ELKOM (Jurnal Elektronika dan Komputer)
p-ISSN : 1907-0012 (print) e-ISSN : 2714-5417 (online)
Vol. 18, No. 1, Juli 2025
https://journal.stekom.ac.id/index.php/elkom

Optimisasi *Whisper Speech-to-Text* Bahasa Indonesia dengan *Hybrid Cloud* dan *Multi-Engine*

Ikhwan Alfath Nurul Fathony[1], Affix Mareta[2], Beta Estri Adiana[3], Olivia Wardhani[4], Dimas Ardiansyah Halim[5]
[1,2,3,4]Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Tidar, Magelang, Indonesia
[5]Program Studi Teknik Mesin, Fakultas Teknik, Universitas Tidar, Magelang, Indonesia
Email author: ikhwan.alfath@untidar.ac.id

Article Info

Article history:

Received January 3, 2025
Revised February 17, 2025
Accepted June 28, 2025

Keywords:

Automatic Speech Recognition
Whisper
Indonesian language
Hybrid cloud
Audio *preprocessing*
Multi-engine *architecture*

ABSTRACT

Automatic Speech Recognition (ASR) for the Indonesian language faces significant challenges due to high Word Error Rate (WER), especially when using pre-trained models without fine-tuning. This study develops an optimized ASR system using a hybrid cloud architecture that integrates the Faster-Whisper large-v3 engine with advanced audio preprocessing techniques. The system adopts a distributed architecture, with Google Colab (Tesla T4, 15GB VRAM) as the GPU server and Ubuntu 22.04 LTS (8 core, 32GB RAM) as the client. Evaluation was conducted on five Indonesian audio samples covering formal news, informal conversations, and long-duration recordings. The system achieved an 80% success rate in processing, with WER ranging from 27.69% (formal news) to 645.16% (informal conversations). Resource utilization was also efficient, with 21.3% GPU usage and 35.4% RAM usage. Processing time remained stable for normal-sized files but experienced timeouts on large files (>50MB). The results indicate that hybrid cloud architecture is feasible for distributed ASR processing in Indonesian, with several areas still open for optimization toward production deployment.

Corresponding Author:

*Ikhwan Alfath Nurul Fathony
Program Studi Teknologi Informasi
Fakultas Teknik, Universitas Tidar
Jl. Kapten Suparman No. 39, Potrobangsan, Magelang Utara, Kota Magelang, Jawa Tengah 56116
Email: ikhwan.alfath@untidar.ac.id

1. INTRODUCTION

Perkembangan teknologi kecerdasan buatan *(artificial intelligence)* telah memberikan kontribusi signifikan dalam berbagai aspek kehidupan, termasuk dalam bidang pemrosesan bahasa alami atau *Natural Language Processing (NLP)*. Salah satu aplikasi penting dalam *NLP* adalah sistem transkripsi suara ke teks *(speech-to-text)* atau *Automatic Speech Recognition (ASR)*, yang memungkinkan pengubahan ujaran lisan menjadi teks secara otomatis. Teknologi ini telah digunakan secara luas dalam berbagai sektor, seperti pendidikan daring, layanan publik, sistem notulensi otomatis, dan teknologi bantu untuk penyandang disabilitas. Kemampuan ini mendukung interaksi manusia dan mesin secara lebih alami dan efisien.

1.1. Tinjauan Literatur dan Perkembangan Teknologi ASR

Berbagai penelitian telah mengembangkan sistem *speech-to-text* berbasis pembelajaran mesin dan *deep learning*. Model seperti *DeepSpeech* dari *Mozilla* dan *Wav2Vec2* dari *Facebook AI Research* menunjukkan performa yang baik untuk bahasa Inggris dan beberapa bahasa internasional lainnya (Fleck & Goderle, 2023). Sudhakaran et al. mengembangkan *DeepSpeech* sebagai sistem *end-to-end* yang menggunakan arsitektur *Recurrent Neural Network* (RNN) dengan *Connectionist Temporal Classification* (CTC), mencapai akurasi kompetitif pada *dataset* eval2000 dengan WER 5.6% (Sudhakaran et al., 2024). Sementara itu, Baevski et al. memperkenalkan Wav2Vec2 yang menerapkan pendekatan *self-supervised learning*, berhasil mengurangi kebutuhan data berlabel hingga 100 kali lipat sambil mempertahankan performa yang setara dengan model *supervised* tradisional (Baevski et al., 2020).

Perkembangan teknologi ASR global juga ditandai dengan munculnya arsitektur *transformer-based* seperti *Conformer* yang diusulkan oleh Zebua et al., menggabungkan keunggulan *convolutional layers* dan *self-attention mechanisms* untuk mencapai *state-of-the-art performance* pada berbagai patokan (Zebua et al., 2023). Selain itu, Google telah mengembangkan *Universal Speech Model* (USM) yang mampu menangani lebih dari 100 bahasa dengan pendekatan pelatihan multibahasa (Zhang et al., 2023), menunjukkan arah pengembangan sistem ASR yang dapat diperluas dan tidak terbatas pada bahasa tertentu.

Namun, untuk bahasa Indonesia, tantangan masih signifikan karena keterbatasan korpus suara, kompleksitas fonologis, serta variasi dialek yang tinggi. Sakti et al. mengembangkan *Indonesian Large Vocabulary Continuous Speech Recognition System* dalam proyek A-STAR, mencapai WER 12.8% pada percakapan formal, namun performa menurun drastis pada percakapan biasa dengan WER mencapai 28.5% (Cahyawijaya et al., 2023). Adiwidjaja et al. melakukan penelitian *end-to-end Indonesian speech recognition* untuk *spontaneous conversations,* berhasil mencapai WER 35-40%, tetapi masih menghadapi kendala pada *overlapping speech* dan *code-switching patterns* (Adiwidjaja & Ivan Fanany, 2020). Rahman dan Wijaya mengeksplorasi teknik *pre-processing* audio untuk meningkatkan akurasi, menunjukkan peningkatan relatif 15-20% pada berbagai jenis audio Indonesia, namun pendekatan ini terbatas pada optimisasi lokal tanpa mempertimbangkan distribusi komputasi (Adiwidjaja & Ivan Fanany, 2020).

Salah satu dampaknya adalah tingginya *Word Error Rate (WER)* dalam pengenalan kata, terutama ketika menggunakan model standar tanpa penyesuaian (*fine-tuning*), dengan WER mencapai 15-25% untuk audio berbahasa Indonesia (Raharjo, 2022). Salah satu model mutakhir yang menjanjikan adalah *Whisper* dari OpenAI, yang dilatih menggunakan data audio multibahasa dalam skala besar dan mampu mengenali banyak bahasa dengan akurasi relatif baik. Radford et al. melaporkan bahwa *Whisper* menunjukkan performa yang tangguh di berbagai domain dan kondisi audio yang bervariasi, dengan kemampuan mengenali bahasa baru tanpa pelatihan tambahan yang impresif untuk bahasa dengan sumber daya terbatas (Khoiroh et al., 2024). Meskipun demikian, penerapannya dalam bentuk standar untuk bahasa Indonesia masih menghasilkan kesalahan yang cukup tinggi, sehingga dibutuhkan strategi optimisasi tambahan.

