentasikan struktur teks yang kompleks, sesuai dengan konteks penelitian *Retrieval-Augmented Generation* (RAG) yang membutuhkan pemrosesan teks formal dan semi-terstruktur. Sebanyak 10 dokumen publikasi digunakan dalam penelitian ini. Ringkasan dokumen yang digunakan dapat dilihat pada Tabel 3.1.

Tabel 3.1 Ringkasan Data Penelitian

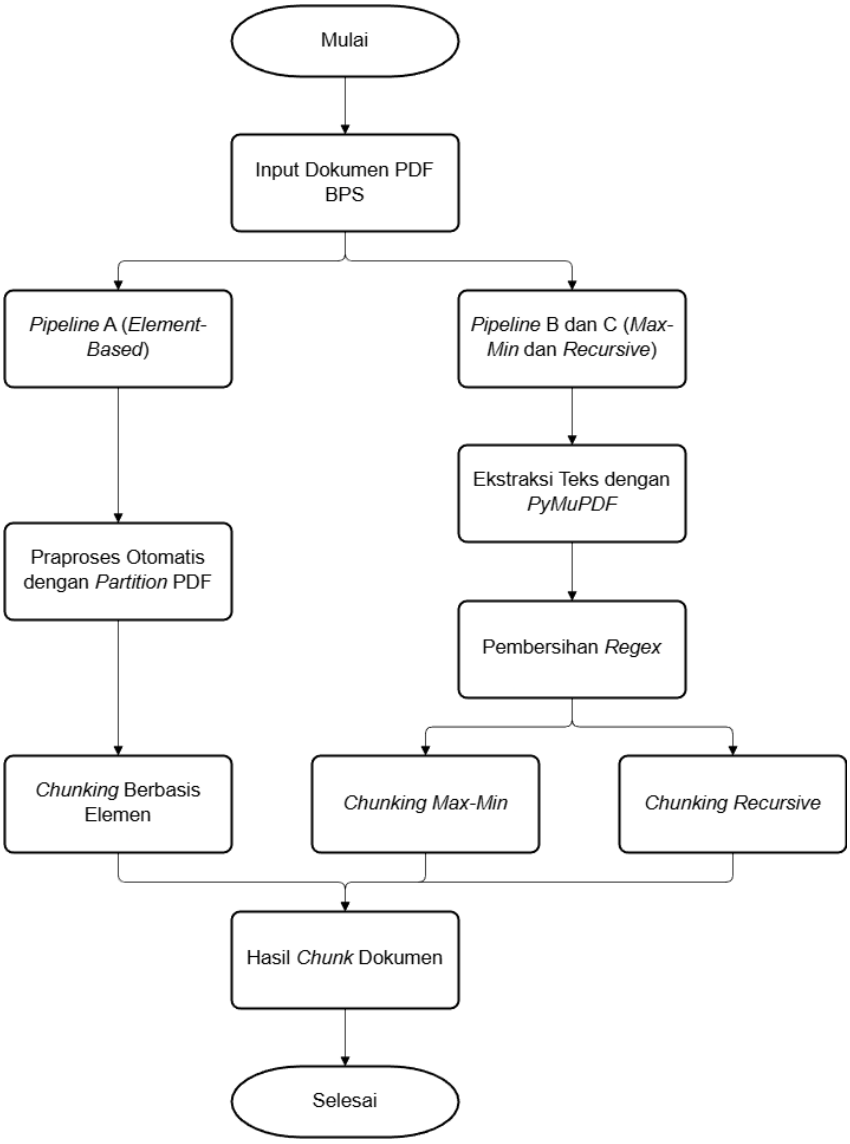
No	Judul Publikasi	Tahun Cakupan Data	Tahun Terbit	Keterangan
1	*Benchmark* Indeks Konstruksi	2018-2023	2025	Menyajikan indeks konstruksi tiap provinsi; digunakan untuk menganalisis sektor konstruksi nasional
2	*Benchmark* Statistik Konstruksi	2018-2023	2025	Menyajikan statistik pekerja, upah, dan nilai konstruksi; pasangan dari dokumen No. 1
3	Cerita Data Statistik untuk Indonesia: *Mismatch* Pendidikan-Pekerjaan Pemuda Indonesia	2025	2025	Artikel tematik (CERDAS) berisi analisis ketidaksesuaian pendidikan dan pekerjaan
4	Direktori Industri Manufaktur 2025	2025	2025	Basis data perusahaan manufaktur, berisi persebaran dan profil industri

5	Indeks Unit Value Ekspor-Impor	2024-2025	2025	Indeks harga ekspor-impor, mencerminkan perubahan nilai barang perdagangan luar negeri
6	Neraca Lembaga Nonprofit yang Melayani Rumah Tangga (LNPRT)	2022-2024	2025	Analisis kontribusi lembaga nonprofit dalam ekonomi nasional
7	Neraca Pemerintahan Umum Indonesia	2019-2024	2025	Neraca produksi, pendapatan, konsumsi, tabungan, dan investasi pemerintah
8	Neraca Rumah Tangga Indonesia	2022-2024	2025	Aktivitas ekonomi rumah tangga: produksi, konsumsi, pendapatan, tabungan
9	Statistik Perdagangan Luar Negeri Bulanan - Impor	2024-2025	2025	Nilai dan volume impor berdasarkan komoditas HS, negara asal, provinsi
10	Statistik Perdagangan Luar Negeri Menurut Moda Transportasi	2023-2024	2025	Data ekspor-impor berdasarkan moda (laut, udara, darat, pos, pipa)

Sepuluh publikasi yang tercantum pada Tabel 3.1 mewakili berbagai topik statistik nasional dengan struktur penyajian yang seragam, yaitu kombinasi antara teks naratif, tabel, dan deskripsi grafik. Variasi konten ini memastikan bahwa data yang digunakan dapat merepresentasikan karakteristik umum dokumen publikasi BPS, sekaligus memberikan variasi semantik yang memadai untuk pengujian metode *chunking*. Jumlah dan keragaman dokumen tersebut juga dianggap cukup untuk menghasilkan volume teks yang memadai bagi proses *retrieval* dan *generation* dalam sistem *Retrieval-Augmented Generation* (RAG).

3.2.2 Praproses dan *Chunking*

Tahap praproses dan *chunking* merupakan komponen utama dalam penelitian ini karena secara langsung menentukan kualitas representasi data yang digunakan oleh sistem *Retrieval-Augmented Generation* (RAG). Pada tahap ini, dilakukan serangkaian proses untuk mengekstraksi, membersihkan, dan membagi dokumen publikasi Badan Pusat Statistik (BPS) menjadi potongan-potongan informasi yang terstruktur atau disebut *chunk*. Setiap metode *chunking* dirancang dengan pendekatan yang berbeda guna menguji pengaruh strategi pemecahan dokumen terhadap efektivitas proses *retrieval* dan kualitas hasil generatif model. Secara umum, alur kerja dari ketiga *pipeline* yang diusulkan dapat dilihat pada Gambar 3.2, yang menggambarkan tahapan praproses dan *chunking* dari input dokumen mentah hingga menghasilkan kumpulan *chunk* siap diolah pada tahap *embedding*.



Gambar 3.2 *Flowchart* Tahapan Praproses dan *Chunking*

Berdasarkan diagram alir tersebut, penelitian ini mengimplementasikan tiga variasi *pipeline* dengan karakteristik dan pendekatan *chunking* yang berbeda.

Setiap *pipeline* merepresentasikan strategi pemrosesan dokumen yang unik, dimulai dari pendekatan berbasis elemen visual dokumen (*layout-aware)*, pendekatan berbasis kesamaan semantik antar kalimat (*layout-blind)*, hingga pendekatan konvensional berbasis panjang teks sebagai *baseline* pembanding.

3.2.2.1 Pipeline A: Element-Based Chunking

Pipeline ini diimplementasikan menggunakan *library unstructured*.io, yang mampu melakukan analisis tata letak dokumen dan memisahkan elemen berdasarkan posisi spasial serta tipe kontennya. Fungsi utama yang digunakan adalah *partition_pdf* dengan parameter strategy="*hi_res*", yang memungkinkan deteksi elemen beresolusi tinggi seperti paragraf, tabel, dan daftar dalam dokumen PDF.

