

BAB II

LANDASAN TEORI

2.1 Kajian Pustaka

Kajian pustaka merupakan salah satu bagian penting dalam penelitian yang berfungsi sebagai landasan teoritis untuk mendukung penyelesaian permasalahan penelitian. Melalui kajian pustaka, peneliti memperoleh pemahaman mengenai perkembangan ilmu pengetahuan, konsep, metode, serta berbagai penelitian terdahulu yang berkaitan dengan topik penelitian sehingga dapat menentukan posisi penelitian sekaligus menemukan *research gap* yang masih dapat dikembangkan. Menurut Muhammad Zahrul Jihadurrohim Bajuri dkk. (2024), kajian pustaka tidak sekadar berfungsi sebagai kumpulan referensi, melainkan juga menjadi dasar dalam menyusun *state of the art*, menemukan kesenjangan penelitian, dan menunjukkan nilai kebaruan (*novelty*) dari penelitian yang dilakukan.

Menurut Dian Kurniati (2023), kajian pustaka dilakukan melalui telaah terhadap berbagai sumber ilmiah, seperti jurnal, prosiding, buku, serta publikasi ilmiah lain yang sesuai dengan topik penelitian sehingga peneliti dapat memperoleh pemahaman mengenai perkembangan penelitian pada bidang yang dikaji. Kajian pustaka juga membantu peneliti dalam menentukan metode penelitian, memilih pendekatan yang sesuai, serta menghindari terjadinya duplikasi penelitian yang telah dilakukan sebelumnya sebelumnya.

Dalam penelitian ini, kajian pustaka difokuskan pada pembahasan mengenai teori dan penelitian yang berkaitan dengan *Artificial Intelligence* (AI), *Natural Language Processing* (NLP), *Machine Translation*, *AI Translation Engine*, *Hybrid AI Translation*, *Google Gemini* sebagai *Large Language Model* (LLM), *Flutter Framework*, *Retrieval-Based Voice Conversion* REST API, *Edge Text-to-Speech* (Edge TTS), *Retrieval-Based Voice Conversion* (RVC), *Mean Opinion Score* (MOS), Black Box Testing, serta metode pengembangan perangkat lunak Prototyping. Kajian terhadap teori-teori tersebut digunakan sebagai dasar dalam merancang dan mengembangkan aplikasi LingoNusantara AI, yaitu aplikasi *mobile* penerjemah Bahasa Indonesia ke Bahasa Sunda yang mengintegrasikan *AI Translation Engine* berbasis dataset lokal dengan *Google Gemini* melalui mekanisme *Hybrid AI Translation*, serta teknologi *Retrieval-Based Voice Conversion* (RVC) untuk menghasilkan terjemahan yang lebih kontekstual dan keluaran suara yang tetap mempertahankan karakteristik vokal pengguna.

Selain sebagai landasan teoritis, kajian pustaka pada penelitian ini juga digunakan untuk mengidentifikasi perkembangan penelitian terkini yang berkaitan dengan penerjemahan bahasa daerah berbasis *Artificial Intelligence*, pengolahan bahasa alami, teknologi *Large Language Model* (LLM), *Machine Translation*, serta teknologi konversi suara. Hasil kajian tersebut selanjutnya digunakan untuk menyusun *State of the Art*, menentukan *Research Gap*, dan mempertegas *Novelty* penelitian sehingga kontribusi ilmiah yang dihasilkan dapat dibedakan dengan penelitian-penelitian sebelumnya. Menurut Muhammad Zahrul Jihadurrohim Bajuri dkk. (2024), penyusunan *State of the Art* harus mampu menggambarkan perkembangan penelitian terkini, menunjukkan posisi penelitian yang sedang dilakukan, serta menjelaskan kontribusi baru yang ditawarkan oleh penelitian tersebut.

2.1.1. *State of the Art*

State of the Art merupakan kajian terhadap penelitian-penelitian terdahulu yang bertujuan untuk mengetahui perkembangan penelitian, mengidentifikasi kelebihan dan kekurangan penelitian sebelumnya, serta menentukan posisi penelitian yang dilakukan. Melalui *State of the Art*, peneliti dapat menunjukkan nilai kebaruan (*novelty*) dari penelitian sehingga kontribusi ilmiah yang diberikan menjadi lebih jelas. Dalam penelitian ini, kajian dilakukan terhadap berbagai penelitian yang berkaitan dengan *Artificial Intelligence* (AI), *Natural Language Processing* (NLP), *Machine Translation*, *Large Language Model* (LLM), *Flutter*, *Retrieval-Based Voice Conversion* REST API, *Speech Processing*, serta *Retrieval-Based Voice Conversion* (RVC).

Menurut Novitasari, Tjandra, Sakti, dan Nakamura (2020), pengembangan sistem *Cross-Lingual Machine Speech Chain* untuk bahasa daerah di Indonesia, termasuk Bahasa Sunda, menunjukkan bahwa teknologi *Automatic Speech Recognition* (ASR) dan *Text-to-Speech* (TTS) dapat dikembangkan pada bahasa dengan sumber daya terbatas (*low-resource languages*). Penelitian tersebut berhasil meningkatkan kualitas pengolahan suara bahasa daerah, namun belum mengintegrasikan proses penerjemahan bahasa maupun personalisasi identitas suara pengguna.

Selanjutnya, Hsu et al. (2021) memperkenalkan *model* HuBERT (*Hidden Unit BERT*) sebagai metode *self-supervised learning* untuk representasi suara. Penelitian tersebut menunjukkan bahwa HuBERT mampu menghasilkan representasi akustik yang lebih baik dibandingkan pendekatan sebelumnya sehingga banyak digunakan sebagai *feature extractor* pada berbagai penelitian *Voice Conversion*, termasuk teknologi *Retrieval-Based Voice*

Conversion (RVC). Namun demikian, penelitian tersebut hanya berfokus pada representasi suara dan belum diterapkan pada sistem penerjemahan bahasa daerah.

Menurut Shi et al. (2022) dalam survei mengenai *Low-Resource Neural Machine Translation*, pengembangan sistem penerjemahan untuk bahasa dengan sumber daya terbatas masih menghadapi berbagai tantangan, antara lain keterbatasan dataset paralel, rendahnya kualitas korpus, dan kompleksitas struktur bahasa. Penelitian tersebut menegaskan bahwa peningkatan kualitas dataset merupakan salah satu faktor utama dalam meningkatkan akurasi sistem penerjemahan berbasis *Artificial Intelligence*.

Menurut Amien (2023), perkembangan *Natural Language Processing* (NLP) di Indonesia menunjukkan peningkatan yang cukup pesat dengan pemanfaatan berbagai metode pembelajaran mesin dan *Deep Learning*. Penelitian tersebut menjelaskan bahwa penerapan NLP pada bahasa Indonesia telah berkembang pada berbagai bidang, seperti analisis sentimen, klasifikasi teks, dan penerjemahan mesin. Akan tetapi, penelitian mengenai NLP untuk bahasa daerah, khususnya Bahasa Sunda, masih relatif terbatas sehingga diperlukan penelitian lebih lanjut untuk meningkatkan kualitas sistem penerjemahan bahasa daerah.

Menurut Yuyun et al. (2024), peningkatan kualitas *Neural Machine Translation* pada bahasa daerah dengan sumber daya terbatas dapat dilakukan melalui optimalisasi dataset, proses *fine-tuning*, serta pemanfaatan *model Deep Learning*. Hasil penelitian tersebut menunjukkan bahwa kualitas dataset memiliki pengaruh yang sangat besar terhadap performa sistem penerjemahan sehingga proses validasi dan normalisasi data menjadi tahapan yang penting sebelum *model* digunakan.

Menurut Purnomo dan Sutopo (2024), *model* berbasis BERT yang dilatih menggunakan dataset bahasa daerah Indonesia mampu memberikan performa yang lebih baik dibandingkan *model multilingual* umum. Penelitian tersebut menunjukkan bahwa *model* yang dilatih menggunakan karakteristik bahasa lokal memiliki kemampuan yang lebih baik dalam memahami konteks bahasa dibandingkan *model* yang hanya dilatih menggunakan data multibahasa.

Haddow, B., Bawden, Miceli Barone, Helcl, dan Birch (2022) menjelaskan bahwa pengembangan *Machine Translation* pada bahasa dengan sumber daya rendah (*low-resource languages*) masih menghadapi berbagai tantangan, seperti keterbatasan korpus paralel, rendahnya kualitas dataset, serta kompleksitas linguistik antarbahasa. Oleh karena itu, diperlukan pendekatan yang mampu memanfaatkan sumber daya bahasa secara lebih efektif agar kualitas hasil penerjemahan dapat meningkat. Penelitian tersebut menjadi salah satu landasan dalam pengembangan *Hybrid AI Translation* pada penelitian ini karena menekankan

pentingnya kombinasi *model* penerjemahan dan kualitas data dalam menghasilkan terjemahan yang lebih akurat.

Berdasarkan berbagai penelitian tersebut dapat disimpulkan bahwa sebagian besar penelitian masih berfokus pada salah satu aspek, seperti *Machine Translation*, *Natural Language Processing*, *Large Language Model* (LLM), maupun *Voice Conversion* secara terpisah. Belum banyak penelitian yang mengintegrasikan *AI Translation Engine* berbasis dataset lokal, *Hybrid AI Translation* menggunakan *Google Gemini*, *Flutter Mobile Application*, *Retrieval-Based Voice Conversion* REST API, *Edge Text-to-Speech* (Edge TTS), serta *Retrieval-Based Voice Conversion* (RVC) ke dalam satu aplikasi penerjemah Bahasa Indonesia ke Bahasa Sunda. Oleh karena itu, penelitian ini mengusulkan pengembangan aplikasi LingoNusantara AI yang menggabungkan *Dictionary Engine*, *Phrase Engine*, *Grammar Engine*, *Sentence Engine*, *Context Engine*, dan *Idiom Engine* sebagai *AI Translation Engine* berbasis dataset lokal, kemudian mengintegrasikannya dengan *Google Gemini* melalui mekanisme *Hybrid AI Translation* untuk menghasilkan terjemahan yang lebih kontekstual ketika perangkat terhubung ke internet. Hasil terjemahan selanjutnya diproses menggunakan *Edge Text-to-Speech* (Edge TTS) dan *Retrieval-Based Voice Conversion* (RVC) melalui *platform Applio* sehingga menghasilkan keluaran suara yang tetap mempertahankan karakteristik vokal pengguna.

Tabel 2. 1 Perbandingan Penelitian Sebelumnya (State of the Art)

No	Penulis	Tahun	Judul Penelitian	Jurnal / Prosiding	Vol.	No.	Hal.	DOI / ISBN / URL	Metode	Objek Penelitian	Perbandingan dengan Penelitian
1	Novita sari, S., Tjandra, A., Sakti, S., & Nakamura, S.	2020	<i>Cross-Lingual Machine Speech Chain for Javanese, Sundanese, Balinese, and</i>	Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-	—	—	131–138	ISBN: 979-10-95546-35-1 DOI: 10.48550/arXiv.201	<i>Cross-Lingual Machine Speech Chain</i>	<i>Speech Recognition dan Speech Synthesis</i> bahasa daerah Indonesia	Mengembangkan <i>Automatic Speech Recognition</i> (ASR) dan <i>Text-to-Speech</i> (TTS) untuk beberapa bahasa daerah, termasuk

			<i>Bataks Speech Recognition and Synthesis</i>	resourced Languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)				1.02128			Bahasa Sunda. Namun penelitian tersebut belum mengintegrasikan penerjemahan Bahasa Indonesia–Bahasa Sunda maupun <i>Retrieval-Based Voice Conversion</i> (RVC). Penelitian ini mengintegrasikan <i>AI Translation Engine, Hybrid AI Translation</i> , dan RVC dalam satu aplikasi.
2	Wibowo, H. A., Prawiro, T. A., Ihsan, M., Aji, A. F., Prasoj, R.	2020	<i>Semi-Supervised Low-Resource Style Transfer of Indonesian Informal to Formal Language</i>	2020 International Conference on Asian Language Processing (IALP)	–	–	310–315	DOI: 10.1109/IALP5139.2020.9310459	<i>Semi-Supervised Learning</i>	Transformasi Bahasa Indonesia Informal–Formal	Berfokus pada transformasi gaya bahasa Indonesia informal menjadi formal menggunakan pendekatan <i>Machine Translation</i> . Penelitian tersebut belum