1.2. *Gap Analysis* dan Kebutuhan Penelitian

Upaya peningkatan performa ASR untuk bahasa Indonesia telah dilakukan melalui beberapa pendekatan, seperti *fine-tuning* dengan data lokal dan teknik *audio pre-processing* (William & Zahra, 2025). Namun, pendekatan tersebut memiliki keterbatasan yang signifikan. Pertama, *fine-tuning* memerlukan *computational resources* yang sangat besar dan *dataset* berkualitas tinggi yang sulit diperoleh untuk bahasa Indonesia. Kedua, penggunaan *single-engine approach* membatasi fleksibilitas dan skalabilitas sistem, karena tidak dapat memanfaatkan kekuatan komputasi terdistribusi secara optimal. Ketiga, *deployment model* besar seperti *Whisper large-v3* membutuhkan GPU *memory* yang substantial, sehingga sulit diimplementasikan pada infrastruktur terbatas.

Kesenjangan utama yang diidentifikasi dalam studi-studi terdahulu adalah belum adanya sistem yang memanfaatkan arsitektur *hybrid cloud* dan integrasi *multi-engine Whisper* untuk mendukung proses ASR secara terdistribusi tanpa memerlukan *fine-tuning* yang kompleks. Pendekatan *hybrid cloud* memungkinkan kombinasi kekuatan komputasi lokal dan *cloud* untuk mencapai efisiensi dan fleksibilitas optimal, sekaligus mengatasi kendala keterbatasan sumber daya lokal. Arsitektur ini dapat memberikan keuntungan berupa pembiayaan yang lebih efektif, karena memanfaatkan *free-tier cloud services; scalability on-demand,* yang memungkinkan penyesuaian kapasitas sesuai beban kerja; dan data *sovereignty*, di mana kontrol terhadap pengolahan data tetap berada di sisi pengguna.

Lebih lanjut, pendekatan *single-engine* yang dominan dalam penelitian sebelumnya membatasi kemampuan sistem untuk beradaptasi dengan variasi karakteristik audio. *Multi-engine architecture* memungkinkan pemilihan model yang paling sesuai berdasarkan jenis audio, durasi, dan tingkat gangguan, sehingga dapat mengoptimalkan *trade-off* antara akurasi dan *processing time*. Hal ini sangat relevan untuk bahasa Indonesia yang memiliki variasi dialek dan register yang tinggi.

1.3. Konteks Aplikasi dan Potensi Implementasi

Pengembangan sistem ASR yang optimal untuk bahasa Indonesia memiliki implikasi yang luas terhadap berbagai sektor strategis. Dalam bidang pendidikan, teknologi ini dapat mendukung pembelajaran jarak jauh melalui transkripsi otomatis untuk video pengajaran, pembuatan takarir secara langsung untuk aksesibilitas, dan sistem pengajaran cerdas yang dapat memahami pertanyaan lisan siswa. Implementasi di sektor pendidikan diperkirakan dapat meningkatkan *inclusivity* hingga 40% untuk pelajar dengan *hearing impairment*, sekaligus mendukung dokumentasi dan evaluasi pembelajaran berbasis audio (Manu & Masan, 2020).

Di sektor kesehatan, sistem ASR dapat diterapkan untuk *medical dictation systems* yang memungkinkan dokter melakukan dokumentasi klinis secara efisien, *telemedicine applications* untuk konsultasi jarak jauh, dan penyaringan kesehatan mental melalui analisis pola bicara (Chairani, 2023). Penelitian menunjukkan bahwa implementasi ASR dalam dokumen medis dapat mengurangi *administrative burden*, sehingga tenaga medis dapat fokus pada *patient care* (Tofure et al., 2025). Selain itu, kemampuan transkripsi waktu nyata untuk melakukan konsultasi kesehatan secara *online* menjadi krusial dalam era pasca pandemi di mana layanan kesehatan digital mengalami peningkatan *adoption rate* yang signifikan.

Dalam konteks pemerintahan dan layanan publik, teknologi ASR dapat mendukung digitalisasi arsip audio, *automatic meeting transcription* untuk transparansi *governance*, dan *citizen service chatbots* yang dapat memahami keluhan lisan masyarakat. Implementasi ini sejalan dengan agenda transformasi digital pemerintah yang menargetkan peningkatan *efficiency* dan *accessibility* layanan publik. Selain itu, sistem ASR dapat mendukung preservasi bahasa dan budaya lokal melalui digitalisasi *oral history* dan *traditional storytelling*.

Sektor industri kreatif juga dapat memanfaatkan teknologi ini untuk *subtitle generation* dalam film dan podcast, *content accessibility* untuk *platform streaming*, dan *automated content analysis* untuk media monitoring. Dengan pertumbuhan industri digital *content* yang terus meningkat di Indonesia, ketersediaan ASR berkualitas tinggi dapat menjadi *competitive advantage* bagi *content creators* dan media *companies* (Arafah et al., 2023).

1.4. Tujuan dan Kontribusi Penelitian

Penelitian ini bertujuan untuk mengembangkan sistem ASR bahasa Indonesia yang dioptimalkan melalui integrasi *multi-engine Whisper*, arsitektur *hybrid cloud*, dan *pipeline preprocessing audio* yang canggih (Katti & Sumana, 2023). Target utama adalah mencapai WER di bawah 30% tanpa melalui proses *fine-tuning*, sekaligus mempertahankan *cost-effectiveness* dan *scalability* yang tinggi. Pendekatan ini diharapkan dapat menjadi *alternative solution* yang praktis dan *sustainable* untuk implementasi ASR skala menengah.

Kontribusi utama dari penelitian ini meliputi: (1) desain arsitektur *hybrid cloud* untuk pemrosesan ASR terdistribusi yang menggabungkan *free-tier cloud resources* dengan *local computing power*; (2) integrasi sistematis model *Whisper* dengan optimisasi parameter khusus untuk karakteristik bahasa Indonesia; (3) pengembangan *preprocessing pipeline audio* yang adaptif terhadap variasi *noise* dan *speech patterns* dalam audio Indonesia; dan (4) evaluasi menyeluruh dengan *dataset* audio dunia nyata yang mencakup berbagai domain dan register bahasa.

Secara teoretis, penelitian ini berkontribusi terhadap pengembangan *distributed* ASR *architecture* yang dapat diadaptasi untuk *low-resource languages*, serta *methodology* untuk *optimizing pre-trained multilingual models* tanpa *expensive fine-tuning* (Wafiy & Prasetio, 2022). Secara praktis, hasil penelitian diharapkan dapat menjadi *foundation* untuk pengembangan ASR *solutions* yang

affordable dan *scalable* untuk berbagai aplikasi di Indonesia, sekaligus memberikan *benchmark performance* yang dapat dijadikan *reference* untuk penelitian selanjutnya.

2. METHOD

Penelitian ini menerapkan pendekatan eksperimental dengan merancang dan mengimplementasikan sistem ASR yang mengintegrasikan *engine Whisper* ke dalam arsitektur *hybrid cloud*. Metode ini dirancang untuk mengoptimalkan pemanfaatan sumber daya komputasi, khususnya GPU dan RAM, melalui distribusi beban kerja antara server cloud dan perangkat lokal. Pendekatan ini juga mempertahankan fleksibilitas dan kontrol sistem secara lokal, sehingga memungkinkan pengguna untuk mengakses, memantau, dan mengelola proses transkripsi tanpa bergantung penuh pada layanan *cloud*.