Proses pelaksanaan *pipeline* dilakukan melalui beberapa tahapan berikut:

1. Input: File PDF asli dari publikasi BPS digunakan tanpa konversi format agar struktur *layout* tetap terjaga.
2. Analisis *Layout: unstructured*.*partition_pdf* membaca halaman demi halaman untuk mendeteksi elemen visual seperti judul, paragraf, tabel, dan daftar.
3. Ekstraksi dan Pembersihan: Teks dari setiap elemen diekstraksi secara otomatis; proses ini juga menghapus *header*, *footer,* dan elemen non-teks yang tidak relevan.
4. Pembentukan *Chunk*: Setiap elemen hasil deteksi dijadikan satu unit *chunk* lengkap dengan *metadata* (jenis elemen, posisi halaman, urutan kemunculan).
5. *Output*: Hasil akhir berupa kumpulan *chunk* terstruktur dalam format JSON yang siap diproses pada tahap *embedding* di arsitektur RAG.

3.2.2.2 Pipeline B: Max-Min Chunking

Pipeline ini bertujuan untuk menguji efektivitas metode *chunking* berbasis kesamaan semantik yang tidak memperhatikan tata letak visual dokumen. Pendekatan ini berfokus pada hubungan makna antar kalimat dengan asumsi bahwa pemisahan teks berdasarkan kemiripan semantik akan menghasilkan potongan informasi yang lebih koheren dan relevan dibandingkan segmentasi yang hanya bergantung pada struktur fisik. Proses *chunking* dilakukan menggunakan *library* MaxMin*Chunk*er, sebuah alat yang mengimplementasikan algoritma berbasis prinsip *maximum-minimum similarity*.

Pipeline ini membutuhkan tahapan praproses yang berbeda dari *pipeline* sebelumnya karena seluruh operasi dilakukan pada teks mentah hasil ekstraksi.

1. Input Data
   - Menggunakan teks bersih yang sama dengan *Pipeline* B.
   - Sumber teks berasal dari hasil ekstraksi PyMuPDF dan pembersihan menggunakan *Regex*.
2. Tahapan Praproses

- Ekstraksi Teks: Menggunakan PyMuPDF (fitz) untuk mengekstrak teks mentah dari file PDF secara terstruktur per halaman.
- Pembersihan Teks: Menggunakan *Regex Cleaning* untuk menghapus elemen yang tidak relevan seperti *header*, *footer,* nomor halaman, dan karakter khusus.
- Input Bersih: Hasil akhir tahap ini menjadi input utama untuk proses *chunking*.

3. Proses *Chunking* (*Max-Min* Algorithm)
   - Menggunakan *library* MaxMin*Chunk*er berbasis algoritma *maximum-minimum similarity*.
   - Setiap kalimat direpresentasikan dalam bentuk *embedding* (vektor semantik).
   - Sistem menghitung:
     1) Kemiripan maksimum (*max_sim*) antara kalimat baru dan *chunk* aktif.
     2) Kemiripan minimum (*min_sim*) antar kalimat dalam *chunk* aktif.
   - Logika Pemisahan:
     1) Jika *max_sim* ≥ *min_sim*, kalimat ditambahkan ke *chunk* yang sama.
     2) Jika tidak, sistem memulai *chunk* baru.
4. *Output* Akhir
   - Menghasilkan *chunk* yang dibentuk secara dinamis berdasarkan kesamaan makna antar kalimat.
   - Setiap *chunk* disimpan dalam format JSON yang berisi urutan kalimat dan indeks posisi teks.

3.2.2.3 Pipeline C: Recursive Chunking

Metode ini bersifat *structure-aware segmentation* yang mempertahankan kontinuitas konteks serta menggunakan input teks bersih yang sama seperti pada *Pipeline* B (hasil ekstraksi PyMuPDF dan pembersihan *Regex*). Teks tersebut kemudian dipecah secara rekursif berdasarkan batas panjang karakter tertentu dengan sedikit tumpang tindih antar *chunk* agar konteks tidak terputus. Implementasi dilakukan menggunakan *RecursiveCharacterTextSplit*ter dari LangChain, yang secara otomatis membagi teks berdasarkan pemisah seperti paragraf, kalimat, dan spasi hingga mencapai ukuran *chunk* yang diinginkan.

Pipeline ini dijalankan melalui serangkaian tahapan berikut:

1. Input Data
   - Menggunakan teks bersih yang sama dengan *Pipeline* B.
   - Sumber teks berasal dari hasil ekstraksi PyMuPDF dan pembersihan menggunakan *Regex*.

2. Penentuan Parameter *Chunking*
    - Menentukan panjang maksimal karakter per *chunk* (1.000 karakter).
    - Menetapkan *overlap* kecil antar *chunk* (100-200 karakter) agar konteks tetap berlanjut.
3. Proses Pemisahan Teks (*Recursive Split*ting)
    - Menggunakan *class RecursiveCharacterTextSplit*ter dari LangChain.
    - Sistem membagi teks berdasarkan urutan prioritas pemisah: paragraf → kalimat → spasi → karakter.
    - Jika potongan melebihi batas karakter, teks akan dipecah kembali secara rekursif hingga memenuhi ukuran yang ditetapkan.
4. Rekonstruksi & Validasi *Chunk*
    - Potongan hasil pemisahan digabung ulang untuk memastikan tidak ada bagian teks yang hilang.
    - *Overlap* diterapkan agar makna antar*chunk* tetap berkesinambungan.
5. *Output* Akhir
    - Menghasilkan sekumpulan *chunk* berukuran tetap dalam format teks/JSON.

3.2.3 Arsitektur SIstem RAG

1. Model *Embedding*: Qwen3-8B *Embedding* Model
2. *Vector Database*: ChromaDB
3. Model *Generator*: Qwen3-8B *Base* Model

3.2.4 Rancangan Evaluasi

3.2.4.1 Metrik Evaluasi Retrieval

1. *Precision*@k
    - Mengukur proporsi *chunk* relevan dari total *chunk* yang diambil sebanyak k teratas.
    - Semakin tinggi nilai *Precision*@k, semakin akurat sistem dalam memilih konteks yang tepat.
2. *Recall*@k
    - Mengukur proporsi *chunk* relevan yang berhasil ditemukan dari seluruh *chunk* relevan yang tersedia.
    - Nilai *Recall*@k yang tinggi menunjukkan kemampuan sistem dalam menjangkau konteks yang lebih lengkap.
3. *Mean Reciprocal Rank* (MRR)
    - Menilai posisi peringkat pertama *chunk* yang relevan pada hasil *retrieval*.