	E., Mahendra, R., & Fitriany, S.		<i>with Iterative Forward-Translation</i>								membahas bahasa daerah maupun teknologi <i>Voice Conversion</i> . Penelitian ini mengembangkan sistem penerjemahan Bahasa Indonesia–Bahasa Sunda dengan integrasi RVC.
3	Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., & Mohamed, A.	2021	<i>HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units</i>	IEEE/ACM Transactions on Audio, Speech, and Language Processing	29	–	3451–3460	DOI: 10.1109/TASLP.2021.3122291	<i>Self-Supervised Learning</i>	Representasi fitur suara	Memperkenalkan model HuBERT sebagai metode ekstraksi fitur suara berbasis <i>Self-Supervised Learning</i> . Penelitian tersebut belum mengembangkan sistem penerjemahan bahasa. Penelitian ini memanfaatkan konsep HuBERT sebagai dasar

											implementasi <i>Retrieval-Based Voice Conversion</i> (RVC).
4	Budaya, Kesiman, & Sunarya	2022	<i>Perancangan Mesin Translasi Berbasis Neural dari Bahasa Kawi ke dalam Bahasa Indonesia menggunakan Microframework Flask</i>	Jurnal Sistem dan Informatika (JSI)	16	2	94–103	DOI: 10.30864/jsi.v16i2.440	<i>Neural Machine Translation</i>	Mesin Translasi Bahasa Kawi–Bahasa Indonesia	Mengembangkan mesin translasi berbasis <i>Neural Machine Translation</i> menggunakan <i>Flask</i> . Penelitian tersebut belum mengembangkan aplikasi <i>mobile</i> , belum menerapkan <i>Hybrid AI Translation</i> , dan belum mengintegrasikan teknologi <i>Retrieval-Based Voice Conversion</i> (RVC).
5	Cahya wijaya, S., Winata, G. I., Wilie, B.,	2021	<i>IndoNLG: Benchmark and Resources for Evaluating</i>	Proceedings of the 2021 Conference on Empirical Methods	–	–	8875–8898	DOI: 10.18653/v1/2021.emnlp-main.699	<i>Natural Language Generation</i>	<i>Natural Language Processing</i> Bahasa Indonesia	Menyediakan <i>benchmark</i> dan sumber daya untuk pengembangan <i>Natural Language</i>

	Vincen- tio, K., Li, X., Kunco- ro, A., Ruder, S., Lim, Z. Y., Bahar, S., Khodr- a, M. L., Purwa- rianti, A., & Fung, P.		<i>Indonesian Natural Language Generation</i>	in Natural Language Processing (EMNLP)							<i>Processing Bahasa Indonesia. Penelitian tersebut tidak mengembangkan sistem penerjemahan Bahasa Indonesia– Bahasa Sunda maupun integrasi Retrieval-Based Voice Conversion (RVC).</i>
6	Susanto, Y., Diandaru, R., Krisnadh- i, A., Purwa- rianti, A., & Wijaya, D.	2023	<i>Replicable Benchmarking of Neural Machine Translation on Low-Resource Local Languages in Indonesia</i>	Proceedings of the 13th Workshop on South and Southeast Asian Natural Language Processing (SEALP 2023)	–	–	60–69	DOI: 10.18653/v1/2023.sealp-1.8	<i>Neural Machine Translation</i>	Bahasa daerah Indonesia	Menyajikan <i>benchmark</i> dan evaluasi <i>Neural Machine Translation</i> untuk bahasa daerah Indonesia. Penelitian tersebut belum mengembangkan aplikasi <i>mobile</i> , belum menerapkan <i>Hybrid AI</i>

											Translation, serta belum mengintegrasikan <i>Retrieval-Based Voice Conversion</i> (RVC).
7	Permatasari, S. S., Santosa, S. Z. D., & Santosa, S. Z. T.	2024	<i>Performance of Machine Translation for Indonesian Students' Perception: A Systematic Literature Review of DeepL Machine Translation</i>	Proceedings of Academic Seminar and International Conference (ASIC)	1	1	13–18	DOI: 10.53893/asic.v1i1.288	<i>Systematic Literature Review</i>	Evaluasi <i>Machine Translation</i>	Mengevaluasi performa <i>Machine Translation</i> berdasarkan persepsi pengguna terhadap DeepL. Penelitian tersebut tidak mengembangkan sistem penerjemahan maupun teknologi <i>Voice Conversion</i> . Penelitian ini mengembangkan sistem penerjemahan berbasis AI yang terintegrasi dengan <i>Hybrid AI Translation</i> dan RVC.

8	Mustofa, M., Mustofa, A., & Surur, M.	2025	<i>Variations of Arabic-Indonesian Translation Strategies in Automatic Machine Translation</i>	Journal of Language and Linguistic Studies	5	3	522–540	DOI: 10.36312/jolls.v5i3.3075	<i>Comparative Study</i>	<i>Automatic Machine Translation Arab-Indonesia</i>	Membahas strategi penerjemahan Arab-Indonesia. Penelitian tersebut belum mendukung bahasa daerah Indonesia, belum mengembangkan aplikasi <i>mobile</i> , dan belum mengintegrasikan <i>Retrieval-Based Voice Conversion</i> (RVC).
9	Novitasari, S., Tjandra, A., Sakti, S., & Nakamura, S. (implementasi <i>Speech Chain</i> sebagai	2020	<i>Cross-Lingual Machine Speech Chain</i>	arXiv	–	–	–	DOI: 10.48550/arXiv.2011.02128	<i>Speech Chain</i>	Bahasa daerah Indonesia	Menunjukkan pendekatan lintas bahasa daerah pada pengolahan suara, tetapi belum mengintegrasikan <i>Machine Translation</i> , <i>Hybrid AI Translation</i> , maupun <i>Retrieval-Based Voice Conversion</i>

	<i>i refere nsi lintas bahas a daerah)</i>										dalam satu aplikasi <i>mobile</i> .
10	Muhamad Aksay Kumar	2026	<i>Rancang Bangun Sistem Penerjemahan Bahasa Sunda Berbasis Artificial Intelligence (AI) dan Integrasi Retrieval-Based Voice Conversion (RVC) Menggunakan Metode Prototype</i>	Penelitian Tugas Akhir, Program Studi Teknik Informatika, Universitas Islam Nusantara	–	–	–	Belum dipublikasikan	<i>Prototype, Artificial Intelligence, Natural Language Processing, Hybrid AI Translation, dan Retrieval-Based Voice Conversion</i>	Aplikasi <i>mobile</i> penerjemahan Bahasa Indonesia–Bahasa Sunda	Mengintegrasikan <i>AI Translation Engine</i> berbasis <i>dataset</i> lokal, Google Gemini melalui <i>Hybrid AI Translation, Edge Text-to-Speech</i> , dan <i>Retrieval-Based Voice Conversion</i> (RVC) dalam satu aplikasi <i>mobile</i> berbasis Flutter sehingga mampu melakukan penerjemahan secara <i>offline</i> maupun <i>online</i> serta menghasilkan

											suara yang mempertahankan karakteristik vokal pengguna.
--	--	--	--	--	--	--	--	--	--	--	---

2.2 Artificial Intelligence (AI)

Artificial Intelligence (AI) atau kecerdasan buatan merupakan cabang ilmu komputer yang bertujuan untuk mengembangkan sistem yang mampu meniru kemampuan berpikir manusia, seperti belajar dari data, melakukan penalaran, mengambil keputusan, serta menyelesaikan permasalahan secara otomatis. Menurut Russell dan Norvig (2021), *Artificial Intelligence* merupakan studi mengenai agen cerdas (*intelligent agents*), yaitu sistem yang mampu mengamati lingkungan, memproses informasi, dan melakukan tindakan yang dapat meningkatkan peluang keberhasilan dalam mencapai tujuan tertentu. Perkembangan AI saat ini telah mendorong berbagai inovasi pada bidang pengolahan bahasa alami, pengenalan suara, visi komputer, sistem rekomendasi, hingga kendaraan otonom.

Menurut Chowdhary (2020), perkembangan *Artificial Intelligence* mengalami peningkatan yang sangat pesat seiring berkembangnya teknologi *Machine Learning* dan *Deep Learning*. Kedua pendekatan tersebut memungkinkan komputer mempelajari pola secara otomatis dari sejumlah besar data (*big data*) tanpa harus diprogram menggunakan aturan-aturan yang kaku. Pendekatan ini menjadikan *Artificial Intelligence* mampu memberikan hasil yang lebih akurat pada berbagai permasalahan kompleks, termasuk penerjemahan bahasa, pengenalan suara, sintesis suara, serta pemrosesan bahasa alami.

Menurut Jurafsky dan Martin (2023), *Artificial Intelligence* pada bidang bahasa berkembang melalui teknologi *Natural Language Processing* (NLP) yang memungkinkan komputer memahami struktur bahasa manusia, melakukan analisis makna kalimat, serta menghasilkan teks secara otomatis. Penerapan NLP telah banyak digunakan pada sistem penerjemahan bahasa (*Machine Translation*), *chatbot*, *speech recognition*, *text summarization*, hingga *question answering system*.

Dalam beberapa tahun terakhir, penerapan *Artificial Intelligence* pada bahasa daerah juga mulai berkembang. Susanto et al. (2023) menjelaskan bahwa pendekatan *Neural Machine Translation* mampu meningkatkan kualitas penerjemahan pada bahasa daerah Indonesia

meskipun masih menghadapi keterbatasan jumlah dataset dan kompleksitas struktur bahasa. Oleh karena itu, pengembangan sistem penerjemahan berbasis AI memerlukan dataset yang berkualitas, *model* penerjemahan yang mampu memahami konteks kalimat, serta integrasi dengan teknologi pendukung sehingga hasil terjemahan menjadi lebih akurat, alami, dan mudah dipahami.

Menurut Jurafsky dan Martin (2023), perkembangan *Natural Language Processing* memungkinkan komputer memahami struktur bahasa, makna semantik, dan konteks kalimat sehingga mampu menghasilkan sistem penerjemahan yang lebih akurat. seiring perkembangan *Large Language Model* (LLM), penerapan *Artificial Intelligence* pada bidang penerjemahan bahasa tidak lagi hanya mengandalkan *model* berbasis dataset lokal, tetapi juga memanfaatkan *model* bahasa berbasis *cloud* yang memiliki kemampuan memahami konteks kalimat secara lebih komprehensif. Integrasi antara *AI Translation Engine* berbasis dataset lokal dengan *Large Language Model* (LLM) memungkinkan sistem penerjemahan memiliki fleksibilitas yang lebih tinggi, baik pada kondisi *offline* maupun *online*. Pendekatan ini dikenal sebagai *Hybrid AI Translation*, yaitu mekanisme yang menggabungkan dua sumber penerjemahan untuk meningkatkan kualitas hasil terjemahan sesuai kondisi jaringan dan kebutuhan pengguna.

Pada penelitian ini, *Artificial Intelligence* menjadi teknologi utama dalam pengembangan aplikasi LingoNusantara AI. AI digunakan untuk membangun *AI Translation Engine* yang terdiri atas *Dictionary Engine*, *Phrase Engine*, *Sentence Engine*, *Grammar Engine*, *Context Engine*, dan *Idiom Engine*, sehingga proses penerjemahan Bahasa Indonesia ke Bahasa Sunda tidak hanya dilakukan berdasarkan pencocokan kata, tetapi juga mempertimbangkan struktur kalimat, konteks percakapan, serta ragam Bahasa Sunda. Selain itu, sistem juga mengintegrasikan *Google Gemini* sebagai *Large Language Model* (LLM) melalui mekanisme *Hybrid AI Translation* untuk menghasilkan terjemahan yang lebih alami dan kontekstual ketika perangkat terhubung ke internet. Selanjutnya, hasil terjemahan diproses menggunakan *Edge Text-to-Speech* (Edge TTS) dan *Retrieval-Based Voice Conversion* (RVC) sehingga menghasilkan keluaran suara yang tetap mempertahankan karakteristik vokal pengguna.

2.3 Machine Learning

Machine Learning merupakan salah satu cabang dari *Artificial Intelligence* (AI) yang berfokus pada pengembangan algoritma dan model komputer agar mampu mempelajari pola dari data tanpa harus diprogram secara eksplisit untuk setiap permasalahan. Menurut Alpaydin

(2024), *Machine Learning* memungkinkan komputer meningkatkan performanya melalui pengalaman yang diperoleh dari data sehingga sistem mampu melakukan prediksi, klasifikasi, maupun pengambilan keputusan secara otomatis.

Menurut Russell dan Norvig (2021), *Machine Learning* merupakan pendekatan yang memungkinkan suatu sistem mempelajari hubungan antar data, mengenali pola, serta menghasilkan *model* yang dapat digunakan untuk menyelesaikan berbagai permasalahan. Berbeda dengan pemrograman konvensional yang bergantung pada aturan (*rule-based*), *Machine Learning* memanfaatkan data sebagai sumber utama pembelajaran sehingga sistem mampu beradaptasi terhadap berbagai kondisi yang belum pernah ditemui sebelumnya.