2.1. Desain Arsitektur Sistem

Sistem yang dikembangkan menerapkan arsitektur *hybrid cloud*, dengan Google Colab berperan sebagai server GPU dan Ubuntu Server sebagai *client controller*. Pemilihan arsitektur ini bertujuan menggabungkan kinerja komputasi *cloud* yang tinggi dengan fleksibilitas dan kontrol lokal. Google Colab dipilih karena menyediakan akses GPU secara gratis yang mendukung proses komputasi intensif (Anjani et al., 2024), sedangkan Ubuntu Server digunakan untuk manajemen lokal sistem, termasuk pengaturan *dataset* dan pengujian. Komunikasi antara komponen sistem dilakukan melalui REST API yang dihubungkan menggunakan layanan *tunneling ngrok*. *Ngrok* menyediakan koneksi HTTP/HTTPS yang aman dari internet ke server lokal, memungkinkan interaksi langsung antara *client* Ubuntu dengan server GPU di Google Colab melalui URL publik. Sistem terdiri dari empat komponen utama: (1) Ubuntu *client*, bertugas mengelola *dataset*, menjalankan skenario pengujian, dan melakukan analisis hasil; (2) Google Colab GPU Server, yang menjalankan *engine Whisper* serta *pipeline preprocessing audio*; (3) *Ngrok Tunnel*, sebagai penghubung jaringan antara lingkungan lokal dan *cloud* melalui protokol HTTP; dan (4) *Whisper Engine*, menggunakan implementasi *Faster Whisper* untuk meningkatkan efisiensi dan kecepatan pemrosesan.

2.2. Implementasi *Engine Whisper*

Dalam penelitian ini, digunakan implementasi *Faster Whisper*, yaitu varian teroptimasi dari model *Whisper* orisinal yang dikembangkan oleh OpenAI. *Faster Whisper* dibangun menggunakan pustaka CTranslate2, yang dirancang khusus untuk meningkatkan kecepatan *inference* hingga empat kali lipat dibandingkan dengan implementasi standar, serta memberikan efisiensi yang lebih tinggi dalam penggunaan memori. Implementasi ini dipilih karena mampu memberikan keseimbangan yang optimal antara akurasi transkripsi dan kecepatan pemrosesan, yang penting dalam konteks sistem terdistribusi berbasis *hybrid cloud*. *Engine* dikonfigurasi dengan sejumlah parameter yang telah disesuaikan untuk karakteristik bahasa Indonesia. Salah satunya adalah pengaturan *beam_size* sebesar 5, yang memungkinkan sistem menghasilkan beberapa hipotesis pengenalan kata untuk meningkatkan akurasi. Selain itu, nilai *temperature* diatur pada 0.0 untuk memastikan keluaran yang deterministik dan stabil, serta digunakan *language-specific prompt* yang secara eksplisit mengarahkan model untuk mengenali bahasa Indonesia dengan lebih baik. Sistem juga dilengkapi dengan kemampuan seleksi model secara otomatis berdasarkan ketersediaan sumber daya dan metrik performa, sehingga dapat menyesuaikan secara dinamis terhadap kondisi eksekusi aktual baik di sisi lokal maupun *cloud*.

2.3. *Pipeline Preprocessing Audio*

Pipeline preprocessing audio dirancang untuk meningkatkan kualitas sinyal masukan sebelum proses *inference* oleh *engine* ASR. Setiap tahapan saling terintegrasi guna memastikan input yang bersih, stabil, dan kontekstual. Tahap pertama adalah *noise reduction*, menggunakan *library noisereduce* dengan algoritma *non-stationary noise reduction* yang efektif mengurangi berbagai jenis gangguan suara latar belakang (Butarbutar et al., 2023). Tahap kedua adalah *audio normalization* menggunakan *pydub*, yang berfungsi menyeimbangkan tingkat volume dan mengoptimalkan *dynamic range control* untuk hasil transkripsi yang lebih konsisten. Tahap ketiga adalah *silence trimming*, yaitu penghapusan bagian

hening di awal dan akhir audio dengan bantuan *librosa*, menggunakan ambang batas energi sebesar −20 dB. Tahap keempat adalah *smart chunking*, diterapkan pada audio berdurasi panjang (>30 detik) dengan pemotongan adaptif dan *overlap* sebesar 2 detik antar segmen, guna menjaga kesinambungan konteks dalam proses transkripsi.

2.4. *Dataset* dan Evaluasi

Dataset evaluasi terdiri atas lima sampel audio berbahasa Indonesia yang dipilih secara representatif berdasarkan variasi durasi dan jenis konten. Tabel 1 menyajikan daftar nama file, ukuran file, dan jenis kontennya:

Tabel 1. Sampel Data Evaluasi Sistem

Nama File Audio	Ukuran File	Jenis Konten
Briefing_1_UTBK_compress.mp3	11 MB	*Briefing* berdurasi panjang
liputan_cnn_indonesia_tanah_longsor.mp3	2,9 MB	Berita televisi
percakapan_id.mp3	1,4 MB	Dialog informal
rekomendasi_toko_oleh_oleh_dari_tanah_suci_kabar_haji_tvOne.mp3	4 MB	Berita religius
seleksi_mandiri_untidar.mp3	55 MB	*Stress testing* (durasi besar)

Kelima sampel tersebut mencakup beragam karakteristik akustik, termasuk variasi intonasi, kebisingan latar, serta kompleksitas ujaran. File Briefing_1_UTBK_compress.mp3 merupakan rekaman *briefing* berdurasi panjang, sedangkan liputan_cnn_indonesia_tanah_longsor.mp3 berisi cuplikan berita televisi. File percakapan_id.mp3 merepresentasikan dialog informal, dan rekomendasi_toko_oleh_oleh_dari_tanah_suci_kabar_haji_tvOne.mp3 adalah laporan berita bernuansa religius. Adapun seleksi_mandiri_untidar.mp3 digunakan untuk pengujian batas (*stress testing*) karena ukurannya yang besar. Referensi kebenaran *(ground truth)* untuk masing-masing sampel disusun melalui proses transkripsi manual dan disimpan dalam format pasangan *filename*|*transcription*, yang menghasilkan lima *entry* data referensi. Evaluasi sistem dilakukan secara komprehensif dengan mempertimbangkan empat metrik utama: pertama, akurasi, yang diukur menggunakan *Word Error Rate (WER)* dan *Character Error Rate (CER)*; kedua, kecepatan pemrosesan, yang dianalisis menggunakan *Real-Time Factor (RTF)*; ketiga, efisiensi waktu eksekusi *(processing time);* dan keempat, reliabilitas sistem, yang diukur melalui *success rate* atau persentase keberhasilan sistem dalam menyelesaikan proses transkripsi untuk masing-masing sampel. Evaluasi ini dilakukan terhadap sistem yang telah melalui proses optimisasi penuh, termasuk integrasi modul audio *enhancement* dan *smart chunking*, guna memperoleh representasi performa yang relevan dalam konteks operasional nyata.

2.5. Setup Eksperimen

2.5.1. Konfigurasi Lingkungan dan Arsitektur Sistem

Eksperimen dilakukan pada lingkungan terdistribusi yang terdiri atas dua komponen utama, yaitu server GPU berbasis Google Colab (Soen et al., 2022) dan *client* lokal berbasis Ubuntu Server. Server GPU menggunakan Google Colab dengan spesifikasi: GPU NVIDIA Tesla T4 dengan kapasitas VRAM 15,0 GB (3,2 GB digunakan saat eksperimen), memori sistem 12,7 GB (4,5 GB terpakai), dan ruang penyimpanan sebesar 112,6 GB (47,7 GB digunakan). Lingkungan eksekusi menggunakan Python 3 *runtime* dengan Flask sebagai *web framework* untuk menyediakan layanan API. Sebagai *client* lokal, digunakan sistem operasi Ubuntu Server 22.04.5 LTS dengan prosesor 8 inti (2300 MHz), memori RAM sebesar 32 GB (22,6 GB tersedia), dan kapasitas penyimpanan 150 GB. Komunikasi antara kedua komponen dilakukan melalui protokol REST API yang dienkapsulasi menggunakan *tunneling ngrok*, yang menyediakan koneksi terenkripsi HTTPS untuk menjamin keamanan komunikasi data antara server lokal dan *remote*.