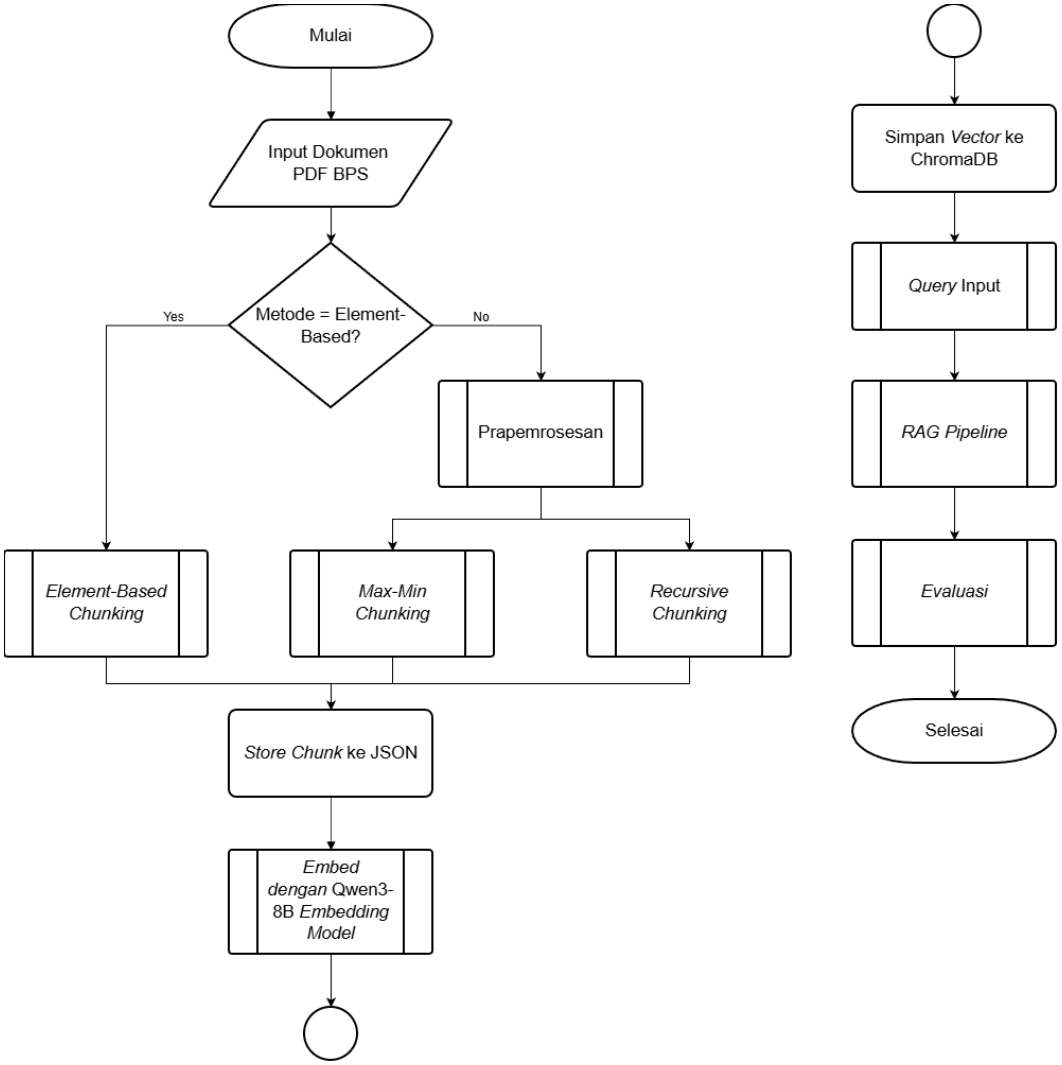
- MRR tinggi berarti sistem mampu menempatkan *chunk* relevan di posisi teratas hasil pencarian, yang penting untuk efisiensi *retrieval*.

3.2.4.2 Metrik Evaluasi Generation

1. *BLEU (Bilingual Evaluation Understudy)*
   - Mengukur tingkat kesamaan n-gram antara jawaban yang dihasilkan model dan jawaban referensi.
   - Nilai *BLEU* yang tinggi menunjukkan bahwa model menghasilkan teks yang lebih akurat secara leksikal dan mendekati jawaban rujukan.
2. *ROUGE-L (Recall-Oriented Understudy for Gisting Evaluation - Longest Common Subsequence)*
   - Mengukur kesamaan struktur kalimat berdasarkan urutan kata yang paling panjang dan konsisten antara hasil generasi dan referensi.
   - Nilai *ROUGE-L* tinggi menunjukkan bahwa model menghasilkan kalimat yang koheren dan kontekstual terhadap pertanyaan yang diajukan.

BAB 4 PERANCANGAN

4.1 Perancangan Algoritma



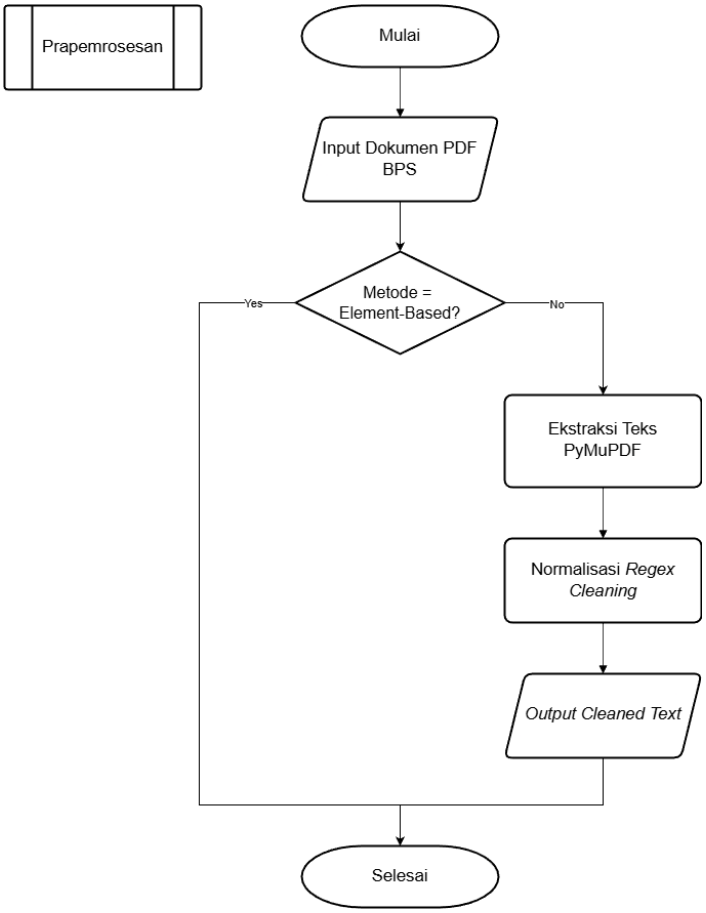
Gambar 4.1 *Flowchart* Alur Umum

Gambar 4.1 memperlihatkan alur umum sistem *Retrieval-Augmented Generation* (RAG) yang dikembangkan pada penelitian ini. Proses dimulai dengan tahap input dokumen publikasi Badan Pusat Statistik (BPS) yang akan menjadi sumber data utama. Dokumen PDF tersebut diproses melalui dua pendekatan praproses berbeda sesuai metode *chunking* yang digunakan. Pada *Element-Based Chunking*, praproses dilakukan menggunakan fungsi *partition_pdf* dari *library unstructured*.io untuk mengekstraksi elemen-elemen terstruktur seperti paragraf, tabel, dan subjudul. Sementara pada *Max-Min Semantic Chunking* dan *Recursive Chunking*, tahap praproses dilakukan dengan kombinasi PyMuPDF dan *regex clean*ing untuk mengekstraksi teks mentah dan menghapus karakter atau simbol yang tidak relevan.

Tahap berikutnya adalah *chunking* dokumen, di mana teks dibagi menjadi beberapa segmen sesuai tiga metode yang diuji: *Element-Based*, *Max-Min Semantic*, dan *Recursive Chunking*. Setiap hasil segmentasi kemudian disimpan dalam format JSON untuk memudahkan proses *embedding*. Tahap selanjutnya, sistem melakukan *embedding* menggunakan model Qwen3-8B *Embedding*, yang mengubah setiap *chunk* menjadi representasi vektor berdimensi tinggi. Hasil *embedding* kemudian disimpan ke dalam ChromaDB sebagai *vector database* untuk mendukung pencarian berbasis kesamaan semantik.

Selanjutnya, proses *Retrieval-Augmented Generation* (RAG) dijalankan dengan menerima *query* pengguna, melakukan pencarian kesamaan antar vektor, mengambil top-k *chunk* paling relevan, dan menghasilkan jawaban akhir menggunakan model Qwen3-8B sebagai generator. Tahap terakhir adalah evaluasi kinerja sistem, yang dilakukan dengan menghitung metrik *Precision*@k, *Recall*@k, Mean Reciprocal Rank (MRR) untuk aspek *retrieval*, serta BLEU dan ROUGE-L untuk aspek *generation*. Seluruh alur ini diakhiri dengan proses validasi hasil untuk menilai pengaruh masing-masing metode *chunking* terhadap performa sistem RAG.