Menurut Alpaydin (2024), *Machine Learning* secara umum dibagi menjadi tiga kategori utama, yaitu *Supervised Learning*, *Unsupervised Learning*, dan *Reinforcement Learning*. *Supervised Learning* menggunakan data berlabel (*labeled data*) untuk membangun model prediksi, *Unsupervised Learning* digunakan untuk menemukan pola pada data tanpa label, sedangkan *Reinforcement Learning* mempelajari tindakan terbaik berdasarkan mekanisme penghargaan (*reward*) dan hukuman (*penalty*).

Perkembangan *Machine Learning* telah memberikan kontribusi yang signifikan dalam berbagai bidang, seperti pengenalan wajah, pengenalan suara, analisis citra, sistem rekomendasi, serta pengolahan bahasa alami (*Natural Language Processing*). Pada bidang penerjemahan bahasa, *Machine Learning* menjadi dasar dalam pengembangan berbagai metode *Machine Translation*, termasuk *Statistical Machine Translation* (SMT) dan *Neural Machine Translation* (NMT), yang mampu menghasilkan terjemahan lebih akurat dibandingkan metode berbasis aturan (*rule-based*) (Jurafsky & Martin, 2023).

Dalam penelitian ini, konsep *Machine Learning* diterapkan pada beberapa komponen sistem LingoNusantara AI. *AI Translation Engine* memanfaatkan dataset Bahasa Indonesia–Bahasa Sunda sebagai basis pengetahuan untuk menghasilkan terjemahan berdasarkan kosakata, *frasa*, struktur kalimat, konteks, dan *idiom*. Selain itu, sistem juga mengintegrasikan *Google Gemini* sebagai *Large Language Model* (LLM) berbasis *cloud* melalui mekanisme *Hybrid AI Translation* untuk meningkatkan kualitas penerjemahan ketika perangkat terhubung ke internet.

Machine Learning juga berperan pada proses *Retrieval-Based Voice Conversion* (RVC). *Model* RVC yang digunakan pada penelitian ini diperoleh melalui proses pelatihan (*training*) menggunakan dataset suara pengguna sehingga mampu mempelajari karakteristik vokal, intonasi, dan warna suara. *Model* hasil pelatihan tersebut kemudian digunakan pada

proses inferensi untuk mengubah suara hasil *Edge Text-to-Speech* (Edge TTS) menjadi suara yang memiliki karakteristik menyerupai suara pengguna.

2.4 Deep Learning

Deep Learning merupakan salah satu cabang dari *Machine Learning* yang menggunakan jaringan saraf tiruan (*Artificial Neural Network*) dengan banyak lapisan (*multi-layer neural network*) untuk mempelajari representasi data secara otomatis. Menurut Zhang, Lipton, Li, dan Smola (2023), *Deep Learning* memungkinkan sistem mempelajari pola yang kompleks dari data dalam jumlah besar sehingga mampu menghasilkan performa yang lebih baik dibandingkan metode *Machine Learning* konvensional pada berbagai permasalahan, seperti pengolahan citra, pengenalan suara, serta pemrosesan bahasa alami.

Menurut Russell dan Norvig (2021), perkembangan *Deep Learning* telah membawa perubahan besar pada bidang *Artificial Intelligence* karena mampu melakukan proses pembelajaran secara end-to-end tanpa memerlukan ekstraksi fitur secara manual. Melalui proses pelatihan (*training*), *model Deep Learning* dapat mempelajari hubungan antar data secara bertahap sehingga menghasilkan representasi yang lebih abstrak dan akurat.

Dalam bidang *Natural Language Processing* (NLP), *Deep Learning* digunakan untuk membangun berbagai *model* penerjemahan bahasa, *chatbot*, sistem tanya jawab, analisis sentimen, serta *Large Language Model* (LLM). Menurut Jurafsky dan Martin (2023), penerapan *Deep Learning* pada NLP mampu meningkatkan kemampuan sistem dalam memahami konteks kalimat, hubungan antar kata, serta makna semantik sehingga menghasilkan keluaran yang lebih alami dibandingkan pendekatan berbasis aturan (*rule-based*).

Selain pada bidang bahasa, *Deep Learning* juga banyak diterapkan pada teknologi pengolahan suara (*speech processing*). Menurut Hsu et al. (2021), *model* HuBERT (*Hidden Unit BERT*) memanfaatkan pendekatan *self-supervised learning* untuk mempelajari representasi akustik dari data suara tanpa memerlukan anotasi dalam jumlah besar. Representasi suara yang dihasilkan HuBERT memiliki kualitas yang tinggi sehingga banyak digunakan sebagai *feature extractor* pada berbagai penelitian *Voice Conversion*, termasuk *Retrieval-Based Voice Conversion* (RVC).

Pada teknologi *Retrieval-Based Voice Conversion* (RVC), *Deep Learning* digunakan untuk mempelajari karakteristik suara berdasarkan dataset hasil pelatihan sehingga *model* mampu mempertahankan identitas vokal pengguna pada saat proses konversi suara dilakukan.

Model yang telah dilatih kemudian digunakan pada tahap inferensi untuk mengubah suara hasil *Edge Text-to-Speech* (Edge TTS) menjadi suara yang memiliki karakteristik menyerupai suara asli pengguna (RVC Project, 2023).

Dalam penelitian ini, *Deep Learning* diterapkan pada dua komponen utama sistem LingoNusantara AI. Komponen pertama adalah *Google Gemini* sebagai *Large Language Model* (LLM) yang dimanfaatkan pada mekanisme *Hybrid AI Translation* untuk menghasilkan terjemahan yang lebih kontekstual ketika perangkat terhubung ke internet. Komponen kedua adalah *Retrieval-Based Voice Conversion* (RVC) melalui *platform Applio* yang memanfaatkan *model* hasil pelatihan untuk menghasilkan suara terjemahan dengan karakteristik vokal yang menyerupai suara pengguna. Integrasi kedua teknologi tersebut memungkinkan aplikasi tidak hanya menghasilkan terjemahan teks yang lebih alami, tetapi juga keluaran suara yang lebih personal dan natural.

2.5 Natural Language Processing (NLP)

Natural Language Processing (NLP) merupakan salah satu cabang dari *Artificial Intelligence* (AI) yang mempelajari interaksi antara komputer dan bahasa manusia. Tujuan utama NLP adalah memungkinkan komputer memahami, menganalisis, mengolah, serta menghasilkan bahasa alami sehingga manusia dapat berkomunikasi dengan komputer menggunakan bahasa sehari-hari. Menurut Jurafsky dan Martin (2023), NLP merupakan bidang yang menggabungkan ilmu linguistik, kecerdasan buatan, serta ilmu komputer untuk membangun sistem yang mampu memahami makna, struktur, dan konteks bahasa manusia.

Menurut Russell dan Norvig (2021), penerapan NLP tidak hanya berfokus pada proses pengenalan kata (*word recognition*), tetapi juga mencakup analisis sintaksis (*syntax*), analisis semantik (*semantic analysis*), analisis konteks (*context analysis*), serta pembentukan respons berdasarkan informasi yang dipahami oleh sistem. Kemampuan tersebut menjadikan NLP sebagai teknologi utama pada berbagai aplikasi, seperti *Machine Translation*, *chatbot*, *question answering system*, analisis sentimen, *speech recognition*, dan *text summarization*.

Perkembangan NLP semakin pesat dengan hadirnya pendekatan berbasis *Deep Learning* dan *Large Language Model* (LLM) yang mampu memahami hubungan antar kata dan konteks kalimat secara lebih baik dibandingkan metode konvensional. Menurut Jurafsky dan Martin (2023), *model* berbasis *Transformer* memungkinkan sistem menghasilkan teks yang lebih alami karena mampu mempelajari keterkaitan antar kata dalam suatu kalimat secara menyeluruh.

Dalam sistem penerjemahan bahasa, NLP memiliki peran penting dalam melakukan proses tokenisasi, analisis struktur kalimat, identifikasi makna kata, pemilihan padanan kata, serta penyusunan kembali hasil terjemahan sesuai kaidah bahasa tujuan. Oleh karena itu, penerapan NLP mampu meningkatkan kualitas hasil terjemahan dibandingkan pendekatan yang hanya mengandalkan pencocokan kosakata (*dictionary-based translation*).

Pada penelitian ini, teknologi *Natural Language Processing* (NLP) diterapkan melalui *AI Translation Engine* yang terdiri atas *Dictionary Engine*, *Phrase Engine*, *Grammar Engine*, *Sentence Engine*, *Context Engine*, dan *Idiom Engine*. Masing-masing komponen memiliki fungsi yang berbeda dalam memahami struktur bahasa, mengenali *frasa*, menentukan konteks kalimat, serta memilih padanan kata yang sesuai dengan ragam Bahasa Sunda sehingga menghasilkan terjemahan yang lebih akurat dan alami.

Selain menggunakan *AI Translation Engine* berbasis dataset lokal, penelitian ini juga mengintegrasikan *Google Gemini* sebagai *Large Language Model* (LLM) berbasis *cloud*. Integrasi tersebut membentuk mekanisme *Hybrid AI Translation*, yaitu sistem yang mampu memilih proses penerjemahan berdasarkan kondisi koneksi internet. Ketika perangkat terhubung ke internet, sistem memanfaatkan *Google Gemini* untuk menghasilkan terjemahan yang lebih kontekstual. Sebaliknya, ketika koneksi internet tidak tersedia, proses penerjemahan tetap dapat dilakukan menggunakan *AI Translation Engine* berbasis dataset lokal sehingga aplikasi tetap dapat digunakan dalam kondisi *offline*.

Dengan penerapan teknologi *Natural Language Processing* (NLP) dan *Hybrid AI Translation*, aplikasi LingoNusantara AI mampu menghasilkan terjemahan Bahasa Indonesia ke Bahasa Sunda yang lebih akurat, mempertimbangkan konteks kalimat, mendukung ragam bahasa Loma, Lemes, dan Kasar, serta memberikan fleksibilitas penggunaan baik pada kondisi *online* maupun *offline*.

2.6 Machine Translation

Machine Translation (MT) atau penerjemahan mesin merupakan salah satu bidang dalam *Natural Language Processing* (NLP) yang bertujuan untuk menerjemahkan teks secara otomatis dari bahasa sumber (*source language*) ke bahasa sasaran (*target language*) menggunakan bantuan komputer. Menurut Jurafsky dan Martin (2023), *Machine Translation* merupakan salah satu penerapan *Artificial Intelligence* yang terus berkembang melalui berbagai pendekatan, mulai dari metode berbasis aturan (*Rule-Based Machine Translation*), metode statistik (*Statistical Machine Translation*), hingga metode berbasis jaringan saraf (*Neural Machine Translation*).

Menurut Koehn (2020), perkembangan *Machine Translation* telah mengalami perubahan yang signifikan dari sistem yang hanya mengandalkan aturan tata bahasa menuju sistem yang mampu mempelajari pola bahasa secara otomatis menggunakan data dalam jumlah besar. Pendekatan *Neural Machine Translation* (NMT) menjadi metode yang paling banyak digunakan saat ini karena mampu menghasilkan terjemahan yang lebih alami, mempertimbangkan konteks kalimat, serta mengurangi kesalahan penerjemahan yang sering terjadi pada metode konvensional.

Menurut Jurafsky dan Martin (2023), kualitas hasil *Machine Translation* dipengaruhi oleh beberapa faktor, di antaranya kualitas dataset, jumlah data pelatihan, kompleksitas struktur bahasa, kemampuan *model* dalam memahami konteks, serta proses pembelajaran yang digunakan. Oleh karena itu, pengembangan sistem penerjemahan modern tidak hanya bergantung pada kamus atau aturan bahasa, tetapi juga memanfaatkan *model Artificial Intelligence* yang mampu memahami hubungan antar kata dan makna kalimat secara lebih komprehensif.

Dalam beberapa tahun terakhir, perkembangan *Large Language Model* (LLM) telah memberikan peningkatan yang signifikan terhadap kualitas *Machine Translation*. *Model* berbasis *Transformer* mampu memahami konteks kalimat secara lebih luas sehingga menghasilkan terjemahan yang lebih alami dan sesuai dengan maksud penutur. Salah satu implementasi teknologi tersebut adalah *Google Gemini*, yang menyediakan layanan penerjemahan berbasis *cloud* melalui antarmuka *Application Programming Interface* (API).