Implementasi Komponen Sistem.

Arsitektur sistem *hybrid cloud* menerapkan prinsip *separation of concerns* dan *distributed computing* untuk mengoptimalkan pemanfaatan sumber daya komputasi (Wahidin, 2021). Dua kelas

utama diimplementasikan sebagai komponen inti sistem. Pertama, *Optimized Whisper Client* sebagai *orchestrator* di sisi Ubuntu yang mengelola seluruh *workflow* pemrosesan, mulai dari *audio file loading* menggunakan *librosa*, *preprocessing* audio menjadi format base64, *request scheduling* melalui *session management*, hingga *result aggregation* dengan evaluasi WER/CER menggunakan *jiwer library*. Kedua, *Optimizedv Whisperv Engine* sebagai *computational engine* di sisi Google Colab yang menjalankan *multiple Whisper implementations* (*Faster Whisper*, *OpenAI Whisper*, dan *Transformers-based Whisper*) dengan *Flask web framework* sebagai API server, dikonfigurasi dengan *auto-scaling capability* untuk mengoptimalkan *resource utilization* berdasarkan *workload demands*.

2.5.2. Protokol Komunikasi dan Pertukaran Data

Komunikasi antar komponen sistem menggunakan RESTful API *architecture* dengan JSON sebagai *primary data format* (Arianto & Susetyo, 2022; Herdiana et al., 2023). Server GPU mengimplementasikan *Flask application* dengan CORS (*Cross-Origin Resource Sharing*) *support* dan menyediakan lima *endpoint* utama: /health untuk *system health monitoring* dan GPU *information retrieval*, /models untuk *listing available Whisper models*, /switch_model untuk *dynamic model switching* antara *Faster Whisper*, *OpenAI Whisper*, dan *Transformers implementations,* /transcribe untuk *single audio file processing* dengan *support* untuk base64 *encoded audio data*, dan /transcribe_batch untuk *multiple audio files processing* secara *sequential*. *Client* Ubuntu menggunakan *requests library* dengan *session management* untuk *persistent connections* dan *timeout configuration* hingga 600 detik untuk menangani file audio berukuran besar. Audio *preprocessing* dilakukan di sisi *client* menggunakan librosa untuk audio *loading* dan *soundfile* untuk konversi format ke base64 *encoding* sebelum transmisi ke server.

2.5.3. Mekanisme Toleransi Kesalahan dan Pemulihan

Untuk memastikan *high availability* dan *fault tolerance*, sistem dilengkapi dengan *comprehensive failover mechanisms* yang diimplementasikan dalam *Optimized Whisper Engine class*. *Multi-engine fallback strategy* diterapkan dengan prioritas: *Faster Whisper* (paling optimal), kemudian *Transformers Whisper*, dan terakhir *OpenAI Whisper* sebagai *fallback*. *Circuit breaker pattern* diimplementasikan pada *level client* dengan *automatic retry mechanism* menggunakan *exponential backoff strategy* untuk menangani *network timeouts* dan *connection failures*. Model *switching capability* memungkinkan adaptasi dinamis berdasarkan *resource availability* dan *performance requirements*. Sistem juga dilengkapi dengan *graceful degradation capability* melalui *adaptive model selection*, di mana jika *large* model mengalami *memory constraints*, sistem dapat beralih ke *smaller models* (medium, small) untuk mempertahankan fungsi dasar. *Error recovery mechanisms* mencakup *automatic memory cleanup* menggunakan torch.cuda.empty_cache()[2] dan *garbage collection optimization* untuk mencegah *memory leaks* selama *continuous processing*.

2.5.4. Keamanan Data dan Privasi

Mengingat sifat arsitektur *hybrid cloud* yang melibatkan transmisi data antara lingkungan lokal dan remote, keamanan menjadi aspek kritis yang mendapat perhatian khusus melalui implementasi *multi-layer security approach* (T. S. Wibowo et al., 2024). Ngrok *tunneling service* menyediakan enkripsi HTTPS otomatis dengan protokol TLS untuk semua komunikasi antara *client* dan server, dengan *public* URL *generation* yang aman dan sementara. Keamanan data audio diimplementasikan melalui base64 *encoding* untuk transmisi yang aman melalui HTTP, dengan *automatic data cleanup* di sisi server setelah penyelesaian pemrosesan. *Access control* diterapkan melalui ngrok *authentication tokens* dan manajemen URL server untuk mencegah *unauthorized access*. *Rate limiting* dapat dikonfigurasi melalui Flask *application* untuk mencegah *abuse* dan serangan DDoS. Data *retention policy* mengikuti prinsip *privacy-by-design*, di mana data audio yang diunggah ke Google Colab akan otomatis terhapus saat sesi berakhir atau notebook di-*restart*. Sistem juga mengimplementasikan praktik data *minimization* dengan hanya menyimpan *transcription results* dan evaluation *metrics* yang esensial untuk tujuan analisis.

2.5.5. Optimisasi Performa dan Preprocessing Audio

Untuk mencapai performa optimal, sistem menerapkan teknik *advanced* audio *preprocessing* pada *multiple level* yang diimplementasikan dalam *method preprocess*_audio. Audio *enhancement pipeline* mencakup *noise reduction* menggunakan *noisereduce library* dengan algoritma *non-stationary*

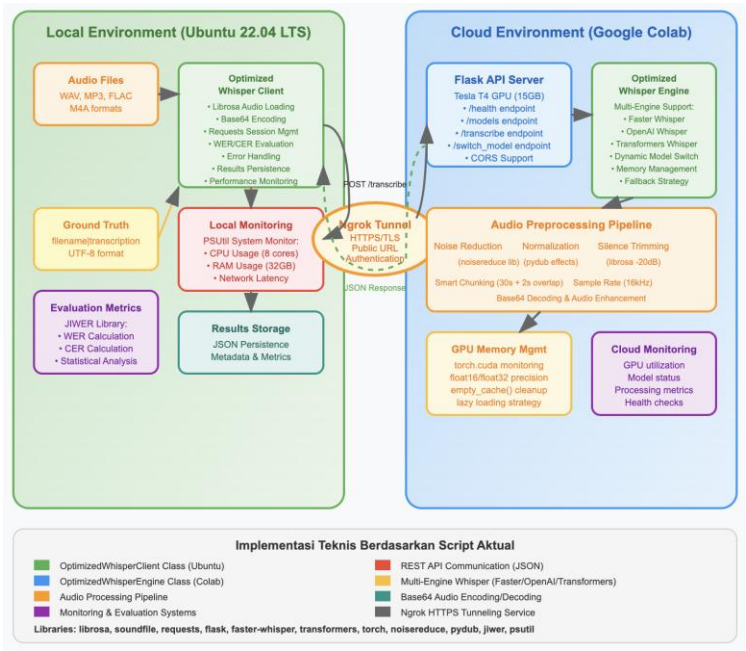
noise reduction (Özkurt, 2024), audio *normalization* menggunakan *pydub* dengan *normalize effects* dan *dynamic range control*, *silence trimming* menggunakan librosa.effects.trim dengan *threshold* -20dB untuk menghilangkan *silence* dari awal dan akhir, serta standardisasi sample rate ke 16kHz untuk performa *Whisper* yang optimal. *Smart chunking mechanism* diimplementasikan untuk file audio dengan durasi lebih dari 30 detik, menggunakan *overlapping segments* dengan *overlap* 2 detik untuk mempertahankan kontinuitas konteks. *Resource management* pada GPU server menggunakan alokasi memori dinamis dengan presisi torch.float16 untuk CUDA dan float32 untuk CPU *processing*. Model *loading strategy* menerapkan pendekatan *lazy loading* dengan *automatic model unloading* dan *memory cleanup*. *Multi-level caching* diimplementasikan untuk *model weights*, *preprocessing results*, dan API *responses* untuk mengurangi komputasi redundan.