4.1.1 Prapemrosesan



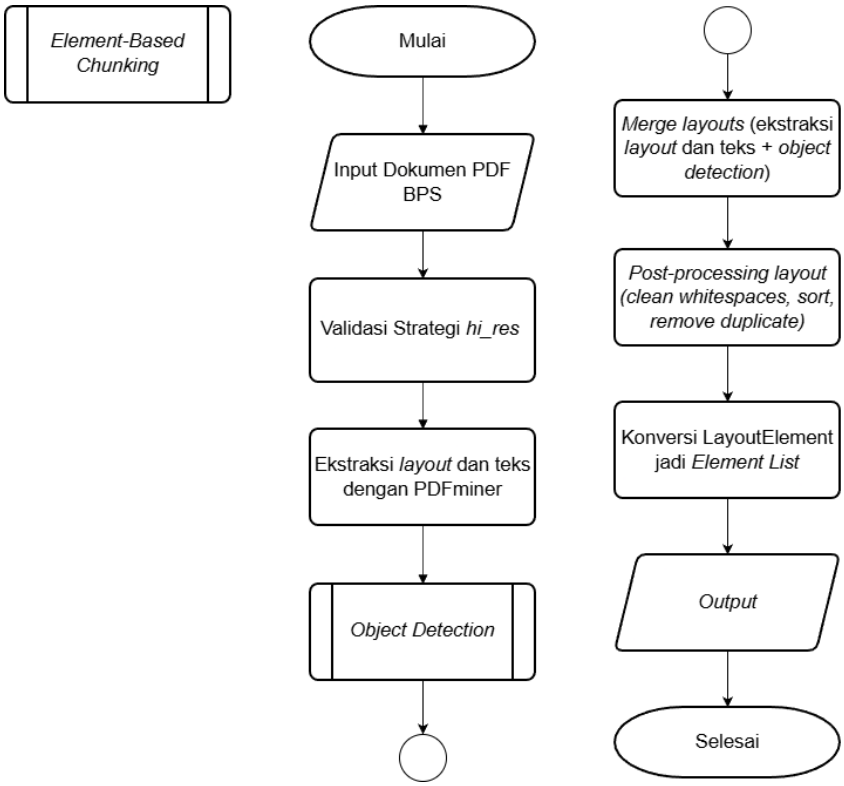
Gambar 4.2 *Flowchart* Praproses

Gambar 4.2 menunjukkan alur proses *preprocessing* dokumen yang digunakan pada penelitian ini. Tahapan *preprocessing* hanya diterapkan pada *pipeline* yang menggunakan metode *Max-Min Semantic Chunking* dan *Recursive Chunking*, sementara metode *Element-Based Chunking* tidak melalui tahap ini karena proses ekstraksi dan segmentasi elemen dokumen telah ditangani secara langsung oleh fungsi *partition_pdf*. Oleh karena itu, pada awal alur dilakukan proses pengecekan metode *chunking* yang digunakan. Jika metode yang dipilih adalah *Element-Based*, sistem akan melewati seluruh tahapan *preprocessing* dan langsung menuju tahap *chunking* pada subbagian berikutnya.

Apabila metode yang digunakan adalah *Max-Min* atau *Recursive*, sistem melanjutkan ke proses ekstraksi teks menggunakan PyMuPDF, yang bertujuan memperoleh teks mentah dari dokumen PDF BPS. Teks mentah ini kemudian melalui tahap normalisasi menggunakan *Regex Clean*ing, yang mencakup penghapusan karakter tidak relevan, perapian struktur *whitespace*, penyesuaian tanda baca, serta penyelarasan format teks agar konten lebih bersih dan konsisten. Hasil akhir dari tahap *preprocessing* adalah *cleaned text* yang siap diproses pada tahap *chunking* sesuai metode masing-masing. Dengan demikian, alur ini memastikan bahwa setiap metode *chunking* memperoleh masukan teks yang sesuai kebutuhan *pipeline*-nya.

4.1.2 *Chunking*

4.1.2.1 Rancangan Element-Based Chunking



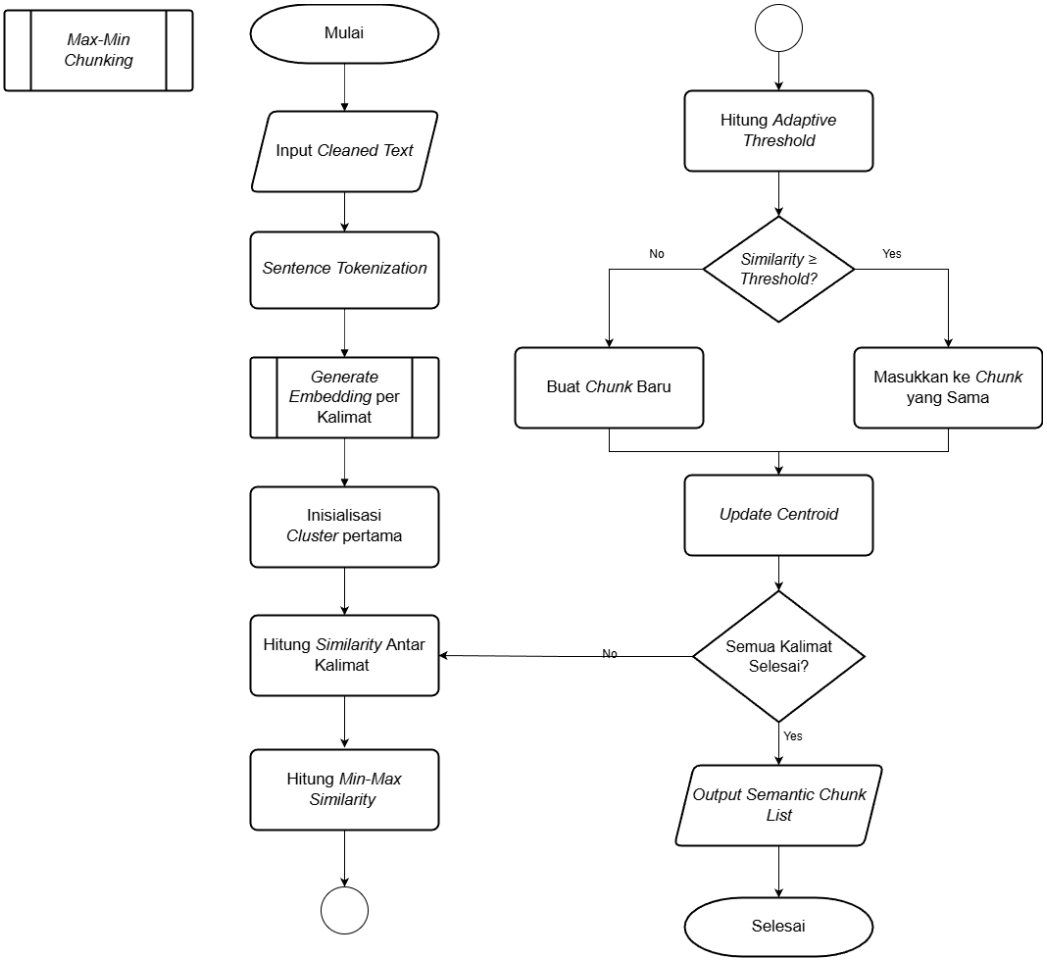
Gambar 4.3 *Flowchart Element-Based Chunking*

Gambar 4.3 menunjukkan alur proses *Element-Based Chunking* yang diimplementasikan menggunakan fungsi *partition_pdf* dengan strategi *hi_res*. Proses dimulai dengan menerima masukan berupa dokumen PDF publikasi BPS, kemudian sistem melakukan validasi terhadap parameter *hi_res* untuk memastikan bahwa strategi ekstraksi yang digunakan sesuai dengan karakteristik dokumen. Setelah validasi dilakukan, PDF diproses menggunakan PDFMiner untuk mengekstraksi teks ter-*embed* dan struktur *layout* dasar seperti posisi elemen, *bounding box*, serta objek grafis yang terdapat pada halaman dokumen.

Tahap berikutnya adalah proses *object detection*, yang menjalankan model deteksi tata letak untuk mengidentifikasi elemen-elemen penting seperti judul, paragraf, daftar, tabel, maupun gambar. Hasil inferensi model ini kemudian digabungkan dengan hasil ekstraksi PDFMiner melalui proses *merge layouts*. Pada tahap ini dilakukan serangkaian aturan penggabungan untuk menghilangkan duplikasi elemen, mempertahankan struktur dokumen, serta menyelaraskan hasil deteksi model dengan informasi *layout* yang diekstrak sebelumnya.

Setelah proses penggabungan selesai, sistem melakukan *post-processing layout* yang mencakup pembersihan *whitespace*, penyaringan elemen kosong, pengurutan elemen berdasarkan alur visual dokumen, serta penentuan *list* item secara otomatis. Tahap terakhir adalah konversi setiap *LayoutElement* menjadi *Element List* yang terstruktur, yaitu sekumpulan elemen dokumen yang telah siap digunakan sebagai *chunk* dalam *pipeline* RAG. Dengan demikian, metode *Element-Based Chunking* menghasilkan potongan teks yang mempertahankan struktur asli dokumen tanpa memerlukan *preprocessing* tambahan.