Pada penelitian ini, proses *Machine Translation* tidak hanya mengandalkan satu metode penerjemahan, tetapi menggunakan pendekatan *Hybrid AI Translation* yang menggabungkan *AI Translation Engine* berbasis dataset lokal dengan *Google Gemini* sebagai *Large Language Model* (LLM) berbasis *cloud*. *AI Translation Engine* digunakan sebagai penerjemah utama

ketika perangkat tidak memiliki koneksi internet sehingga aplikasi tetap dapat beroperasi secara *offline*. Sementara itu, ketika perangkat terhubung ke internet, sistem memanfaatkan *Google Gemini* untuk menghasilkan terjemahan yang lebih alami, kontekstual, dan sesuai dengan struktur Bahasa Sunda.

AI Translation Engine yang dikembangkan pada penelitian ini terdiri atas *Dictionary Engine*, *Phrase Engine*, *Grammar Engine*, *Sentence Engine*, *Context Engine*, dan *Idiom Engine*. Keenam komponen tersebut bekerja secara terintegrasi untuk menganalisis kosakata, mengenali *frasa*, memahami struktur kalimat, menentukan konteks, serta memilih padanan kata yang sesuai dengan ragam Bahasa Sunda. Pendekatan ini memungkinkan sistem menghasilkan terjemahan yang lebih baik dibandingkan metode yang hanya mengandalkan pencocokan kata (*dictionary-based translation*).

Melalui penerapan *Hybrid AI Translation*, aplikasi LingoNusantara AI mampu memanfaatkan keunggulan kedua pendekatan tersebut, yaitu ketersediaan layanan penerjemahan secara *offline* menggunakan *AI Translation Engine* berbasis dataset lokal serta peningkatan kualitas terjemahan secara *online* menggunakan *Google Gemini*. Pendekatan ini memberikan fleksibilitas penggunaan sekaligus meningkatkan kualitas hasil terjemahan sesuai dengan kondisi jaringan yang tersedia.

2.7 Karakteristik Bahasa Sunda

Bahasa Sunda merupakan salah satu bahasa daerah terbesar di Indonesia yang digunakan oleh masyarakat di Provinsi Jawa Barat, Banten, serta sebagian wilayah Jawa Tengah. Menurut Badan Pengembangan dan Pembinaan Bahasa (2024), Bahasa Sunda masih menjadi salah satu bahasa daerah dengan jumlah penutur terbanyak di Indonesia serta memiliki peran penting sebagai identitas budaya, media komunikasi, dan sarana pelestarian nilai-nilai kearifan lokal.

Menurut Sudaryat (2022), Bahasa Sunda memiliki karakteristik yang berbeda dengan Bahasa Indonesia karena mengenal sistem *undak usuk basa*, yaitu tingkatan bahasa yang digunakan berdasarkan hubungan sosial, usia, dan situasi komunikasi antara penutur dan lawan bicara. Penggunaan ragam bahasa yang tepat menjadi salah satu unsur penting dalam menjaga etika dan kesantunan berbahasa dalam masyarakat Sunda.

Secara umum, ragam Bahasa Sunda terdiri atas tiga tingkatan utama, yaitu Loma, Lemes, dan Kasar. Ragam Loma digunakan dalam komunikasi sehari-hari dengan teman sebaya atau orang yang telah akrab. Ragam Lemes digunakan ketika berkomunikasi dengan

orang yang lebih tua atau dalam situasi yang bersifat formal, sedangkan ragam Kasar digunakan pada kondisi tertentu sesuai konteks sosial dan hubungan antarpemutar (Sudaryat, 2022).

Perbedaan ragam Bahasa Sunda tidak hanya terlihat pada pemilihan kosakata, tetapi juga pada struktur kalimat, penggunaan kata ganti orang, serta ungkapan yang digunakan. Sebagai contoh, kata "saya" dapat diterjemahkan menjadi *urang* pada ragam Loma dan *abdi* pada ragam Lemes. Demikian pula kata kerja "makan" dapat diterjemahkan menjadi *dahar* atau *tuang* sesuai dengan ragam bahasa dan konteks komunikasi (Sudaryat, 2022).

Selain memiliki tingkatan bahasa, pengembangan sistem penerjemahan Bahasa Sunda juga menghadapi berbagai tantangan. Menurut Shi *et al.* (2022) dan Haddow *et al.* (2022), bahasa dengan sumber daya rendah (*low-resource language*) umumnya memiliki keterbatasan *dataset* paralel berkualitas, variasi kosakata yang dipengaruhi oleh wilayah penggunaan, serta kompleksitas dalam memahami konteks semantik dan pragmatik. Karakteristik tersebut juga ditemukan pada Bahasa Sunda sehingga proses penerjemahan tidak cukup dilakukan melalui pencocokan kosakata, tetapi memerlukan pemahaman terhadap konteks kalimat dan tingkat kesopanan bahasa.

Menurut Jurafsky dan Martin (2023), sistem penerjemahan modern harus mampu mempertimbangkan aspek semantik, sintaksis, dan pragmatik agar hasil terjemahan terdengar alami serta sesuai dengan budaya bahasa sasaran. Oleh karena itu, pemilihan ragam bahasa menjadi salah satu tantangan utama dalam pengembangan sistem penerjemahan Bahasa Sunda berbasis *Artificial Intelligence*.

Pada penelitian ini, aplikasi LingoNusantara AI mendukung tiga ragam Bahasa Sunda, yaitu Loma, Lemes, dan Kasar. Proses pemilihan ragam bahasa dilakukan berdasarkan pilihan pengguna pada antarmuka aplikasi. Selanjutnya, *AI Translation Engine* menyesuaikan proses penerjemahan menggunakan *Dictionary Engine*, *Phrase Engine*, *Grammar Engine*, *Sentence Engine*, *Context Engine*, dan *Idiom Engine* sesuai dengan ragam bahasa yang dipilih. Selain itu, ketika perangkat terhubung ke internet, sistem memanfaatkan Google Gemini melalui mekanisme *Hybrid AI Translation* untuk menghasilkan terjemahan yang lebih kontekstual tanpa mengabaikan ragam Bahasa Sunda yang dipilih oleh pengguna.

Dengan mempertimbangkan karakteristik Bahasa Sunda, tantangan penerjemahan pada bahasa dengan sumber daya terbatas, serta penerapan *Hybrid AI Translation*, aplikasi LingoNusantara AI diharapkan mampu menghasilkan terjemahan yang tidak hanya benar secara linguistik, tetapi juga sesuai dengan norma kesantunan dan budaya masyarakat Sunda.

2.8 *AI Translation Engine*

AI Translation Engine merupakan mesin penerjemah berbasis *Artificial Intelligence* (AI) yang dikembangkan untuk menerjemahkan teks dari bahasa sumber ke bahasa sasaran dengan mempertimbangkan kosakata, struktur kalimat, konteks, serta makna semantik. Berbeda dengan sistem penerjemahan berbasis kamus (*dictionary-based translation*) yang hanya melakukan pencocokan kata secara langsung, *AI Translation Engine* memanfaatkan beberapa komponen yang saling terintegrasi sehingga mampu menghasilkan terjemahan yang lebih akurat dan kontekstual (Jurafsky & Martin, 2023).

Menurut Russell dan Norvig (2021), sistem *Artificial Intelligence* yang baik tidak hanya mampu melakukan pencarian informasi, tetapi juga harus mampu melakukan penalaran (*reasoning*), memahami konteks, serta memilih tindakan yang paling sesuai berdasarkan informasi yang tersedia. Konsep tersebut menjadi dasar dalam pengembangan *AI Translation Engine* pada penelitian ini sehingga proses penerjemahan tidak hanya bergantung pada satu metode, tetapi menggabungkan beberapa proses analisis bahasa.

Pada penelitian ini, *AI Translation Engine* dikembangkan menggunakan enam komponen utama yang bekerja secara berurutan dalam menghasilkan terjemahan Bahasa Indonesia ke Bahasa Sunda. Setiap komponen memiliki fungsi yang berbeda sehingga proses penerjemahan tidak hanya mempertimbangkan arti kata, tetapi juga memperhatikan struktur kalimat, hubungan antar kata, serta ragam Bahasa Sunda yang dipilih oleh pengguna.

2.8.1 *Dictionary Engine*

Dictionary Engine merupakan komponen yang bertugas menerjemahkan kosakata dasar dari Bahasa Indonesia ke Bahasa Sunda berdasarkan kamus yang tersimpan pada dataset lokal. Komponen ini digunakan sebagai proses awal untuk mencari padanan kata yang sesuai sehingga dapat mempercepat proses penerjemahan. Menurut Jurafsky dan Martin (2023), kamus digital masih memiliki peranan penting dalam sistem *Machine Translation* sebagai sumber pengetahuan awal sebelum dilakukan analisis bahasa yang lebih kompleks.

2.8.2 *Phrase Engine*

Phrase Engine berfungsi mengenali *frasa* atau gabungan beberapa kata yang memiliki makna khusus sehingga tidak diterjemahkan secara terpisah. Dengan mengenali *frasa*, sistem dapat menghasilkan terjemahan yang lebih alami dan sesuai dengan konteks kalimat.

Pendekatan ini mengurangi kesalahan yang sering muncul apabila setiap kata diterjemahkan secara individual.

2.8.3 Grammar Engine

Grammar Engine bertugas menganalisis struktur tata bahasa (*grammar*) pada kalimat Bahasa Indonesia sehingga susunan kalimat hasil terjemahan tetap mengikuti kaidah Bahasa Sunda. Komponen ini membantu sistem menentukan urutan kata, bentuk kalimat, serta hubungan antar unsur bahasa agar hasil terjemahan lebih mudah dipahami.

2.8.4 Sentence Engine

Sentence Engine digunakan untuk mengenali pola kalimat secara keseluruhan. Apabila suatu kalimat telah tersedia pada dataset lokal, sistem dapat langsung menggunakan hasil terjemahan yang telah disimpan sehingga proses penerjemahan menjadi lebih cepat dan konsisten. Pendekatan ini juga mengurangi kemungkinan kesalahan penerjemahan pada kalimat yang sering digunakan.

2.8.5 Context Engine

Context Engine berfungsi menganalisis konteks percakapan sehingga sistem mampu memilih padanan kata yang sesuai dengan makna kalimat. Menurut Jurafsky dan Martin (2023), analisis konteks merupakan salah satu aspek penting dalam *Natural Language Processing* karena suatu kata dapat memiliki arti yang berbeda bergantung pada kalimat yang digunakan.

2.8.6 Idiom Engine

Idiom Engine bertugas mengenali ungkapan atau *idiom* yang memiliki makna khusus sehingga tidak diterjemahkan secara harfiah. Dengan adanya komponen ini, sistem mampu menghasilkan terjemahan yang lebih alami dan sesuai dengan kebiasaan penggunaan Bahasa Sunda dalam kehidupan sehari-hari.

Seluruh komponen tersebut bekerja secara terintegrasi dalam *AI Translation Engine* untuk menghasilkan terjemahan Bahasa Indonesia ke Bahasa Sunda yang lebih akurat. *AI Translation Engine* menggunakan dataset lokal sebagai basis pengetahuan sehingga aplikasi tetap dapat melakukan proses penerjemahan ketika perangkat tidak memiliki koneksi internet. Selanjutnya, hasil penerjemahan dari *AI Translation Engine* dapat dipadukan dengan *Google Gemini* melalui mekanisme *Hybrid AI Translation* untuk menghasilkan terjemahan yang lebih

kontekstual ketika perangkat terhubung ke internet. Pendekatan tersebut memberikan fleksibilitas penggunaan sistem baik dalam kondisi *offline* maupun *online*.

Pada penelitian ini, *AI Translation Engine* menjadi komponen utama dalam aplikasi LingoNusantara AI. Hasil terjemahan yang dihasilkan kemudian diteruskan ke proses *Edge Text-to-Speech* (Edge TTS) untuk menghasilkan suara dasar dan selanjutnya diproses menggunakan *Retrieval-Based Voice Conversion* (RVC) sehingga menghasilkan keluaran suara yang mempertahankan karakteristik vokal pengguna.

2.9 Hybrid AI Translation

Hybrid AI Translation merupakan pendekatan penerjemahan yang menggabungkan dua atau lebih metode penerjemahan untuk memperoleh hasil yang lebih akurat, kontekstual, serta fleksibel sesuai dengan kondisi sistem. Pada pendekatan ini, setiap metode memiliki peran yang saling melengkapi sehingga kelemahan dari satu metode dapat ditutupi oleh metode lainnya. Menurut Jurafsky dan Martin (2023), sistem penerjemahan modern tidak hanya bergantung pada satu *model* penerjemahan, tetapi dapat mengombinasikan berbagai sumber pengetahuan agar mampu memahami makna kalimat secara lebih baik.