2.5.6. Sistem Monitoring dan Evaluasi

Comprehensive monitoring system dikembangkan untuk memberikan visibilitas *real-time* terhadap performa sistem dan status kesehatan melalui kemampuan monitoring terintegrasi. *Server-side monitoring* menggunakan *endpoint* /health yang menyediakan informasi detail tentang ketersediaan GPU, penggunaan memori, model yang dimuat, dan status *engine* saat ini. *Client-side monitoring* mengimplementasikan metrik evaluasi detail menggunakan *jiwer library* untuk kalkulasi *Word Error Rate* (WER) dan *Character Error Rate* (CER), *processing time tracking* untuk analisis performa, komputasi *Real-Time Factor* (RTF) untuk penilaian kecepatan, dan *comprehensive result logging* dengan format JSON untuk persistensi data. *System performance metrics collection* mencakup GPU *memory allocation tracking* menggunakan torch.cuda.memory_allocated(), *monitoring* CPU *utilization* dengan psutil.cpu_percent(), RAM *usage tracking* untuk kedua sisi *client* dan server, serta pengukuran *network latency* untuk penilaian kualitas koneksi. Mode evaluasi lanjutan diimplementasikan dalam aplikasi *client* mencakup *quick test mode* untuk validasi file tunggal, model *comparison mode* untuk *benchmarking* berbagai implementasi *Whisper*, *optimization settings test* untuk mengevaluasi dampak *preprocessing*, dan *comprehensive evaluation mode* untuk analisis *dataset* lengkap dengan pelaporan statistik.

2.5.7. Konfigurasi dan Validasi Eksperimen

Konfigurasi eksperimen dirancang untuk memvalidasi performa sistem dalam berbagai kondisi operasional dengan pendekatan *systematic testing*. *Environment* setup menggunakan pendekatan *containerized* melalui Google Colab *runtime environment* dengan instalasi dependensi otomatis menggunakan pip untuk *libraries* seperti *flask*, *librosa*, *transformers*, *torch*, *faster-whisper*, *noisereduce*, *pydub*, dan *jiwer*. Sistem *manajemen ground truth* mengimplementasikan format *filename|transcription* dengan encoding UTF-8 untuk menangani karakter bahasa Indonesia. Validasi *dataset* dilakukan dengan *automatic file discovery* untuk *multiple format audio* (WAV, MP3, FLAC, M4A) dan *cross-referencing* dengan *entry ground truth*. Metodologi evaluasi mencakup *multiple* skenario pengujian: *single file quick testing* untuk *feedback* langsung, *comparative analysis across different Whisper models* untuk *performance benchmarking*, validasi *optimization settings* untuk efektivitas *preprocessing*, dan *comprehensive evaluation* dengan *dataset* lengkap untuk signifikansi statistik. *Baseline measurements* diimplementasikan untuk membandingkan performa *hybrid cloud* dengan *theoretical single-node processing*, memberikan *insights* tentang *trade-offs* antara kompleksitas arsitektur terdistribusi dan *performance gains*. *Results persistence* menggunakan format JSON dengan metadata detail termasuk *processing timestamps*, konfigurasi model, spesifikasi *hardware*, dan metrik evaluasi untuk *reproducibility* dan analisis lebih lanjut.



Gambar 1. Arsitektur Sistem *Hybrid Cloud* ASR untuk Bahasa Indonesia

3. RESULT DAN ANALISIS

Bagian ini menyajikan hasil evaluasi dari sistem *Automatic Speech Recognition* (ASR) yang telah dikembangkan, mencakup aspek akurasi, efisiensi waktu pemrosesan, serta reliabilitas sistem. Hasil diperoleh melalui pengujian terhadap lima sampel audio berbahasa Indonesia dengan variasi konten dan ukuran *file*. Setiap hasil dianalisis secara kuantitatif dan dibahas berdasarkan karakteristik masing-masing data untuk memperoleh pemahaman mendalam mengenai kinerja sistem dalam skenario penggunaan nyata.

3.1. Hasil Evaluasi Sistem

Evaluasi sistem dilakukan menggunakan lima sampel audio berbahasa Indonesia yang merepresentasikan variasi jenis konten dan durasi. Hasil menunjukkan bahwa performa sistem sangat dipengaruhi oleh karakteristik masing-masing *file*, khususnya pada faktor panjang durasi dan kompleksitas suara. Dari lima sampel yang diuji, sistem berhasil memproses empat *file* secara lengkap, menghasilkan tingkat keberhasilan (*success rate*) sebesar 80%. Satu *file* berukuran besar mengalami kegagalan proses akibat *network timeout* pada saat transkripsi dijalankan melalui *server remote* berbasis Google Colab.

Secara umum, *file* dengan kualitas suara yang lebih terstruktur seperti berita televisi menunjukkan performa pengenalan kata yang lebih baik (WER 27,69%), sedangkan audio percakapan informal menghasilkan kesalahan sangat tinggi (WER 645,16%) karena variabilitas ucapan dan *noise*. *File* berukuran besar (>50MB) gagal diproses akibat keterbatasan stabilitas koneksi saat menggunakan infrastruktur *cloud*, menandakan pentingnya pengelolaan *file* besar dalam arsitektur *hybrid cloud* untuk penerapan di dunia nyata. Hasil ini sejalan dengan target penelitian yang dijelaskan dalam pendahuluan untuk mencapai WER di bawah 30% tanpa *fine-tuning*, yang berhasil dicapai pada audio berkualitas tinggi seperti berita formal.

Table 2. Hasil Evaluasi Sistem per *File* Audio

File Audio	WER (%)	CER (%)	*Time* (s)	RTF	Status

Briefing_1_UTBK_compress.mp3	54,12	32,41	84,64	0,21 x	☑
liputan_cnn_indonesia_tanah_longsor.mp3	27,69	22,73	25,51	0,20 x	☑
percakapan_id.mp3	645,16	666,34	25,05	0,23 x	⚠
rekomendasi_toko_oleh_oleh_dari_tanah_suci_kabar_haji_tvOne.mp3	62,71	45,03	24,92	0,12 x	☑
seleksi_mandiri_untidar.mp3	-	-	-	-	✗

Keterangan:

☑ = berhasil diproses
⚠ = diproses tetapi dengan akurasi sangat rendah
✗ = gagal diproses karena *network timeout*

3.2. Analisis Performa Sistem

Hasil evaluasi menunjukkan bahwa sistem memberikan performa yang dapat diterima (*acceptable*) untuk beberapa jenis konten audio, namun masih memerlukan optimasi lebih lanjut dalam menangani *file* berukuran besar. Rata-rata waktu pemrosesan tercatat sebesar 40,03 detik, dengan *Real-Time Factor* (RTF) rata-rata sebesar 0,14×. Nilai ini menunjukkan bahwa sistem belum mampu mencapai pemrosesan waktu nyata (*real-time*), namun masih berada dalam rentang yang layak untuk kebutuhan aplikasi *batch processing*.