4.1.2.2 Max-Min Chunking



Gambar 4.4 *Flowchart Max-Min Chunking*

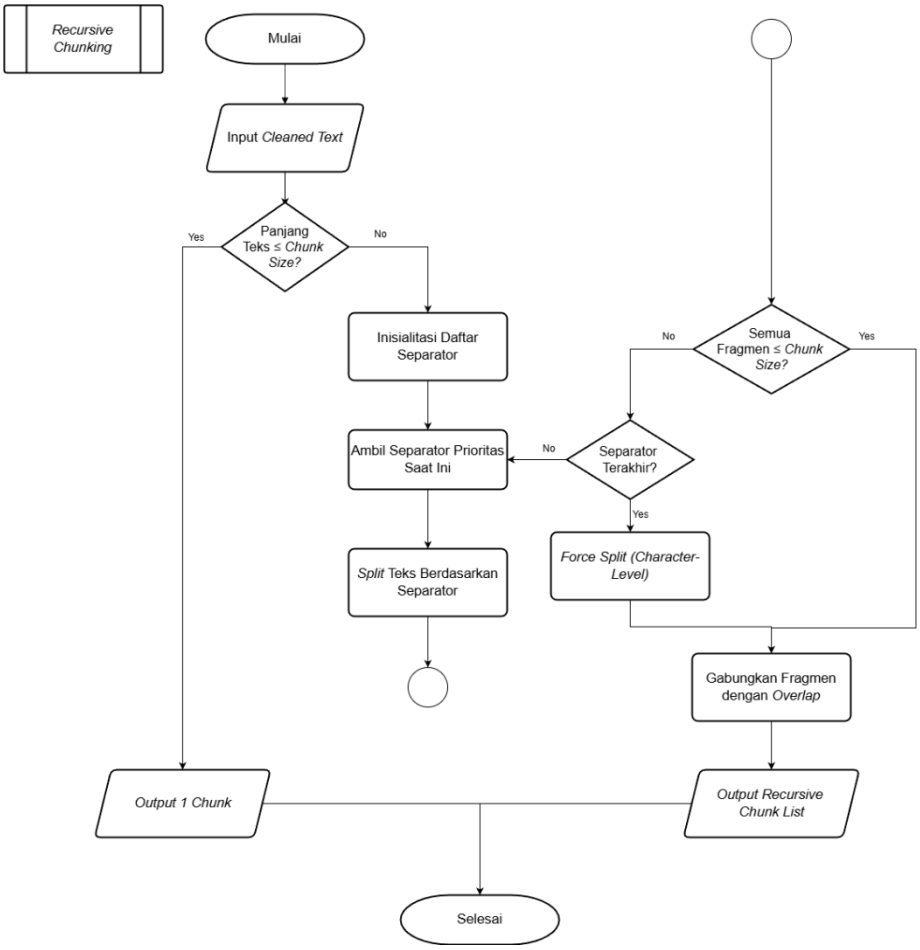
Flowchart pada Gambar 4.4 menggambarkan alur kerja metode *Max-Min Semantic Chunking* yang digunakan untuk membagi dokumen hasil praproses menjadi satuan informasi berbasis kesamaan semantik antar kalimat. Proses dimulai dengan menerima masukan berupa *cleaned text* yang telah melalui tahap ekstraksi dan normalisasi pada subbab sebelumnya. Teks tersebut kemudian dipecah menjadi satuan kalimat melalui proses *sentence tokenization*. Setiap kalimat selanjutnya direpresentasikan ke dalam bentuk vektor semantik menggunakan model Qwen3-*Embedding*, sehingga masing-masing kalimat memiliki representasi numerik yang dapat dibandingkan secara matematis.

Setelah *embedding* berhasil dihasilkan, sistem melakukan inisialisasi *cluster* pertama dengan mengambil kalimat awal sebagai pusat *cluster*. Tahap berikutnya adalah menghitung nilai *similarity* antar kalimat berdasarkan *similarity search* antar *embedding*. Nilai kesamaan ini digunakan untuk menghitung *min-max similarity*, yaitu nilai minimum dan maksimum yang diamati dalam proses iterasi. Kedua nilai tersebut kemudian dipakai untuk menghasilkan *adaptive threshold*, sebuah ambang batas dinamis yang menentukan apakah suatu kalimat dianggap cukup mirip untuk digabungkan ke dalam *cluster* yang sama.

Proses pengambilan keputusan dilakukan melalui *node decision*, yaitu apakah nilai *similarity* terhadap *centroid cluster* ≥ *adaptive threshold*. Jika kondisi terpenuhi, kalimat tersebut dimasukkan ke dalam *chunk* yang sama. Sebaliknya, jika tidak memenuhi ambang batas, sistem membuat *chunk* baru. Pada kedua kasus tersebut, sistem akan melakukan *update centroid*, yaitu memperbarui representasi vektor pusat *cluster* berdasarkan kalimat terbaru yang ditambahkan. Proses ini memastikan bahwa representasi *cluster* tetap konstan dan relevan terhadap konten terkini.

Setelah pembaruan *centroid*, sistem memeriksa apakah seluruh kalimat telah diproses melalui keputusan Semua Kalimat Selesai?. Jika belum, proses kembali ke tahap perhitungan *similarity* untuk kalimat berikutnya. Jika seluruh kalimat sudah selesai diproses, maka sistem menghasilkan keluaran berupa *Semantic Chunk List*, yaitu kumpulan *chunk* yang telah terbentuk berdasarkan hubungan semantik antar kalimat. Metode ini memungkinkan pembentukan unit informasi yang lebih koheren dibandingkan *chunking* berbasis panjang karakter, sehingga meningkatkan kualitas konteks dalam sistem *Retrieval-Augmented Generation*.

4.1.2.3 Recursive Chunking



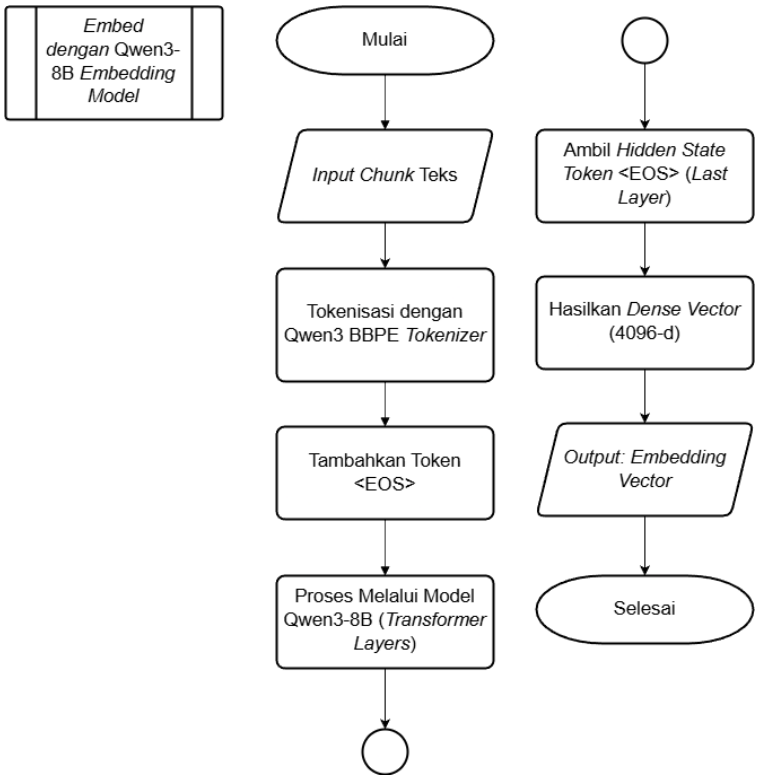
Gambar 4.5 *Flowchart Recursive Chunking*

Flowchart pada Gambar 4.5 menjelaskan mekanisme kerja metode *Recursive Chunking* yang digunakan untuk memecah dokumen menjadi potongan teks berukuran seragam berdasarkan pendekatan hierarkis. Proses dimulai dengan menerima masukan berupa *cleaned text* hasil tahap praproses. Sistem terlebih dahulu memeriksa apakah panjang teks sudah berada di bawah batas *chunk size* yang ditentukan. Jika terpenuhi, teks tersebut langsung dijadikan satu *chunk* dan proses selesai. Namun apabila panjang dokumen masih melampaui batas, sistem akan melanjutkan dengan melakukan inisialisasi daftar separator, yaitu urutan pemisah teks yang akan digunakan secara bertingkat, dimulai dari struktur yang paling besar hingga paling granular.