Perkembangan *Large Language Model* (LLM) telah memberikan perubahan yang signifikan terhadap teknologi *Machine Translation*. *Model* berbasis *Transformer* memiliki kemampuan memahami hubungan antar kata, konteks kalimat, serta makna semantik yang lebih kompleks dibandingkan pendekatan penerjemahan konvensional. Salah satu implementasi teknologi tersebut adalah *Google Gemini*, yang menyediakan layanan penerjemahan berbasis *cloud* melalui *Application Programming Interface* (API).

Meskipun demikian, penggunaan layanan berbasis *cloud* memiliki beberapa keterbatasan, seperti ketergantungan terhadap koneksi internet, waktu respons layanan, serta batas penggunaan (*API quota* atau *rate limit*) yang ditentukan oleh penyedia layanan. Oleh karena itu, sistem penerjemahan modern memerlukan mekanisme yang tetap mampu beroperasi ketika layanan berbasis *cloud* tidak tersedia.

Pada penelitian ini diterapkan pendekatan *Hybrid AI Translation* dengan menggabungkan *AI Translation Engine* berbasis dataset lokal dan *Google Gemini* sebagai *Large Language Model* (LLM) berbasis *cloud*. *AI Translation Engine* berfungsi sebagai penerjemah lokal yang dapat digunakan ketika perangkat tidak memiliki koneksi internet, sedangkan *Google Gemini* dimanfaatkan untuk menghasilkan terjemahan yang lebih alami dan kontekstual ketika perangkat terhubung ke internet.

Mekanisme *Hybrid AI Translation* pada penelitian ini diawali dengan pemeriksaan status koneksi internet. Apabila perangkat terhubung ke internet dan layanan *Google Gemini* tersedia, sistem akan mengirimkan teks Bahasa Indonesia beserta parameter ragam Bahasa Sunda yang dipilih pengguna kepada *Google Gemini* melalui API. Selanjutnya, hasil terjemahan diterima oleh backend dan diteruskan ke proses sintesis suara menggunakan *Edge Text-to-Speech* (Edge TTS) sebelum dikonversi menggunakan *Retrieval-Based Voice Conversion* (RVC).

Sebaliknya, apabila perangkat tidak memiliki koneksi internet atau layanan *Google Gemini* tidak dapat digunakan, sistem secara otomatis menggunakan *AI Translation Engine* berbasis dataset lokal. Engine tersebut memanfaatkan *Dictionary Engine*, *Phrase Engine*, *Grammar Engine*, *Sentence Engine*, *Context Engine*, dan *Idiom Engine* untuk menghasilkan terjemahan berdasarkan basis pengetahuan yang tersimpan pada sistem sehingga proses penerjemahan tetap dapat dilakukan tanpa bergantung pada layanan eksternal.

Pendekatan *Hybrid AI Translation* memberikan beberapa keunggulan, yaitu meningkatkan fleksibilitas sistem, menjaga ketersediaan layanan penerjemahan pada kondisi *online* maupun *offline*, serta menghasilkan kualitas terjemahan yang lebih baik melalui pemanfaatan *Large Language Model* (LLM) ketika koneksi internet tersedia. Dengan demikian, aplikasi LingoNusantara AI mampu memberikan pengalaman penerjemahan yang lebih adaptif sesuai dengan kondisi penggunaan.

2.10 Google Gemini

Google Gemini merupakan *Large Language Model* (LLM) yang dikembangkan oleh Google untuk memahami, menghasilkan, serta memproses berbagai jenis informasi, termasuk teks, gambar, audio, video, dan kode program. Menurut Google AI (2024), Gemini dirancang menggunakan arsitektur berbasis *Transformer* yang mampu memahami hubungan antar kata, konteks kalimat, serta makna semantik sehingga menghasilkan keluaran yang lebih alami dibandingkan pendekatan penerjemahan konvensional.

Large Language Model (LLM) merupakan *model* kecerdasan buatan yang dilatih menggunakan data dalam jumlah sangat besar sehingga mampu memahami pola bahasa, struktur kalimat, hubungan antar kata, serta konteks percakapan. Menurut Jurafsky dan Martin (2023), perkembangan LLM telah memberikan peningkatan yang signifikan pada berbagai bidang *Natural Language Processing* (NLP), seperti penerjemahan bahasa (*Machine*

Translation), peringkasan teks (*Text Summarization*), sistem tanya jawab (*Question Answering System*), hingga asisten virtual (*Virtual Assistant*).

Google Gemini menyediakan layanan berbasis *Application Programming Interface* (API) yang memungkinkan pengembang mengintegrasikan kemampuan *Artificial Intelligence* ke dalam aplikasi yang dikembangkan. Melalui API tersebut, aplikasi dapat mengirimkan masukan (*prompt*) berupa teks kepada *model Gemini*, kemudian menerima keluaran berupa respons yang dihasilkan berdasarkan pemahaman konteks dari *model* tersebut. Pendekatan ini memberikan fleksibilitas dalam membangun aplikasi berbasis AI tanpa harus melakukan pelatihan *model* secara mandiri (Google AI, 2024).

Pada penelitian ini, *Google Gemini* digunakan sebagai penerjemah berbasis *cloud* yang menjadi bagian dari mekanisme *Hybrid AI Translation*. Ketika perangkat memiliki koneksi internet, backend *Retrieval-Based Voice Conversion* akan mengirimkan teks Bahasa Indonesia beserta parameter ragam Bahasa Sunda yang dipilih pengguna ke layanan *Google Gemini* melalui API. Selanjutnya, *Google Gemini* menghasilkan terjemahan Bahasa Sunda yang lebih alami dan kontekstual berdasarkan pemahaman terhadap struktur kalimat, makna semantik, serta konteks percakapan.

Hasil terjemahan dari *Google Gemini* selanjutnya diteruskan ke proses Edge *Text-to-Speech* (Edge TTS) untuk menghasilkan suara dasar, kemudian dikonversi menggunakan *Retrieval-Based Voice Conversion* (RVC) sehingga menghasilkan keluaran suara yang memiliki karakteristik menyerupai suara pengguna. Dengan mekanisme tersebut, *Google Gemini* berperan sebagai peningkat kualitas hasil terjemahan, sedangkan proses sintesis dan personalisasi suara tetap dilakukan oleh komponen lain dalam sistem.

Meskipun memiliki kemampuan penerjemahan yang baik, penggunaan *Google Gemini* memiliki beberapa keterbatasan, seperti memerlukan koneksi internet yang stabil, bergantung pada ketersediaan layanan API, serta memiliki batas penggunaan (*API quota*) sesuai kebijakan penyedia layanan. Oleh karena itu, penelitian ini menerapkan konsep *Hybrid AI Translation* dengan memanfaatkan *AI Translation Engine* berbasis dataset lokal sebagai mekanisme *fallback*. Ketika layanan *Google Gemini* tidak tersedia, sistem secara otomatis menggunakan *AI Translation Engine* sehingga proses penerjemahan tetap dapat dilakukan secara *offline*.

Integrasi *Google Gemini* dengan *AI Translation Engine* memberikan beberapa keuntungan, antara lain meningkatkan kualitas hasil terjemahan berdasarkan konteks kalimat, memperluas cakupan kosakata yang dapat diterjemahkan, serta menjaga keberlangsungan layanan penerjemahan baik pada kondisi *online* maupun *offline*. Dengan demikian, aplikasi

LingoNusantara AI mampu memberikan pengalaman penerjemahan yang lebih fleksibel, adaptif, dan sesuai dengan kebutuhan pengguna.

2.11 Flask Framework

Retrieval-Based Voice Conversion merupakan salah satu micro web framework berbasis bahasa pemrograman Python yang digunakan untuk membangun aplikasi web dan layanan *Application Programming Interface* (API). Menurut Grinberg (2018), *Retrieval-Based Voice Conversion* dirancang dengan konsep sederhana (*minimalist framework*) sehingga hanya menyediakan komponen inti yang diperlukan dalam pengembangan aplikasi. Meskipun memiliki ukuran yang ringan, *Retrieval-Based Voice Conversion* tetap mampu dikembangkan menjadi sistem berskala besar melalui penambahan berbagai pustaka (*library*) dan ekstensi sesuai kebutuhan.

Menurut Ronacher (2023), *Retrieval-Based Voice Conversion* mendukung pengembangan aplikasi berbasis Representational State Transfer (REST) sehingga banyak digunakan dalam pembangunan layanan backend yang berkomunikasi dengan aplikasi *mobile*, aplikasi web, maupun sistem berbasis Internet of Things (IoT). Melalui mekanisme REST API, proses pertukaran data dilakukan menggunakan protokol HTTP dengan format data yang umum digunakan, seperti JavaScript Object Notation (JSON).

Retrieval-Based Voice Conversion memiliki beberapa keunggulan, antara lain mudah dipelajari, memiliki dokumentasi yang lengkap, mendukung berbagai pustaka Python, serta memiliki fleksibilitas tinggi dalam pengembangan aplikasi berbasis *Artificial Intelligence*. Selain itu, *Retrieval-Based Voice Conversion* dapat dengan mudah diintegrasikan dengan berbagai teknologi pendukung seperti *SQLite*, *Google Gemini* API, *Edge Text-to-Speech* (Edge TTS), dan *Retrieval-Based Voice Conversion* (RVC) sehingga sesuai digunakan sebagai backend pada penelitian ini.

Pada penelitian ini, *Retrieval-Based Voice Conversion* digunakan sebagai backend REST API pada aplikasi LingoNusantara AI. Backend bertugas menerima permintaan (*request*) dari aplikasi *Flutter*, memproses autentikasi pengguna, menjalankan *AI Translation Engine*, menghubungkan sistem dengan *Google Gemini* sebagai layanan penerjemah berbasis *cloud*, mengelola mekanisme *Hybrid AI Translation*, melakukan sintesis suara menggunakan *Edge Text-to-Speech* (Edge TTS), menjalankan proses *Retrieval-Based Voice Conversion* (RVC) melalui *platform Applio*, serta mengelola penyimpanan data ke dalam basis data *SQLite*.

Komunikasi antara aplikasi *Flutter* dan backend *Retrieval-Based Voice Conversion* dilakukan menggunakan REST API melalui protokol HTTP. Setiap permintaan yang dikirimkan oleh aplikasi berupa teks, parameter ragam Bahasa Sunda, maupun parameter konversi suara akan diproses oleh backend, kemudian hasil penerjemahan, informasi proses, serta file audio hasil konversi dikirim kembali kepada aplikasi dalam format JSON dan file audio WAV.

Dengan memanfaatkan *Retrieval-Based Voice Conversion* sebagai backend, seluruh proses pengolahan data dapat dipusatkan pada server sehingga aplikasi *Flutter* hanya berfungsi sebagai antarmuka pengguna (*frontend*). Pendekatan ini mempermudah proses pengembangan, pemeliharaan, serta pengembangan fitur baru pada aplikasi LingoNusantara AI.

2.12 Flutter Framework

Flutter merupakan Software Development Kit (SDK) yang dikembangkan oleh Google untuk membangun aplikasi berbasis *mobile*, web, dan desktop menggunakan satu basis kode (*single codebase*). Menurut Google Developers (2024), *Flutter* menggunakan bahasa pemrograman Dart serta menyediakan berbagai komponen antarmuka (*widget*) yang memungkinkan pengembang membangun aplikasi dengan tampilan yang konsisten pada berbagai *platform*.

Menurut Majid (2021), *Flutter* menerapkan konsep widget-based architecture, yaitu seluruh komponen antarmuka dibangun menggunakan widget yang dapat digunakan kembali (*reusable*). Pendekatan tersebut mempermudah proses pengembangan aplikasi, meningkatkan konsistensi tampilan, serta mempercepat proses pemeliharaan sistem.

Salah satu keunggulan *Flutter* adalah fitur Hot Reload, yang memungkinkan pengembang melihat perubahan kode secara langsung tanpa harus menjalankan ulang aplikasi secara keseluruhan. Selain itu, *Flutter* memiliki performa yang tinggi karena menggunakan mesin grafis (*rendering engine*) sendiri sehingga mampu menghasilkan antarmuka yang responsif dan nyaman digunakan pada berbagai perangkat.

Pada penelitian ini, *Flutter* digunakan sebagai *frontend* dalam pengembangan aplikasi LingoNusantara AI. Seluruh antarmuka pengguna (*User Interface*) dikembangkan menggunakan *Flutter*, mulai dari halaman Splash Screen, Onboarding, Login, Register, Dashboard, Translate, History, Profile, Settings, About, hingga Feedback. Setiap halaman dirancang untuk memberikan pengalaman pengguna (*User Experience*) yang sederhana,

mudah dipahami, dan mendukung seluruh proses penerjemahan Bahasa Indonesia ke Bahasa Sunda.