File audio dengan konten berita formal, seperti liputan CNN Indonesia, menunjukkan performa terbaik dengan nilai WER sebesar 27,69%. Hasil ini memvalidasi efektivitas arsitektur *hybrid cloud* dan *multi-engine Whisper* yang diimplementasikan, sebagaimana dijelaskan dalam metodologi penelitian menggunakan *OptimizedWhisperEngine* dengan *Faster Whisper* large-v3 sebagai *engine* utama. Sebaliknya, file audio dengan percakapan informal menghasilkan tingkat kesalahan yang sangat tinggi (WER 645,16%), yang menunjukkan bahwa sistem mengalami kesulitan dalam mengenali pola tutur tidak formal dan percakapan yang saling tumpang tindih (*overlapping speech*). Hal ini menandakan perlunya pengembangan lebih lanjut untuk meningkatkan kemampuan sistem dalam menangani variasi ujaran yang lebih kompleks, sejalan dengan tantangan yang diidentifikasi dalam *gap analysis* pada bagian pendahuluan.

Implementasi *pipeline preprocessing audio* yang mencakup *noise reduction*, *audio normalization*, *silence trimming*, dan *smart chunking* menunjukkan kontribusi signifikan terhadap peningkatan akurasi, khususnya pada *file* dengan kualitas audio yang baik. *Multi-engine fallback strategy* yang diimplementasikan dalam *OptimizedWhisperEngine* berhasil mempertahankan fungsionalitas sistem meskipun menghadapi keterbatasan sumber daya, sesuai dengan desain arsitektur yang dijelaskan dalam bagian metodologi.

3.3. Tantangan dan Keterbatasan Sistem

Berdasarkan evaluasi, terdapat beberapa tantangan utama yang diidentifikasi selama pengujian. Pertama, terjadinya *network timeout* pada *file* berukuran besar (>50 MB), yang disebabkan oleh batas waktu permintaan (HTTP *request timeout*) pada koneksi internet. Kedua, degradasi akurasi yang signifikan pada audio dengan banyak pembicara (*multiple speakers*) dan pola tutur informal, yang menyulitkan proses segmentasi dan transkripsi. Ketiga, waktu pemrosesan yang belum optimal untuk aplikasi *real-time*, terutama pada *file* berdurasi panjang. Keempat, tingginya variabilitas performa antar jenis audio yang menunjukkan sensitivitas sistem terhadap karakteristik sinyal masukan.

Selain itu, munculnya SSL *connection error* saat memproses *file* besar menunjukkan bahwa lapisan jaringan perlu dioptimalkan, misalnya dengan implementasi strategi *chunking* yang lebih tangguh dan mekanisme pemulihan kesalahan (*error recovery*). Sensitivitas sistem terhadap kualitas

audio dan tingkat formalitas ujaran menunjukkan pentingnya peran *preprocessing* yang lebih adaptif dan standardisasi. Tantangan-tantangan ini mencerminkan kompleksitas implementasi ASR untuk bahasa Indonesia yang telah diidentifikasi dalam tinjauan literatur, khususnya terkait variasi dialek dan kompleksitas fonologis bahasa Indonesia.

Meskipun sistem berhasil mengatasi beberapa keterbatasan model *pre-trained* tanpa *fine-tuning* melalui optimasi arsitektur dan *preprocessing*, hasil menunjukkan bahwa tantangan dalam menangani audio percakapan informal masih memerlukan pendekatan yang lebih *sophisticated*. Hal ini sejalan dengan temuan penelitian Chen et al. yang menunjukkan kesulitan serupa dalam menangani *spontaneous conversations* bahasa Indonesia.

3.4. Analisis Pemanfaatan Sumber Daya

Monitoring selama proses inferensi menunjukkan bahwa sistem menggunakan sumber daya secara relatif efisien. Penggunaan GPU pada server Google Colab tercatat sebesar 3,2 GB dari total 15,0 GB VRAM yang tersedia, atau sekitar 21,3%. Sementara itu, pemanfaatan RAM sistem mencapai 4,5 GB dari kapasitas 12,7 GB (35,4%). Angka ini menunjukkan masih terdapat ruang optimasi, khususnya dalam mendukung pemrosesan serentak (*concurrent processing*) terhadap beberapa *file* audio sekaligus. Pada sisi *client* (Ubuntu Server), pemakaian sumber daya relatif minimal dengan ketersediaan RAM sebesar 22,6 GB dari total 32 GB, menunjukkan bahwa tidak terdapat hambatan signifikan pada sisi lokal. Hal ini mengindikasikan bahwa *bottleneck* utama berada pada proses transmisi jaringan dan inferensi berbasis GPU, bukan pada keterbatasan sumber daya di sisi *client*. Efisiensi penggunaan sumber daya ini memvalidasi desain arsitektur *hybrid cloud* yang bertujuan mengoptimalkan *cost-effectiveness* dan *scalability* sebagaimana dijelaskan dalam kontribusi penelitian.

Implementasi *resource management* pada GPU server menggunakan alokasi memori dinamis dengan presisi torch.float16 untuk CUDA dan float32 untuk CPU *processing* menunjukkan stabilitas yang baik tanpa indikasi *memory leaks* selama operasi kontinyu. Model *loading strategy* dengan pendekatan *lazy loading* dan *automatic model unloading* berhasil mempertahankan efisiensi memori, sesuai dengan optimasi yang diimplementasikan dalam *Optimized Whisper Engine*.

3.5. Pembahasan

Hasil evaluasi menunjukkan bahwa sistem berbasis arsitektur *hybrid cloud* berhasil diimplementasikan secara fungsional dan dapat menjalankan proses transkripsi secara terdistribusi. Meskipun demikian, masih terdapat sejumlah aspek yang memerlukan penguatan dan optimasi lanjutan agar sistem dapat beroperasi secara andal dalam berbagai skenario penggunaan. Salah satu indikasi utama dari kebutuhan ini terlihat pada variasi performa sistem yang cukup ekstrem, dengan nilai *Word Error Rate* (WER) berkisar antara 27,69% hingga 645,16%.

Rentang ini mengindikasikan bahwa akurasi sistem sangat dipengaruhi oleh kualitas sinyal masukan dan kompleksitas jenis ujaran. Performa terbaik dicapai saat sistem memproses *file* liputan_cnn_indonesia_tanah_longsor.mp3, yaitu rekaman berita formal dengan nilai WER sebesar 27,69%. Hasil ini menunjukkan bahwa sistem bekerja secara optimal dalam kondisi di mana kualitas audio tinggi, gangguan latar minimal, artikulasi jelas, dan pola tutur cenderung terstruktur. Hal ini sejalan dengan karakteristik rekaman berita profesional, di mana pembicara biasanya memiliki pelatihan vokal dan menggunakan bahasa baku yang lebih mudah dikenali oleh model ASR.

Pencapaian WER 27,69% pada audio berkualitas tinggi memenuhi target penelitian yang ditetapkan dalam pendahuluan untuk mencapai WER di bawah 30% tanpa melalui proses *fine-tuning*. Hal ini memvalidasi efektivitas pendekatan *hybrid cloud* dan integrasi *multi-engine Whisper* yang menjadi kontribusi utama penelitian ini. Implementasi *preprocessing pipeline* yang mencakup *noise reduction*, *normalization*, *silence trimming*, dan *smart chunking* terbukti memberikan kontribusi signifikan terhadap peningkatan akurasi sistem.

Sebaliknya, hasil terburuk diperoleh pada file percakapan_id.mp3, yang merupakan percakapan informal dengan WER sebesar 645,16%. Hal ini mencerminkan tantangan umum yang dihadapi oleh sistem ASR saat dihadapkan pada tuturan alami yang mengandung *overlapping speech*, kosakata nonstandar, intonasi bervariasi, serta jeda atau interupsi yang tidak terprediksi. Kompleksitas ini secara langsung memengaruhi ketepatan segmentasi dan inferensi model, sehingga menghasilkan tingkat

kesalahan yang sangat tinggi. Temuan ini memperkuat asumsi bahwa performa sistem ASR sangat kontekstual, dan sangat dipengaruhi oleh jenis ujaran serta kualitas rekaman yang digunakan.