Pada tahap berikutnya, sistem mengambil separator prioritas saat ini dan melakukan pemisahan dokumen menggunakan separator tersebut. Hasil pemisahan kemudian dievaluasi melalui proses checking, yaitu pemeriksaan apakah seluruh fragmen yang terbentuk memiliki panjang yang lebih kecil atau sama dengan *chunk size*. Jika seluruh fragmen telah memenuhi syarat, sistem akan menggabungkan fragmen-fragmen tersebut kembali dengan mempertahankan struktur *overlap* untuk menjaga konteks antarpotongan, kemudian menghasilkan *Recursive Chunk List* sebagai keluaran.

Jika hasil pemisahan tidak memenuhi syarat, sistem akan memeriksa apakah separator yang digunakan merupakan separator terakhir dalam daftar. Apabila masih terdapat separator lain, sistem akan mengambil separator berikutnya dan mengulangi proses pemisahan. Namun jika seluruh separator telah digunakan dan pemisahan masih belum menghasilkan fragmen yang cukup kecil, sistem menerapkan strategi *force split* (*character-level*), yaitu pemotongan paksa berdasarkan jumlah karakter tanpa mempertimbangkan struktur dokumen. Potongan-potongan hasil *force split* kemudian digabungkan kembali dengan penyesuaian *overlap* hingga menghasilkan *chunk* yang memenuhi ukuran maksimal.

4.1.3 Embedding

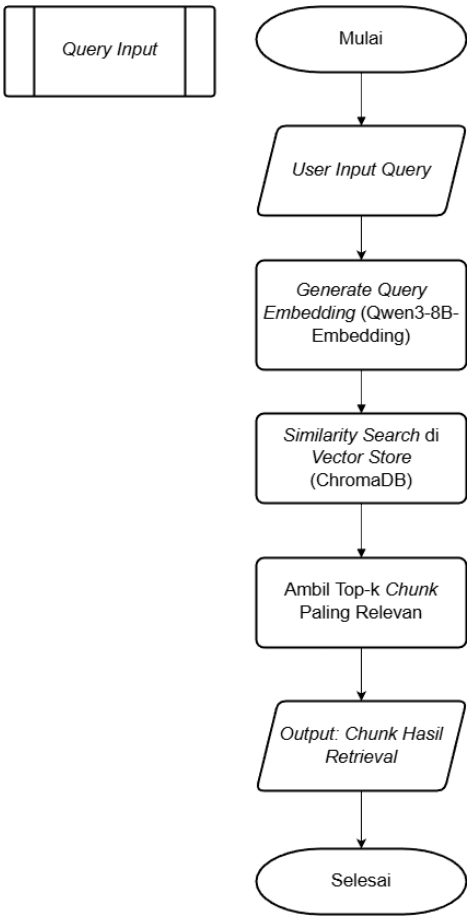


Gambar 4.6 *Flowchart Embedding*

Flowchart pada Gambar 4.6 menggambarkan proses *embedding* yang digunakan untuk mengonversi setiap *chunk* teks menjadi representasi vektor berdimensi tinggi. Tahap ini dimulai dengan menerima masukan berupa *chunk* teks hasil proses chunking sebelumnya. Teks tersebut terlebih dahulu melalui proses tokenisasi menggunakan *Byte*-level BPE (BBPE) *Tokenizer* milik Qwen3, yang berfungsi memecah teks menjadi unit token secara konsisten dan robust terhadap beragam karakteristik bahasa. Setelah proses tokenisasi selesai, model menambahkan token khusus <EOS> pada akhir urutan token sebagai penanda akhir teks dan sebagai titik referensi utama untuk ekstraksi *embedding*.

Selanjutnya, rangkaian token yang telah dipersiapkan diproses melalui seluruh lapisan transformer pada model Qwen3-8B, menghasilkan representasi laten (*hidden states*) untuk setiap token. Pada akhir proses forward pass, sistem mengambil *hidden state* dari token <EOS> pada lapisan terakhir, karena *hidden state* tersebut merepresentasikan ringkasan konteks global dari seluruh teks yang diproses. Representasi inilah yang kemudian diproyeksikan menjadi sebuah dense *vector* berdimensi 4096, yang menjadi *embedding* akhir dari *chunk* tersebut. *Embedding* ini kemudian digunakan dalam tahap *retrieval* untuk melakukan pencarian kemiripan semantik di dalam *vector store*.

4.1.4 Query Input

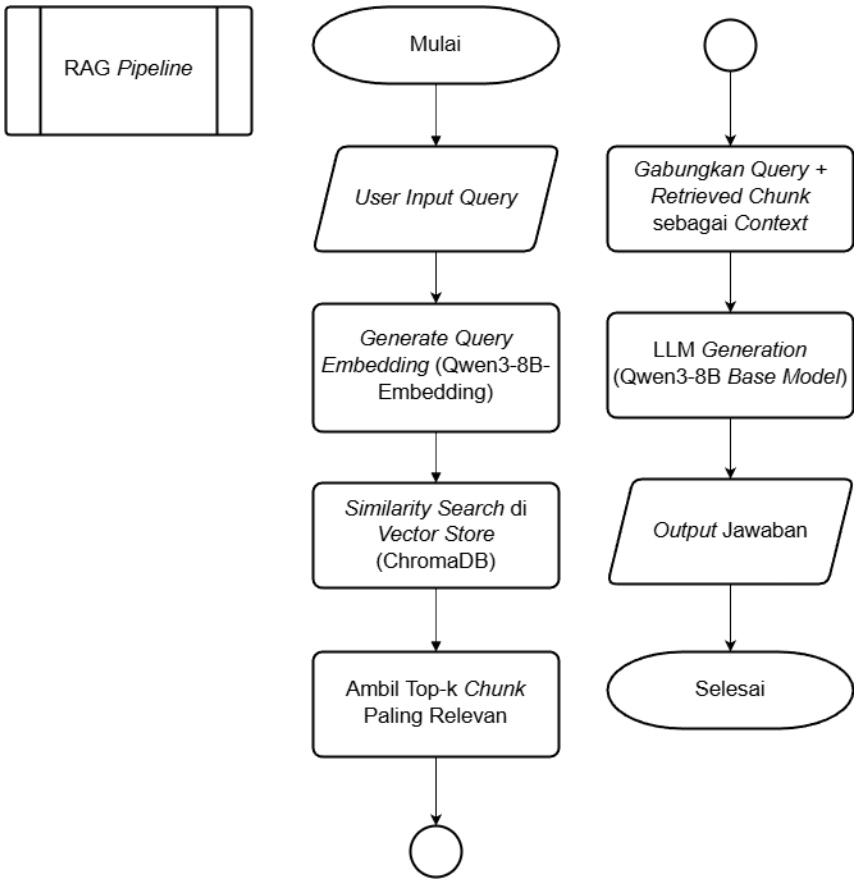


Gambar 4.7 *Flowchart Query Input*

Flowchart pada Gambar 4.7 menggambarkan alur pemrosesan *query* yang dilakukan ketika pengguna berinteraksi dengan sistem RAG. Proses dimulai ketika pengguna memasukkan *query* berupa pertanyaan atau pernyataan yang ingin dicari jawabannya. *Query* tersebut kemudian dikonversi menjadi representasi vektor melalui model Qwen3-8B-*Embedding*, sehingga sistem dapat memahami makna semantik dari input pengguna dalam bentuk numerik. Representasi vektor ini digunakan sebagai dasar untuk melakukan pencarian berbasis kemiripan (*similarity search*) pada *vector store* ChromaDB, yang sebelumnya telah berisi kumpulan *embedding* dari seluruh *chunk* dokumen.

Setelah proses pencarian dilakukan, sistem mengambil sejumlah *chunk* dengan nilai kemiripan tertinggi (top-k), yang dianggap paling relevan dengan konteks *query* pengguna. *Chunk* inilah yang kemudian menjadi *retrieved context* dan digunakan pada tahap berikutnya, yaitu proses *generation* oleh model bahasa untuk menghasilkan jawaban akhir. Dengan mekanisme ini, sistem dapat memastikan bahwa model hanya menerima konteks yang benar-benar relevan, sehingga meningkatkan akurasi dan konsistensi jawaban.