Komunikasi antara aplikasi *Flutter* dan backend dilakukan melalui REST API menggunakan protokol HTTP. Data yang dikirimkan berupa teks masukan, pilihan ragam Bahasa Sunda, parameter konversi suara, serta informasi autentikasi pengguna. Backend kemudian memproses permintaan tersebut menggunakan *Retrieval-Based Voice Conversion Framework*, menjalankan *AI Translation Engine* atau *Google Gemini* melalui mekanisme *Hybrid AI Translation*, melakukan sintesis suara menggunakan *Edge Text-to-Speech* (Edge TTS), serta menjalankan *Retrieval-Based Voice Conversion* (RVC). Hasil pemrosesan dikembalikan ke aplikasi *Flutter* dalam bentuk data JSON dan file audio WAV.

Selain berfungsi sebagai antarmuka pengguna, aplikasi *Flutter* juga menyediakan fitur untuk memutar hasil audio terjemahan, mengunduh file audio, melihat riwayat penerjemahan, mengelola profil pengguna, serta melakukan pengaturan aplikasi. Dengan memanfaatkan *Flutter*, aplikasi LingoNusantara AI mampu memberikan antarmuka yang responsif, mudah digunakan, dan terintegrasi dengan baik dengan backend berbasis *Retrieval-Based Voice Conversion*.

2.13 *Edge Text-to-Speech* (Edge TTS)

Edge Text-to-Speech (Edge TTS) merupakan layanan *Text-to-Speech* (TTS) yang dikembangkan oleh Microsoft untuk mengubah teks menjadi suara secara otomatis menggunakan teknologi kecerdasan buatan (*Artificial Intelligence*). Menurut Microsoft (2024), Edge TTS memanfaatkan *model Neural Text-to-Speech* yang mampu menghasilkan suara dengan intonasi, pelafalan, dan ritme yang lebih alami dibandingkan teknologi *Text-to-Speech* konvensional.

Text-to-Speech (TTS) merupakan salah satu bidang dalam *speech processing* yang bertujuan mengubah teks menjadi sinyal suara yang dapat dipahami oleh manusia. Menurut Jurafsky dan Martin (2023), perkembangan teknologi *Neural Text-to-Speech* telah meningkatkan kualitas sintesis suara secara signifikan sehingga suara yang dihasilkan terdengar lebih natural dan menyerupai cara berbicara manusia.

Edge TTS menyediakan berbagai pilihan bahasa, aksen, dan karakter suara yang dapat digunakan melalui *Application Programming Interface* (API). Pengguna dapat menentukan parameter seperti jenis suara (*voice model*), kecepatan berbicara (*speech rate*), serta tinggi rendah nada (*pitch*) sesuai kebutuhan aplikasi. Fleksibilitas tersebut menjadikan Edge TTS

banyak dimanfaatkan dalam pengembangan sistem berbasis *Artificial Intelligence*, asisten virtual, aplikasi pembelajaran, hingga sistem penerjemahan bahasa.

Pada penelitian ini, *Edge Text-to-Speech* (Edge TTS) digunakan sebagai tahap awal dalam proses pembentukan suara hasil terjemahan. Setelah *AI Translation Engine* atau *Google Gemini* melalui mekanisme *Hybrid AI Translation* menghasilkan teks Bahasa Sunda, backend *Retrieval-Based Voice Conversion* akan mengirimkan hasil terjemahan tersebut ke modul Edge TTS untuk menghasilkan file audio dasar dalam format WAV.

Audio hasil Edge TTS selanjutnya diteruskan ke proses *Retrieval-Based Voice Conversion* (RVC) menggunakan *platform Applio*. Pada tahap ini, *model* RVC akan mengubah karakteristik suara hasil sintesis menjadi suara yang menyerupai identitas vokal pengguna tanpa mengubah isi kalimat yang telah diterjemahkan. Dengan demikian, Edge TTS berfungsi sebagai sumber suara awal (*base voice*), sedangkan proses personalisasi suara dilakukan oleh teknologi *Retrieval-Based Voice Conversion* (RVC).

Penggunaan Edge TTS pada penelitian ini memberikan beberapa keuntungan, antara lain proses sintesis suara yang cepat, kualitas pelafalan yang baik, dukungan terhadap berbagai bahasa, serta kemudahan integrasi dengan backend berbasis Python. Selain itu, kombinasi antara *Edge Text-to-Speech* (Edge TTS) dan *Retrieval-Based Voice Conversion* (RVC) memungkinkan aplikasi LingoNusantara AI menghasilkan suara terjemahan yang lebih natural, jelas, serta memiliki karakteristik vokal yang menyerupai suara pengguna.

2.14 *Retrieval-Based Voice Conversion* (RVC)

Retrieval-Based Voice Conversion (RVC) merupakan salah satu teknologi *Voice Conversion* berbasis *Deep Learning* yang digunakan untuk mengubah karakteristik suara seseorang menjadi karakteristik suara lain tanpa mengubah isi atau makna dari ucapan tersebut. Berbeda dengan teknologi *Text-to-Speech* (TTS) yang menghasilkan suara sintetis secara langsung dari teks, RVC berfungsi mengubah karakteristik suara hasil sintesis menjadi suara yang menyerupai identitas vokal tertentu. Menurut Hsu et al. (2021), perkembangan teknologi *Voice Conversion* mengalami peningkatan yang signifikan melalui pemanfaatan *model* berbasis *Deep Learning* yang mampu mempelajari representasi akustik suara secara lebih akurat.

Menurut RVC Project (2023), *Retrieval-Based Voice Conversion* (RVC) memanfaatkan pendekatan berbasis HuBERT (*Hidden Unit BERT*) sebagai *feature extractor* untuk memperoleh representasi suara dari data pelatihan. Representasi tersebut kemudian

digunakan oleh *model* untuk mempelajari karakteristik vokal pengguna, seperti warna suara (*timbre*), intonasi, dan pola pelafalan sehingga mampu menghasilkan suara yang menyerupai identitas vokal pengguna pada saat proses inferensi dilakukan.

Prinsip kerja *Retrieval-Based Voice Conversion* (RVC) adalah mempertahankan isi atau makna ucapan, kemudian mengubah karakteristik vokal suara sumber menjadi karakteristik suara target berdasarkan model yang telah dilatih. Berbeda dengan teknologi *Text-to-Speech* (TTS) yang menghasilkan suara langsung dari teks, RVC memanfaatkan suara yang sudah ada sebagai masukan (*input*) untuk diubah karakteristik vokalnya. Dengan demikian, kalimat yang diucapkan tetap sama, sedangkan warna suara (*timbre*), intonasi, dan identitas vokalnya diubah agar menyerupai suara target.

Proses kerja RVC terdiri atas dua tahapan utama, yaitu training dan inference. Tahap *training* dilakukan dengan melatih *model* menggunakan dataset suara pengguna sehingga *model* dapat mempelajari karakteristik vokal yang akan digunakan sebagai identitas suara. Setelah proses pelatihan selesai, dihasilkan *model* dengan format .pth serta berkas indeks .index yang digunakan pada tahap *inference*. Pada tahap ini, suara masukan akan diproses menggunakan *model* yang telah dilatih sehingga menghasilkan suara baru dengan karakteristik vokal sesuai pengguna.

Menurut Hsu et al. (2021), kualitas hasil *Voice Conversion* dipengaruhi oleh beberapa faktor, antara lain kualitas dataset suara, jumlah data pelatihan, jumlah *epoch*, spesifikasi perangkat keras, serta metode ekstraksi fitur yang digunakan selama proses pelatihan. Semakin baik kualitas dataset dan semakin optimal proses pelatihan, maka hasil konversi suara yang dihasilkan akan semakin natural dan mendekati karakteristik suara asli pengguna.

Pada penelitian ini, *Retrieval-Based Voice Conversion* (RVC) digunakan untuk mengubah suara hasil Edge *Text-to-Speech* (Edge TTS) menjadi suara yang memiliki karakteristik menyerupai suara pengguna. Setelah proses penerjemahan selesai, backend *Retrieval-Based Voice Conversion* menghasilkan suara dasar menggunakan Edge TTS. Selanjutnya file audio tersebut diproses menggunakan *model* RVC yang telah dilatih sehingga menghasilkan keluaran suara dengan identitas vokal pengguna tanpa mengubah isi kalimat hasil terjemahan.

Implementasi RVC pada penelitian ini dilakukan menggunakan *platform Applio*, yang menyediakan antarmuka untuk proses pelatihan (*training*) maupun inferensi (*inference*) *model* suara. Backend *Retrieval-Based Voice Conversion* menjalankan proses inferensi melalui pemanggilan perintah Python menggunakan mekanisme subprocess, kemudian mengirimkan parameter *model*, nilai *pitch*, *index rate*, serta file audio hasil Edge TTS ke *Applio*. Setelah

proses inferensi selesai, file audio hasil konversi dikirim kembali ke aplikasi *Flutter* sehingga dapat diputar maupun diunduh oleh pengguna.

Meskipun mampu menghasilkan perubahan karakteristik suara yang cukup baik, kualitas hasil konversi pada penelitian ini masih dipengaruhi oleh keterbatasan spesifikasi perangkat keras yang digunakan selama proses pelatihan *model*. Proses pelatihan dilakukan menggunakan perangkat tanpa GPU NVIDIA sehingga jumlah *epoch* yang dapat dicapai masih terbatas. Kondisi tersebut menyebabkan karakteristik logat, intonasi, dan cengkok khas Bahasa Sunda belum dapat direpresentasikan secara optimal, meskipun identitas dasar suara pengguna telah berhasil dipertahankan.

Dengan memanfaatkan teknologi *Retrieval-Based Voice Conversion* (RVC), aplikasi LingoNusantara AI tidak hanya mampu menghasilkan terjemahan teks Bahasa Sunda, tetapi juga menghasilkan keluaran suara yang lebih personal dengan mempertahankan karakteristik vokal pengguna sehingga memberikan pengalaman penggunaan yang lebih alami dibandingkan sistem penerjemahan berbasis *Text-to-Speech* konvensional.

2.15 *Applio*

Applio merupakan *platform* open-source yang dikembangkan untuk mempermudah proses pelatihan (*training*) dan inferensi (*inference*) pada teknologi *Retrieval-Based Voice Conversion* (RVC). *Platform* ini menyediakan antarmuka pengguna (*Graphical User Interface/GUI*) sehingga pengguna dapat melakukan proses pengolahan dataset suara, pelatihan *model*, hingga konversi suara tanpa harus menjalankan seluruh proses melalui perintah terminal. Menurut dokumentasi *Applio Project* (2024), *platform* ini dirancang untuk mendukung berbagai implementasi *Voice Conversion* berbasis RVC dengan proses konfigurasi yang lebih sederhana dan mudah digunakan.

Applio mengintegrasikan berbagai komponen pendukung yang dibutuhkan dalam proses *Voice Conversion*, seperti HuBERT sebagai *feature extractor*, *model Retrieval-Based Voice Conversion* (RVC), serta berbagai metode ekstraksi nada (*Fundamental Frequency/F0 Extraction*) seperti RMVPE, Harvest, dan Crepe. Integrasi tersebut memungkinkan proses pelatihan maupun inferensi dilakukan secara lebih efisien melalui satu *platform* yang terintegrasi.

Dalam proses pelatihan (*training*), *Applio* memanfaatkan dataset suara pengguna untuk membangun *model* yang mampu mengenali karakteristik vokal, seperti warna suara (*timbre*), intonasi, serta pola pelafalan. Setelah proses pelatihan selesai, sistem menghasilkan *model*

dengan format .pth dan berkas indeks .index yang selanjutnya digunakan pada tahap inferensi untuk melakukan konversi suara.

Pada tahap inferensi, *Applio* menerima file audio masukan beserta parameter *model* yang telah ditentukan, kemudian menghasilkan file audio baru dengan karakteristik suara sesuai *model* hasil pelatihan. Proses inferensi tersebut dapat dikendalikan melalui antarmuka aplikasi maupun melalui pemanggilan perintah menggunakan Python sehingga mudah diintegrasikan dengan sistem backend berbasis *Retrieval-Based Voice Conversion*.

Pada penelitian ini, *Applio* digunakan sebagai *platform* utama untuk menjalankan proses *Retrieval-Based Voice Conversion* (RVC). Backend *Retrieval-Based Voice Conversion* mengirimkan file audio hasil *Edge Text-to-Speech* (Edge TTS) beserta parameter *model*, seperti *model* (.pth), index (.index), pitch, index rate, dan metode ekstraksi F0, ke *Applio* menggunakan mekanisme subprocess. Selanjutnya *Applio* menjalankan proses inferensi dan menghasilkan file audio dalam format WAV yang kemudian dikirim kembali ke aplikasi *Flutter* sebagai hasil akhir konversi suara.