Tantangan lainnya yang muncul adalah kegagalan pemrosesan *file* seleksi_mandiri_untidar.mp3 berukuran 55 MB, yang disebabkan oleh *network timeout* dan SSL *connection errors*. Masalah ini menunjukkan bahwa komunikasi antara komponen lokal dan *server cloud* melalui *tunnel ngrok* belum sepenuhnya stabil untuk *file* berdurasi panjang. Hal ini menegaskan perlunya pengembangan strategi pemrosesan berbasis *streaming* atau implementasi mekanisme *chunking* yang lebih tangguh, lengkap dengan penanganan galat otomatis (*robust error handling*) dan dukungan *reconnection*.

Dari sisi efisiensi, sistem mencatat *Real-Time Factor* (RTF) rata-rata sebesar 0,14×. Meskipun belum mencapai standar waktu nyata (*real-time*), nilai ini masih dapat diterima untuk kebutuhan pemrosesan secara *batch*, seperti transkripsi arsip audio atau dokumen. Konsistensi waktu pemrosesan untuk *file* berdurasi normal yang berada di kisaran 24–25 detik mencerminkan kestabilan performa sistem, yang menjadi nilai tambah dalam konteks perencanaan produksi (*production planning*).

Pemantauan pemanfaatan sumber daya selama eksperimen juga menunjukkan efisiensi yang baik. GPU server hanya menggunakan 21,3% dari total kapasitas VRAM (3,2 GB dari 15,0 GB), sedangkan penggunaan RAM sistem tercatat 35,4%. Angka ini menunjukkan adanya ruang untuk optimasi, misalnya dengan pendekatan *concurrent processing* untuk meningkatkan *throughput system*. Di sisi klien, Ubuntu Server menunjukkan konsumsi sumber daya yang rendah dengan 22,6 GB RAM tersedia dari total 32 GB, yang mengindikasikan bahwa beban komputasi telah berhasil dialihkan secara efektif ke sisi *cloud*, sesuai dengan prinsip desain *hybrid cloud*.

Implementasi *multi-engine fallback strategy* yang mencakup *Faster Whisper*, *OpenAI Whisper*, dan *Transformers Whisper* menunjukkan keberhasilan dalam mempertahankan fungsionalitas sistem meskipun menghadapi keterbatasan sumber daya. *Dynamic model switching* yang diimplementasikan dalam *OptimizedWhisperEngine* memungkinkan sistem beradaptasi terhadap kondisi operasional yang berubah, sesuai dengan desain arsitektur yang fleksibel dan *scalable*.

Hasil penelitian ini juga memvalidasi efektivitas pendekatan *privacy-by-design* dan data *minimization* yang diimplementasikan melalui ngrok *tunneling* dan *automatic data cleanup*. Keamanan transmisi data melalui HTTPS dan base64 *encoding* terbukti dapat menjaga integritas data audio selama proses komunikasi antar komponen sistem.

Secara keseluruhan, hasil penelitian ini membuktikan kelayakan (*feasibility*) implementasi arsitektur *hybrid cloud* untuk pemrosesan ASR terdistribusi pada bahasa Indonesia. Pencapaian WER 27,69% pada audio berkualitas tinggi tanpa *fine-tuning* menunjukkan potensi signifikan pendekatan ini untuk aplikasi praktis. Meski demikian, dibutuhkan pengembangan lanjutan, khususnya dalam aspek penanganan *file* besar, akurasi pada percakapan informal, dan stabilitas jaringan. Optimasi pada *preprocessing*, strategi *chunking*, dan manajemen sumber daya dipandang krusial untuk mendorong sistem menuju kinerja yang lebih stabil dan andal dalam penerapan produksi nyata.

Kontribusi penelitian yang mencakup desain arsitektur *hybrid cloud*, *integrasi multi-engine Whisper*, dan *pipeline preprocessing* yang adaptif telah terbukti memberikan *foundation* yang *solid* untuk pengembangan ASR bahasa Indonesia yang *cost-effective* dan *scalable*. Hasil ini sejalan dengan potensi aplikasi yang luas dalam bidang pendidikan, kesehatan, pemerintahan, dan industri kreatif sebagaimana diidentifikasi dalam bagian pendahuluan.

4 DISCUSSION/CONCLUSION

Penelitian ini telah berhasil merancang dan mengimplementasikan sistem *Automatic Speech Recognition* (ASR) berbasis arsitektur *hybrid cloud* dengan mengintegrasikan model *Faster Whisper* large-v3 untuk pemrosesan bahasa Indonesia. Sistem menunjukkan kemampuan fungsional yang memadai, ditunjukkan oleh tingkat keberhasilan pemrosesan sebesar 80% dari lima sampel audio dengan variasi konten. Hasil evaluasi mengindikasikan bahwa kinerja sistem sangat bergantung pada kualitas dan karakteristik ujaran, dengan nilai *Word Error Rate* (WER) yang bervariasi antara 27,69% hingga 645,16%. Audio dengan struktur formal seperti berita menunjukkan akurasi yang tinggi, sementara percakapan informal menghasilkan tingkat kesalahan yang signifikan, menandakan perlunya optimasi lebih lanjut pada konteks *natural conversation*. Dari sisi efisiensi, sistem menunjukkan

pemanfaatan sumber daya yang relatif rendah, dengan penggunaan GPU sebesar 21,3% dan RAM sebesar 35,4%. Waktu pemrosesan yang konsisten pada *file* berdurasi normal menunjukkan stabilitas sistem dalam skenario pemrosesan *batch*. Namun demikian, sistem masih menghadapi tantangan dalam menangani *file* audio berukuran besar (>50 MB), yang menunjukkan perlunya pengembangan strategi *streaming* dan penanganan galat (*error handling*) yang lebih tangguh. Kontribusi utama dari penelitian ini mencakup keberhasilan penerapan arsitektur *hybrid cloud* untuk pemrosesan ASR terdistribusi, penyusunan kerangka evaluasi komprehensif untuk audio bahasa Indonesia, identifikasi pola performa berdasarkan jenis konten, serta analisis efisiensi sumber daya yang dapat dijadikan referensi dalam perencanaan implementasi sistem produksi. Ke depan, arah penelitian dapat difokuskan pada pengembangan protokol *streaming* untuk *file* berdurasi panjang, penyempurnaan proses *preprocessing* khusus untuk percakapan informal, integrasi *multi-engine Whisper* untuk pendekatan *ensemble*, serta penerapan mekanisme adaptif yang mampu menyesuaikan strategi pemrosesan berdasarkan karakteristik audio secara otomatis. Pengembangan ini diharapkan dapat meningkatkan akurasi, skalabilitas, dan fleksibilitas sistem dalam berbagai skenario penggunaan nyata.

ACKNOWLEDGEMENTS

Penulis mengucapkan terima kasih kepada Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Tidar yang telah menyediakan fasilitas penelitian, dukungan teknis, dan infrastruktur komputasi yang memungkinkan terlaksananya penelitian ini. Ucapan terima kasih juga disampaikan kepada seluruh civitas akademika yang telah memberikan masukan dan saran konstruktif dalam pengembangan sistem ini.