4.1.5 RAG *Pipeline*

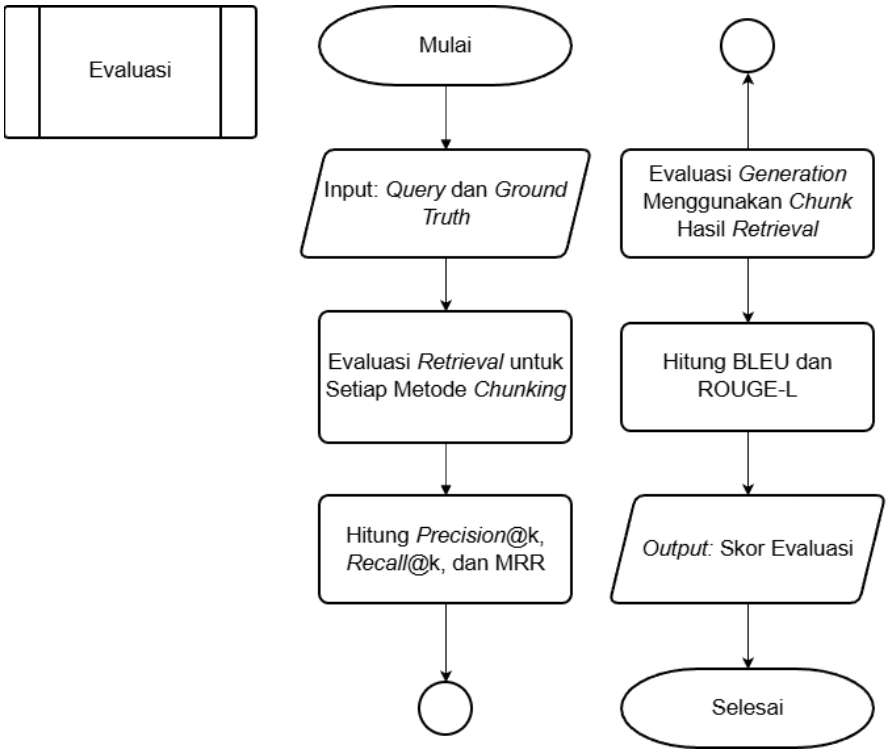


Gambar 4.8 *Flowchart* RAG *Pipeline*

Flowchart pada Gambar 4.8 menggambarkan alur kerja lengkap sistem *Retrieval-Augmented Generation* (RAG) yang digunakan dalam penelitian ini. Proses dimulai ketika pengguna memasukkan sebuah *query* sebagai masukan sistem. *Query* tersebut kemudian diubah menjadi representasi vektor menggunakan Qwen3-8B-*Embedding*, sehingga makna semantis dari teks dapat dipetakan ke dalam ruang vektor berdimensi tinggi. Vektor *query* yang telah terbentuk digunakan untuk melakukan pencarian berbasis kemiripan (*similarity search*) pada ChromaDB, yang sebelumnya telah berisi *embedding* dari seluruh *chunk* hasil pemrosesan dokumen BPS.

Tahap pencarian ini menghasilkan sejumlah *chunk* dengan tingkat kemiripan tertinggi (top-k results), yang dianggap paling relevan dengan kebutuhan informasi pengguna. *Chunk-chunk* yang berhasil ditemukan kemudian digabungkan dengan *query* asli untuk membentuk konteks komprehensif yang akan diberikan kepada model generatif. Pada tahap berikutnya, konteks tersebut diproses oleh Qwen3-8B Base Model untuk menghasilkan jawaban yang informatif, sesuai, dan berdasarkan dokumen yang relevan. Hasil akhir berupa jawaban inilah yang disajikan kembali kepada pengguna sebagai output sistem.

4.1.6 Evaluasi



Gambar 4.9 *Flowchart* Evaluasi

Flowchart pada Gambar 4.9 menggambarkan prosedur evaluasi yang digunakan untuk menilai kinerja sistem *Retrieval-Augmented Generation* (RAG) yang dikembangkan pada penelitian ini. Proses evaluasi diawali dengan menyiapkan *query* dan *ground truth* sebagai acuan pembanding. Tahap pertama adalah evaluasi *retrieval*, di mana sistem melakukan pencarian *chunk* relevan menggunakan ketiga metode chunking yang telah dirancang. Hasil *retrieval* dari setiap metode kemudian dianalisis menggunakan metrik *Precision*@k, *Recall*@k, dan *Mean Reciprocal Rank* (MRR) untuk mengukur ketepatan, kelengkapan, dan kualitas peringkat hasil pencarian. Ketiga metrik ini memberikan gambaran yang komprehensif mengenai kemampuan sistem dalam menemukan informasi yang relevan.

Tahap berikutnya adalah evaluasi *generation*, yang bertujuan menilai kualitas jawaban yang dihasilkan model Qwen3-8B setelah menerima *chunk* hasil *retrieval* sebagai konteks. Pada tahap ini, keluaran model dibandingkan dengan *ground truth* menggunakan dua metrik evaluasi teks, yaitu BLEU dan ROUGE-L. Metrik BLEU digunakan untuk mengukur kesamaan n-gram antara jawaban sistem dan jawaban referensi, sedangkan ROUGE-L menilai kecocokan berdasarkan *longest common subsequence*. Hasil pengukuran kedua metrik ini memberikan indikasi sejauh mana respons sistem sesuai dan informatif. Seluruh skor evaluasi dari kedua tahap ini kemudian disimpan dan digunakan sebagai dasar analisis hasil pada Bab 6.

4.2 Proses Manualisasi

DAFTAR REFERENSI

Aadit Kshirsagar, 2024. Enhancing RAG Performance Through *Chunking* and *Text Splitt*ing Techniques. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 10(5), pp.151–158. https://doi.org/10.32628/cseit2410593.

Ajay Mukund, S. and Easwarakumar, K.S., 2025. Optimizing Legal *Text* Summarization Through Dynamic *Retrieval-Augmented Generation* and Domain-Specific Adaptation. *Symmetry*, 17(5). https://doi.org/10.3390/sym17050633.

Amiri, M. and Bocklitz, T., 2025. *Chunk* Twice, *Embed* Once: A Systematic Study of Segmentation and Representation Trade-offs in Chemistry-Aware *Retrieval-Augmented Generation*. [online] Available at: <http://arxiv.org/abs/2506.17277>.

Anassai, B.R. and Josaphat, P., 2024. *Pembangunan Chatbot Sistem Informasi KBLI dan KBJI Berbasis LLM (Development of LLM-Based KBLI and KBJI Information System Chatbot)*.

Anugrah, I.G., 2021. Penerapan Metode N-Gram dan Cosine *Similarity* Dalam Pencarian Pada Repositori Artikel Jurnal Publikasi. *Building of Informatics, Technology and Science (BITS)*, 3(3), pp.275–284. https://doi.org/10.47065/bits.v3i3.1058.

Barnett, S., Kurniawan, S., Thudumu, S., Brannelly, Z. and Abdelrazek, M., 2024. Seven Failure Points When Engineering a *Retrieval Augmented Generation* System. *CEUR Workshop Proceedings*, [online] 2657, pp.1–9. Available at: <http://arxiv.org/abs/2401.05856>.

Bhat, S.R., Rudat, M., Spiekermann, J. and Flores-Herr, N., 2025. Rethinking *Chunk Size* For Long-Document *Retrieval*: A Multi-Dataset Analysis. [online] Available at: <http://arxiv.org/abs/2505.21700>.

Cheng, M., Luo, Y., Ouyang, J., Liu, Q.I., Liu, H., Li, L.I., Yu, S., Zhang, B., Cao, J., Ma, J., Wang, D., Chen, E., Liu, Q. and Li, L., 2018. *A Survey on Knowledge-Oriented Retrieval-Augmented Generation*. [online] https://doi.org/XXXXXXX.XXXXXXX.

Danter, D., Mühle, H. and Stöckl, A., 2024. Advanced *Chunking* and *Search* Methods for Improved *Retrieval-Augmented Generation* (RAG) System Performance in E-Learning. In: *Human Factors in Design, Engineering, and Computing*. AHFE International. https://doi.org/10.54941/ahfe1005756.