Penggunaan *Applio* pada penelitian ini memberikan beberapa keuntungan, antara lain kemudahan dalam proses pelatihan *model*, dukungan terhadap berbagai metode ekstraksi fitur suara, integrasi yang baik dengan Python, serta kemudahan dalam mengelola *model Retrieval-Based Voice Conversion* (RVC). Selain itu, *Applio* juga mendukung penggunaan *model* yang telah dilatih sebelumnya sehingga proses konversi suara dapat dilakukan secara lebih cepat tanpa harus melakukan pelatihan ulang setiap kali aplikasi digunakan.

Meskipun demikian, performa *Applio* sangat dipengaruhi oleh spesifikasi perangkat keras yang digunakan, terutama pada proses pelatihan *model*. Proses *training* memerlukan sumber daya komputasi yang tinggi, khususnya Graphics Processing Unit (GPU) berbasis NVIDIA yang mendukung teknologi CUDA. Pada penelitian ini, proses pelatihan dilakukan menggunakan perangkat dengan kemampuan komputasi yang terbatas sehingga jumlah *epoch* yang dicapai masih terbatas. Kondisi tersebut menyebabkan kualitas hasil konversi suara belum sepenuhnya mampu merepresentasikan logat dan cengkok khas Bahasa Sunda secara optimal, meskipun karakteristik dasar suara pengguna telah berhasil dipertahankan.

Dengan memanfaatkan *platform Applio*, aplikasi LingoNusantara AI mampu mengintegrasikan proses *Retrieval-Based Voice Conversion* (RVC) ke dalam sistem penerjemahan sehingga menghasilkan keluaran suara yang lebih personal, natural, dan sesuai dengan karakteristik vokal pengguna.

2.16 Mean Opinion Score (MOS)

Mean Opinion Score (MOS) merupakan metode evaluasi subjektif yang digunakan untuk mengukur kualitas suara berdasarkan penilaian langsung dari responden atau pendengar. Metode ini banyak digunakan pada bidang *Speech Processing*, *Text-to-Speech* (TTS), *Voice Conversion*, dan sistem komunikasi suara untuk mengetahui tingkat naturalitas, kejernihan, serta kenyamanan suara yang dihasilkan oleh suatu sistem. Menurut ITU-T Recommendation P.800 (1996), *Mean Opinion Score* merupakan standar internasional yang digunakan untuk mengevaluasi kualitas suara berdasarkan persepsi manusia.

Pada metode MOS, responden diminta untuk mendengarkan sampel suara, kemudian memberikan penilaian menggunakan skala tertentu. Nilai yang diperoleh dari seluruh responden selanjutnya dihitung rata-ratanya sehingga menghasilkan nilai *Mean Opinion Score* (MOS) yang merepresentasikan tingkat kualitas suara sistem secara keseluruhan. Menurut ITU-T P.800 (1996), skala penilaian MOS terdiri atas lima tingkat sebagaimana ditunjukkan pada Tabel 2.2.

Tabel 2. 2 Skala Penilaian Mean Opinion Score (MOS)

Nilai MOS	Kategori	Keterangan
5	Sangat Baik (Excellent)	Suara sangat alami, sangat jernih, dan hampir tidak dapat dibedakan dari suara asli.
4	Baik (Good)	Suara terdengar alami dengan sedikit gangguan yang tidak mengganggu pemahaman.
3	Cukup (Fair)	Suara masih dapat dipahami, tetapi terdapat gangguan yang mulai terasa.
2	Buruk (Poor)	Suara kurang jelas dan mulai mengganggu kenyamanan pendengar.
1	Sangat Buruk (Bad)	Suara sulit dipahami dan tidak layak digunakan.

Sumber: ITU-T Recommendation P.800 (1996).

Menurut Lo et al. (2020), metode MOS masih menjadi salah satu standar evaluasi utama pada penelitian di bidang *Speech Processing*, *Voice Conversion*, dan *Text-to-Speech* (TTS) karena mampu menggambarkan kualitas suara berdasarkan persepsi pengguna secara langsung.

Meskipun terdapat berbagai metode evaluasi objektif seperti PESQ (Perceptual Evaluation of Speech Quality) dan STOI (Short-Time Objective Intelligibility), pengujian subjektif menggunakan MOS tetap menjadi acuan utama dalam menilai tingkat naturalitas suara hasil sintesis.

Pada penelitian ini, metode MOS digunakan untuk mengevaluasi kualitas suara hasil *Retrieval-Based Voice Conversion* (RVC) yang dihasilkan oleh aplikasi LingoNusantara AI. Proses pengujian dilakukan dengan meminta sejumlah responden mendengarkan beberapa sampel suara hasil konversi yang dihasilkan sistem, kemudian memberikan penilaian terhadap kualitas suara berdasarkan beberapa aspek penilaian.

Aspek-aspek yang dinilai dalam penelitian ini meliputi:

1. Kejelasan suara (Speech Clarity), yaitu tingkat kemudahan responden dalam memahami suara hasil konversi.
2. Naturalitas suara (Naturalness), yaitu tingkat kemiripan suara hasil konversi dengan suara manusia secara alami.
3. Kemiripan identitas suara (Speaker Similarity), yaitu tingkat kesamaan karakteristik suara hasil konversi dengan suara asli pengguna.
4. Kenyamanan saat didengarkan (Listening Comfort), yaitu tingkat kenyamanan responden ketika mendengarkan hasil konversi suara.

Nilai yang diberikan oleh setiap responden kemudian dihitung menggunakan persamaan rata-rata sebagai berikut.

$$MOS = \frac{\sum_{i=1}^n x_i}{n}$$

Keterangan:

MOS = *Mean Opinion Score* (nilai rata-rata kualitas suara)

Σ = Simbol penjumlahan seluruh skor responden

x_i = Nilai yang diberikan oleh responden ke- i

n = Jumlah responden

Hasil perhitungan tersebut digunakan untuk menentukan tingkat kualitas suara yang dihasilkan oleh sistem *Retrieval-Based Voice Conversion* (RVC). Semakin tinggi nilai MOS yang diperoleh, maka semakin baik kualitas suara yang dihasilkan oleh aplikasi LingoNusantara AI. Oleh karena itu, metode MOS dipilih sebagai metode evaluasi utama dalam penelitian ini

karena mampu memberikan gambaran mengenai tingkat penerimaan pengguna terhadap kualitas suara hasil konversi yang dihasilkan sistem. Berdasarkan uraian tersebut, metode *Mean Opinion Score* (MOS) dipilih sebagai metode evaluasi subjektif pada penelitian ini karena mampu menggambarkan tingkat penerimaan pengguna terhadap kualitas suara hasil *Retrieval-Based Voice Conversion* (RVC). Pengujian menggunakan MOS dilakukan untuk mengetahui tingkat kejernihan suara, kemiripan dengan suara asli pengguna, naturalitas, serta kenyamanan hasil audio yang dihasilkan oleh aplikasi LingoNusantara AI. Hasil pengujian, proses perhitungan nilai MOS, serta analisis terhadap kualitas suara akan dibahas secara rinci pada Bab IV.

2.17 Black Box Testing

Black Box Testing merupakan salah satu metode pengujian perangkat lunak yang dilakukan dengan menguji fungsi-fungsi sistem berdasarkan masukan (*input*) dan keluaran (*output*) tanpa memperhatikan struktur kode program atau proses internal yang terjadi di dalam sistem. Menurut Pressman dan Maxim (2020), *Black Box Testing* bertujuan untuk memastikan bahwa setiap fungsi pada perangkat lunak berjalan sesuai dengan kebutuhan fungsional yang telah ditetapkan pada tahap analisis dan perancangan sistem.

Menurut Sommerville (2021), *Black Box Testing* berfokus pada perilaku sistem dari sudut pandang pengguna (*user perspective*). Pengujian dilakukan dengan memberikan berbagai variasi data masukan, kemudian membandingkan hasil keluaran yang diperoleh dengan hasil yang diharapkan. Apabila hasil keluaran sesuai dengan spesifikasi kebutuhan, maka fungsi tersebut dinyatakan berhasil.

Metode *Black Box Testing* banyak digunakan dalam pengembangan perangkat lunak karena mampu mengidentifikasi kesalahan pada fungsi sistem, validasi masukan, antarmuka pengguna, komunikasi antar modul, serta proses pengolahan data. Pengujian ini tidak memerlukan pemahaman terhadap struktur kode program sehingga dapat dilakukan berdasarkan dokumen kebutuhan sistem maupun skenario penggunaan aplikasi.

Menurut Pressman dan Maxim (2020), beberapa aspek yang umum diuji menggunakan metode *Black Box Testing* meliputi:

1. Validasi masukan (*input validation*).
2. Kesesuaian keluaran (*output validation*).
3. Fungsionalitas setiap fitur sistem.
4. Interaksi antar modul.

5. Penanganan kesalahan (*error handling*).
6. Respons sistem terhadap berbagai kondisi penggunaan.

Pada penelitian ini, metode *Black Box Testing* digunakan untuk menguji seluruh fitur utama pada aplikasi LingoNusantara AI. Pengujian dilakukan berdasarkan kebutuhan fungsional yang telah dirancang pada Bab III untuk memastikan bahwa setiap fitur dapat berjalan sesuai dengan tujuan penelitian.

Beberapa fitur yang diuji pada penelitian ini meliputi:

1. Registrasi pengguna (*Register*).
2. Login pengguna.
3. Proses penerjemahan Bahasa Indonesia ke Bahasa Sunda menggunakan *AI Translation Engine*.
4. Proses penerjemahan menggunakan *Google Gemini* melalui mekanisme *Hybrid AI Translation*.
5. Pemilihan ragam Bahasa Sunda (Loma, Lemes, dan Kasar).
6. Proses sintesis suara menggunakan *Edge Text-to-Speech* (Edge TTS).
7. Proses *Retrieval-Based Voice Conversion* (RVC) menggunakan *platform Applio*.
8. Pemutaran dan pengunduhan file audio hasil konversi.
9. Penyimpanan dan penampilan riwayat penerjemahan.
10. Pengelolaan profil pengguna dan pengaturan aplikasi.

Hasil pengujian setiap fitur kemudian dibandingkan dengan hasil yang diharapkan berdasarkan kebutuhan sistem. Apabila seluruh fungsi menghasilkan keluaran yang sesuai, maka fitur tersebut dinyatakan berhasil (Pass). Sebaliknya, apabila terdapat ketidaksesuaian antara hasil aktual dengan hasil yang diharapkan, maka fitur tersebut dinyatakan gagal (Fail) dan memerlukan perbaikan.

Pada penelitian ini, hasil *Black Box Testing* digunakan untuk mengevaluasi tingkat keberhasilan implementasi aplikasi LingoNusantara AI sebelum dilakukan pengujian kualitas suara menggunakan metode *Mean Opinion Score* (MOS). Dengan demikian, pengujian Black Box memastikan bahwa seluruh fungsi aplikasi telah berjalan sesuai dengan kebutuhan fungsional sebelum dilakukan evaluasi terhadap kualitas hasil konversi suara.

2.18 Metode Prototyping

Metode *Prototype* merupakan salah satu metode pengembangan perangkat lunak yang dilakukan melalui proses pembuatan model awal (*prototype*) sebagai gambaran sistem yang akan dikembangkan. *Prototype* digunakan sebagai media komunikasi antara pengembang dan pengguna sehingga kebutuhan sistem dapat dipahami dengan lebih baik sebelum sistem dikembangkan secara penuh. Menurut Pressman dan Maxim (2020), metode *Prototype* memungkinkan pengembang memperoleh umpan balik (*feedback*) dari pengguna sejak tahap awal sehingga perubahan kebutuhan dapat diakomodasi dengan lebih cepat.

Menurut Sommerville (2021), metode *Prototype* sangat sesuai digunakan pada pengembangan sistem yang memiliki kebutuhan fungsional yang masih dapat berkembang selama proses penelitian. Dengan adanya *prototype*, pengguna dapat mengevaluasi tampilan antarmuka, alur kerja sistem, maupun fungsi-fungsi utama sebelum dilakukan implementasi secara menyeluruh.