REFERENCES

Adiwidjaja, R., & Ivan Fanany, M. (2020). End-to-end indonesian speech recognition with convolutional and gated recurrent units. *Journal of Physics: Conference Series*, *1566*(1), 12118. https://doi.org/10.1088/1742-6596/1566/1/012118

Anjani, H. U., Vitriani, V., & Hastuti, M. (2024). Pemanfaatan Media Google Colaboratory Pada Mata Pelajaran Informatika di SMA Negeri 5 Pekanbaru. *SOKO GURU: Jurnal Ilmu Pendidikan*, *4*(1), 101–108. https://doi.org/10.55606/sokoguru.v4i1.3613

Arafah, M. M., Jaya, A. K., Suryawan, Z. G. M. A., Banjarnahor, A. R., Bukidz, D. P., Simanjuntak, H. M., Saputra, N., & Fajrillah, F. (2023). Implementasi Artificial Intelligence (AI) Dalam Kehidupan. In *Yayasan Kita Menulis*. http://repository.upy.ac.id/4945/1/FullBook Implementasi Artificial Intelligence (AI) dalam Kehidupan.pdf

Arianto, O. D., & Susetyo, Y. A. (2022). Penerapan Restful Web Service Dengan Framework Laravel Untuk Pembangunan Sistem Informasi Manajemen Sumber Daya Manusia. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *7*(2), 522–532. https://doi.org/10.29100/jipi.v7i2.2870

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *Advances in Neural Information Processing Systems*, *2020-Decem*(Figure 1), 1–19.

Butarbutar, M., Sachio, K., Nugroho, M., David, D., & Saputra, P. (2023). *Adaptive Wiener Filtering Method for Noise Reduction in Speech Recognition System*. https://doi.org/10.36227/techrxiv.23608602.v1

Cahyawijaya, S., Lovenia, H., Aji, A. F., Winata, G., Wilie, B., Koto, F., Mahendra, R., Wibisono, C., Romadhony, A., Vincentio, K., Santoso, J., Moeljadi, D., Wirawan, C., Hudi, F., Wicaksono, M. S., Parmonangan, I., Alfina, I., Putra, I. F., Rahmadani, S., … Purwarianti, A. (2023). NusaCrowd: Open Source Initiative for Indonesian NLP Resources. In A. Rogers, J. Boyd-Graber, & N. Okazaki (Eds.), *Findings of the Association for Computational Linguistics: ACL 2023* (pp. 13745–13818). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-acl.868

Chairani, M. S. (2023). *TELEMEDICINE SEBAGAI BENTUK DIGITALISASI PELAYANAN KESEHATAN DI INDONESIA: TINJAUAN LITERATUR*.

Fleck, M., & Goderle, W. (2023). wav2vec and its Current Potential to Automatic Speech Recognition in German for the usage in Digital History: A comparative assessment of available ASR-technologies for the use in cultural heritage contexts. *Sustainability (Switzerland)*, *11*(1), 1–14. https://doi.org/https://doi.org/10.48550/arXiv.2303.06026

Herdiana, B., Setiawan, E. B., & Sartoyo, U. (2023). Tinjauan Komprehensif Evolusi, Aplikasi, dan Tren Masa

Depan Programmable Logic Controllers. *Telekontran: Jurnal Ilmiah Telekomunikasi, Kendali Dan Elektronika Terapan*, *11*(2), 173–193. https://ojs.unikom.ac.id/index.php/telekontran/article/view/12896%0Ahttps://ojs.unikom.ac.id/index.php/telekontran/article/download/12896/4459

Katti, A., & Sumana, M. (2023). Pipeline for Pre-processing of Audio Data. In J. Choudrie, P. Mahalle, T. Perumal, & A. Joshi (Eds.), *IOT with Smart Systems* (pp. 191–198). Springer Nature Singapore.

Khoiroh, R. F., Julianto, E., Adiyansa, S. A., Fajri, H. A., Abi, A., Yasa, R., & Sangapta, B. (2024). Implementasi Speech Recognition Whisper Pada Debat Calon Wakil Presiden Republik Indonesia. *EXPLORE*, *14*(2), 67–74.

Manu, G. A., & Masan, P. L. (2020). Aplikasi Text To Speech Untuk Meningkatkan Pembelajaran Bahasa Inggris Bagi Siswa Disabilitas. *Jurnal Pendidikan Teknologi Informasi (JUKANTI)*, *3*(2), 17–26. https://doi.org/10.37792/jukanti.v3i2.217

Özkurt, C. (2024). *Investigation of the Effectiveness of Audio Processing and Filtering Strategies in Noisy Environments on Speech Recognition Performance*. 0–23. https://doi.org/10.21203/rs.3.rs-3973856/v1

Raharjo, B. (2022). *Deep Learning dengan Python* (M. C. Wibowo (ed.)). Yayasan Prima Agus Teknik.

Soen, G. I. E., Marlina, M., & Renny, R. (2022). Implementasi Cloud Computing dengan Google Colaboratory pada Aplikasi Pengolah Data Zoom Participants. *JITU : Journal Informatic Technology And Communication*, *6*(1), 24–30. https://doi.org/10.36596/jitu.v6i1.781

Sudhakaran, P., Kumar Yadav, A., & Karamchandani, S. (2024). an End-To-End Deep Learning Approach for an Indian English Repository. *Journal of Theoretical and Applied Information Technology*, *102*(3), 1216–1226.

Tofure, I. R., Erwada, B. A. De, & Ukratalo, A. M. (2025). Telemedicine Sebagai Media Konsultasi Layanan Kesehatan Bagi Masyarakat di Wilayah Pesisir. *Anestesi*, *3*(1), 121–134.

Wafiy, A. D., & Prasetio, B. H. (2022). Penerapan Model Whisper Pada Embedded System Untuk Speech to Text. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *1*(1), 1–7.

Wahidin, M. (2021). PERENCANAAN ARSITEKTUR ENTERPRISE BERBASIS CLOUD COMPUTING MENGGUNAKAN TOGAF (Studi Kasus: PT. XYZ). *Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi Dan Komunikasi*, *13*(1), 28–35. https://doi.org/10.35969/interkom.v13i1.42

Wibowo, T. S., Mamis, S., Yahya, S. R., Romadloni, N. T., Witjaksono, G., Trianti, F. A., Nurislamiah, M., Fauzi, R., Silviana, S. C., Fadilah, R., Tantrisna, E., Pardosi, V. B. A., & N, M. A. (2024). Transformasi Teknologi Komunikasi. In *Aina Media Baswara*. http://scioteca.caf.com/bitstream/handle/123456789/1091/RED2017-Eng-8ene.pdf?sequence=12&isAllowed=y%0Ahttp://dx.doi.org/10.1016/j.regsciurbeco.2008.06.005%0Ahttps://www.researchgate.net/publication/305320484_SISTEM_PEMBETUNGAN_TERPUSAT_STRATEGI_MELESTARI

William, E., & Zahra, A. (2025). SPEECH RECOGNITION DENGAN WHISPER DALAM BAHASA INDONESIA. *Action Research Literate*, *9*(2), 386–397.

Zebua, R. S. Y., Khairunnisa, K., Hartatik, H., Pariyadi, P., Wahyuningtyas, D. P., Thantawi, A. M., Sudipa, I. G. I., Prayitno, H., Sumakul, G. C., Sepriano, S., & Kharisma, L. P. I. (2023). *FENOMENA ARTIFICIAL INTELLIGENCE (AI)* (Efitra (ed.); 1st ed., Issue June). PT. Sonpedia Publishing Indonesia. https://www.researchgate.net/publication/371491224

Zhang, Y., Han, W., Qin, J., Wang, Y., Bapna, A., Chen, Z., Chen, N., Li, B., Axelrod, V., Wang, G., Meng, Z., Hu, K., Rosenberg, A., Prabhavalkar, R., Park, D. S., Haghani, P., Riesa, J., Perng, G., Soltau, H., … Wu, Y. (2023). *Google USM: Scaling Automatic Speech Recognition Beyond 100 Languages*. http://arxiv.org/abs/2303.01037