Es, S., James, J., Espinosa-Anke, L., Schockaert, S. and Gradients, E., 2024. *RAGAS: Automated Evaluation of Retrieval Augmented Generation*. [online] Available at: <https://platform.openai.com>.

Evtikhiev, M., Bogomolov, E., Sokolov, Y. and Bryksin, T., 2023. Out of the BLEU: how should we assess quality of the Code *Generation* models? *CEUR*

Workshop Proceedings, [online] 2657, pp.1–9. https://doi.org/10.1016/j.jss.2023.111741.

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S. and Li, Q., 2024. A Survey on RAG Meeting LLMs: Towards *Retrieval-Augmented* Large Language Models. [online] Available at: <http://arxiv.org/abs/2405.06211>.

Ferrante, M., Ferro, N. and Fuhr, N., 2021. Towards Meaningful Statements in IR Evaluation. Mapping Evaluation Measures to Interval Scales. [online] Available at: <http://arxiv.org/abs/2101.02668>.

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M. and Wang, H., 2024. *Retrieval-Augmented Generation* for Large Language Models: A Survey. [online] Available at: <http://arxiv.org/abs/2312.10997>.

Gargari, O.K. and Habibi, G., 2025. *Enhancing medical AI with retrieval-augmented generation: A mini narrative review*. *Digital Health*, https://doi.org/10.1177/20552076251337177.

Günther, M., Mohr, I., Williams, D.J., Wang, B. and Xiao, H., 2025. Late *Chunking*: Con*text*ual *Chunk Embedding*s Using Long-*Context Embedding* Models. [online] Available at: <http://arxiv.org/abs/2409.04701>.

Jeong, M., Sohn, J., Sung, M. and Kang, J., 2024. Improving Medical Reasoning through *Retrieval* and Self-Reflection with *Retrieval-Augmented* Large Language Models. *Bioinformatics*, [online] 33(16), pp.1–7. Available at: <http://arxiv.org/abs/2401.15269>.

Kiss, C., Nagy, M. and Szilágyi, P., 2025. Max–Min *semantic chunking* of documents for RAG application. *Discover Computing*, 28(1). https://doi.org/10.1007/s10791-025-09638-7.

Mastropaolo, A., Ch, A.M., Ciniselli, M., Penta, M. Di and Bavota, G., 2024. *Evaluating Code Summarization Techniques: A New Metric and an Empirical Characterization*. [online] *Proceedings of 46th International Conference on Software Engineering (ICSE 2024)*, https://doi.org/XXXXXXX.XXXXXXX.

Matarazzo, A. and Torlone, R., 2025. A Survey on Large Language Models with some Insights on their Capabilities and Limitations. [online] Available at: <http://arxiv.org/abs/2501.04040>.

Merola, C. and Singh, J., 2025. Reconstructing *Context*: Evaluating Advanced *Chunking* Strategies for *Retrieval-Augmented Generation*. [online] Available at: <http://arxiv.org/abs/2504.19754>.

Mikulic, I., Vlaic, M., Delac, G., Silic, M. and Vladimir, K., 2025. Integrating External Knowledge with LLMs: A Systematic Review of RAG Approaches. In: *2025 MIPRO 48th ICT and Electronics Convention, MIPRO 2025 - Proceedings*. Institute of Electrical and Electronics Engineers Inc. pp.93–98. https://doi.org/10.1109/MIPRO65660.2025.11131735.

Murtiyoso, M., Tahyudin, I. and Berlilana, B., 2025. A Systematic Review of *Retrieval-Augmented Generation* for Enhancing Domain-Specific Knowledge in Large Language Models. *Sinkron*, 9(2), pp.969–977. https://doi.org/10.33395/sinkron.v9i2.14824.

Nguyen, H.T., Nguyen, T.D. and Nguyen, V.H., 2025. Enhancing *Retrieval Augmented Generation* with Hierarchical *Text* Segmentation *Chunking*. [online] Available at: <http://arxiv.org/abs/2507.09935>.

Oche, A.J., Folashade, A.G., Ghosal, T. and Biswas, A., 2025. A Systematic Review of Key *Retrieval-Augmented Generation* (RAG) Systems: Progress, Gaps, and Future Directions. [online] Available at: <http://arxiv.org/abs/2507.18910>.

Öztürk, E. and Mesut, A., 2024. PERFORMANCE ANALYSIS OF CHROMA, QDRANT, AND FAISS *DATABASE*S. *UNITECH – SELECTED PAPERS*. https://doi.org/10.70456/tbrn3643.

Pradana, A.S. and Fitrianah, D., 2024. *Text Preprocessing* Audit Findings of Financial Statements: Preparing the Data for Further Analysis. In: *7th International Seminar on Research of Information Technology and Intelligent Systems: Advanced Intelligent Systems in Contemporary Society, ISRITI 2024 - Proceedings*. Institute of Electrical and Electronics Engineers Inc. pp.842–846. https://doi.org/10.1109/ISRITI64779.2024.10963569.

Qu, R., Tu, R. and Bao, F., 2024. Is *Semantic Chunking* Worth the Computational Cost? [online] Available at: <http://arxiv.org/abs/2410.13070>.

Ru, D., Qiu, L., Hu, X., Zhang, T., Shi, P., Chang, S., Jiayang, C., Wang, C., Sun, S., Li, H., Zhang, Z., Wang, B., Jiang, J., He, T., Wang, Z., Liu, P., Zhang, Y. and Zhang, Z., 2024. *RAGCHECKER: A Fine-grained Framework for Diagnosing Retrieval-Augmented Generation*. [online] Available at: <https://www.bing.com/chat>.

Sadeli, A.F. and Lawanda, I.I., 2023. *Recall*, *Precision*, and F-Measure for Evaluating Information *Retrieval* System in Electronic Document Management Systems (EDMS). *Khizanah al-Hikmah : Jurnal Ilmu Perpustakaan, Informasi, dan Kearsipan*, 11(2), pp.231–241. https://doi.org/10.24252/kah.v11i2a8.

Sarmah, B., Hall, B., Rao, R., Patel, S., Pasquali, S. and Mehta, D., 2024. HybridRAG: Integrating Knowledge Graphs and *Vector Retrieval Augmented Generation* for Efficient Information Extraction. [online] Available at: <http://arxiv.org/abs/2408.04948>.

Shaik, N., Chitralingappa, P. and Harichandana, B., 2024. The Nexus of AI and *Vector Database*s: Revolutionizing NLP with LLMs. *International Journal of Scientific Research in Engineering and Management*. [online] https://doi.org/10.55041/IJSREM35419.

Singh, I.S., Aggarwal, R., Allahverdiyev, I., Taha, M., Akalin, A., Zhu, K. and O'Brien, S., 2025. *Chunk*RAG: Novel LLM-*Chunk* Filtering Method for RAG Systems. [online] Available at: <http://arxiv.org/abs/2410.19572>.

Singh, P.N., Talasila, S. and Banakar, S.V., 2023. Analyzing *Embedding* Models for *Embedding Vector*s in *Vector Database*s. In: *3rd IEEE International Conference on ICT in Business Industry and Government, ICTBIG 2023*. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ICTBIG59752.2023.10455990.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z. and Qiu, Z., 2025. Qwen3 Technical Report. [online] Available at: <http://arxiv.org/abs/2505.09388>.

Yepes, A.J., You, Y., Milczek, J., Laverde, S. and Li, R., 2024. Financial Report *Chunking* for Effective *Retrieval Augmented Generation*. [online] Available at: <http://arxiv.org/abs/2402.05131>.

Yu, H., Gan, A., Zhang, K., Tong, S., Liu, Q. and Liu, Z., 2024. Evaluation of *Retrieval-Augmented Generation*: A Survey. [online] https://doi.org/10.1007/978-981-96-1024-2_8.

Zhang, Y., Li, M., Long, D., Zhang, X., Lin, H., Yang, B., Xie, P., Yang, A., Liu, D., Lin, J., Huang, F. and Zhou, J., 2025. Qwen3 *Embedding*: Advancing *Text Embedding* and Reranking Through Foundation Models. [online] Available at: <http://arxiv.org/abs/2506.05176>.

Zhao, J., Ji, Z., Feng, Y., Qi, P., Niu, S., Tang, B., Xiong, F. and Li, Z., 2025. Meta-*Chunking*: Learning *Text* Segmentation and *Semantic* Completion via Logical Perception. [online] Available at: <http://arxiv.org/abs/2410.12788>.