Secara umum, metode Prototyping terdiri atas beberapa tahapan, yaitu:

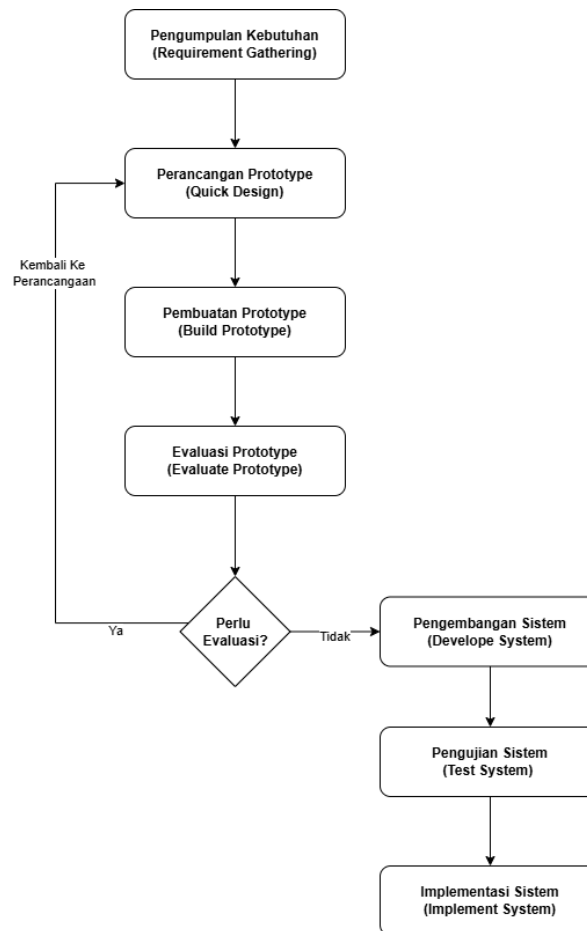
1. Pengumpulan Kebutuhan (Requirement Gathering)
Tahap awal dilakukan dengan mengidentifikasi kebutuhan sistem melalui studi literatur, observasi, serta analisis kebutuhan pengguna. Informasi yang diperoleh digunakan sebagai dasar dalam merancang fungsi-fungsi utama aplikasi.
2. Perancangan Prototype (Quick Design)
Berdasarkan kebutuhan yang telah diperoleh, dibuat rancangan awal sistem yang meliputi arsitektur aplikasi, perancangan antarmuka pengguna (*User Interface*), serta rancangan alur proses yang akan diimplementasikan.
3. Pembangunan Prototype (Build Prototype)
Prototype dikembangkan menggunakan teknologi yang telah dipilih, yaitu *Flutter Framework* sebagai *frontend* dan *Retrieval-Based Voice Conversion Framework* sebagai *backend*. Pada tahap ini mulai diimplementasikan fitur-fitur utama seperti autentikasi pengguna, *AI Translation Engine*, *Hybrid AI Translation*, *Google Gemini*, *Edge Text-to-Speech* (Edge TTS), serta *Retrieval-Based Voice Conversion* (RVC).
4. Evaluasi Prototype (User Evaluation)
Prototype yang telah dibuat diuji untuk mengetahui kesesuaian fungsi dengan kebutuhan pengguna. Hasil evaluasi digunakan sebagai dasar untuk melakukan perbaikan maupun penyempurnaan sistem.

5. Penyempurnaan dan Implementasi Sistem (Refinement and Implementation)

Setelah prototype dinyatakan sesuai dengan kebutuhan, sistem dikembangkan menjadi aplikasi yang lebih lengkap. Tahap ini meliputi penyempurnaan fitur, integrasi seluruh modul, pengujian menggunakan Black Box Testing, serta evaluasi kualitas suara menggunakan *Mean Opinion Score* (MOS).

Pada penelitian ini, metode Prototyping dipilih karena sesuai dengan proses pengembangan aplikasi LingoNusantara AI yang dilakukan secara bertahap. Selama proses penelitian, beberapa fitur mengalami penyempurnaan, seperti pengembangan *AI Translation Engine*, penambahan mekanisme *Hybrid AI Translation* menggunakan *Google Gemini*, integrasi *Edge Text-to-Speech* (Edge TTS), serta implementasi *Retrieval-Based Voice Conversion* (RVC). Pendekatan ini memungkinkan proses pengembangan berlangsung secara fleksibel sesuai dengan hasil evaluasi pada setiap tahap.

Dengan menggunakan metode Prototyping, proses pengembangan aplikasi menjadi lebih terarah karena setiap perubahan dapat diuji secara langsung sebelum diterapkan pada sistem secara keseluruhan. Selain itu, metode ini memudahkan proses evaluasi sehingga aplikasi yang dihasilkan lebih sesuai dengan kebutuhan pengguna dan tujuan penelitian.



Gambar 2. 1 Tahapan Metode Prototyping

Sumber: Pressman, R. S., & Maxim, B. R. (2020). Software Engineering: A Practitioner's Approach (9th Edition). McGraw-Hill Education.

2.1.11 Unified Modeling Language (UML)

Unified Modeling Language (UML) merupakan bahasa pemodelan standar yang digunakan dalam pengembangan perangkat lunak berorientasi objek untuk memvisualisasikan, menspesifikasikan, membangun, dan mendokumentasikan sistem. UML menyediakan sekumpulan diagram yang mampu merepresentasikan kebutuhan sistem, struktur, perilaku, serta interaksi antar komponen sehingga mempermudah proses komunikasi antara analis, perancang, dan pengembang perangkat lunak (Dennis, Wixom, & Tegarden, 2020).

Pada penelitian ini, UML digunakan untuk memodelkan arsitektur aplikasi LingoNusantara AI yang terdiri atas aplikasi *mobile* berbasis *Flutter*, backend berbasis *Retrieval-Based Voice Conversion*, *AI Translation Engine*, Dataset Builder, serta modul

Retrieval-Based Voice Conversion (RVC). Pemodelan dilakukan agar hubungan antar komponen sistem dapat dipahami secara jelas sebelum proses implementasi.

Diagram UML yang digunakan pada penelitian ini meliputi:

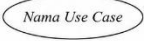
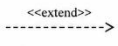
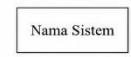
1. Use Case Diagram.
2. Activity Diagram.
3. Sequence Diagram.
4. Class Diagram.
5. Entity Relationship Diagram (ERD).

1. Use Case Diagram

Use Case Diagram merupakan salah satu diagram UML yang digunakan untuk menggambarkan interaksi antara aktor dengan sistem berdasarkan fungsi-fungsi yang tersedia. Menurut Booch, Rumbaugh, dan Jacobson (2005), Use Case Diagram berfungsi untuk menunjukkan layanan yang disediakan sistem kepada pengguna tanpa menjelaskan bagaimana proses tersebut diimplementasikan.

Pada penelitian ini, Use Case Diagram digunakan untuk memodelkan interaksi antara pengguna dengan aplikasi LingoNusantara AI, mulai dari proses memasukkan teks, memilih ragam Bahasa Sunda, merekam suara, mengirim data ke backend *Retrieval-Based Voice Conversion*, menerima hasil terjemahan, hingga memutar suara hasil *Retrieval-Based Voice Conversion (RVC)*.

Tabel 2. 3 Simbol Use Case Diagram

1		<i>Actor</i>	Pihak luar (pengguna atau sistem lain) yang berinteraksi langsung dengan sistem untuk memberikan input atau menerima output.
2		<i>Use Case</i>	Representasi dari fungsi layanan atau fitur utama yang disediakan oleh sistem untuk dijalankan oleh <i>actor</i> .
3		<i>Association</i>	Garis penghubung yang menunjukkan adanya interaksi langsung antara <i>actor</i> dengan <i>use case</i> .
4		<i>Include</i>	Hubungan yang menunjukkan bahwa suatu <i>use case</i> secara eksplisit menyertakan <i>use case</i> lain sebagai bagian dari prosesnya.
5		<i>Extend</i>	Hubungan yang menunjukkan bahwa suatu <i>use case</i> merupakan perluasan (opsional) terhadap <i>use case</i> lain jika kondisi tertentu terpenuhi.
6		<i>System Boundary</i>	Batasan sistem yang menunjukkan ruang lingkup sistem, memisahkan sistem dari <i>actor</i> yang berada di luar sistem.
7		<i>Note</i>	Elemen tambahan untuk memberikan catatan atau keterangan tertentu pada diagram.

Rosa, A. S., & Shalahuddin, M. (2018). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (Edisi Revisi). Bandung: Informatika.

2. Activity Diagram

Activity Diagram digunakan untuk menggambarkan alur aktivitas atau proses bisnis yang terjadi pada sistem. Diagram ini menunjukkan urutan aktivitas yang dilakukan pengguna maupun sistem mulai dari proses awal hingga proses selesai.

Pada penelitian ini, Activity Diagram digunakan untuk memodelkan proses penerjemahan Bahasa Indonesia ke Bahasa Sunda, komunikasi antara aplikasi *Flutter* dengan backend *Retrieval-Based Voice Conversion*, proses *AI Translation Engine*, pemanggilan model *Retrieval-Based Voice Conversion* (RVC), hingga pengiriman hasil audio kepada pengguna.

Tabel 2. 4 Simbol Activity Diagram

1		<i>Initial Node</i>	Titik awal dimulainya suatu aktivitas atau proses pada sistem.
2		<i>Action / Activity</i>	Menunjukkan suatu aktivitas atau proses yang dilakukan oleh sistem maupun pengguna.
3		<i>Decision Node</i>	Simbol pilihan percabangan alur proses yang bekerja berdasarkan pemenuhan kondisi atau parameter tertentu.
4		<i>Merge Node</i>	Simbol yang digunakan untuk menggabungkan kembali beberapa alur proses yang sebelumnya bercabang menjadi satu alur.
5		<i>Fork / Join Node</i>	Simbol berupa garis tebal untuk membagi satu aktivitas menjadi beberapa aktivitas paralel (<i>fork</i>) atau menggabungkan kembali aktivitas paralel (<i>join</i>).
6		<i>Activity Final Node</i>	Menandakan akhir dari seluruh rangkaian alur aktivitas operasional di dalam sistem.

Rosa, A. S., & Shalahuddin, M. (2018). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (Edisi Revisi). Bandung: Informatika.

3. Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan urutan pertukaran pesan antar objek berdasarkan waktu. Diagram ini menunjukkan bagaimana setiap komponen sistem saling berkomunikasi selama proses berlangsung.

Pada penelitian ini, Sequence Diagram digunakan untuk memodelkan komunikasi antara aplikasi *Flutter*, REST API *Retrieval-Based Voice Conversion*, *AI Translation Engine*, Dataset Builder, dan modul *Retrieval-Based Voice Conversion* (RVC) dalam menghasilkan terjemahan Bahasa Sunda beserta keluaran suara.

Tabel 2. 5 Simbol Sequence Diagram

1		<i>Actor / Object</i>	Mewakili peran <i>actor</i> atau objek yang berinteraksi dalam sistem.
2		<i>Lifeline</i>	Garis hidup objek yang menunjukkan keberadaan objek selama interaksi berlangsung.
3		<i>Activation / Execution Specification</i>	Periode waktu ketika objek melakukan suatu operasi atau aktivitas.
4		<i>Message (Call)</i>	Pesan atau permintaan operasi yang dikirim dari satu objek ke objek lainnya.
5		<i>Return Message</i>	Pesan balasan atau hasil dari operasi yang dikirim kembali ke objek pengirim.
6		<i>Create Message</i>	Pesan yang menunjukkan pembuatan objek baru.
7		<i>Destroy Message</i>	Pesan yang menunjukkan penghancuran atau akhir hidup suatu objek.

Rosa, A. S., & Shalahuddin, M. (2018). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (Edisi Revisi). Bandung: Informatika.

4. Class Diagram

Class Diagram digunakan untuk menggambarkan struktur statis sistem yang terdiri atas kelas, atribut, metode, serta hubungan antar kelas. Diagram ini menjadi acuan dalam proses implementasi perangkat lunak.

Pada penelitian ini, Class Diagram digunakan untuk memodelkan struktur program pada aplikasi LingoNusantara AI, meliputi kelas *AI Translation Engine*, *Dataset Builder*, *Retrieval-Based Voice Conversion API*, *Audio Service*, *Voice Model*, serta komponen pendukung lainnya yang digunakan dalam proses penerjemahan dan konversi suara.

Tabel 2. 6 Simbol Class Diagram

1	Nama Kelas - atribut + operasi()	Class	Representasi kelas dalam sistem yang memiliki atribut dan operasi (metode).
2	—————	Association	Hubungan struktural antara dua kelas.
3	—————>	Generalization (Inheritance)	Hubungan pewarisan (<i>inheritance</i>), di mana satu kelas merupakan turunan dari kelas lain.
4	----->	Realization	Hubungan realisasi, di mana kelas mengimplementasikan <i>interface</i> .
5	—————◇	Aggregation	Hubungan agregasi (keseluruhan-bagian) dengan relasi lemah.
6	—————◆	Composition	Hubungan komposisi (keseluruhan-bagian) dengan relasi kuat.

Rosa, A. S., & Shalahuddin, M. (2018). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (Edisi Revisi). Bandung: Informatika.

2.2 Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai landasan akademis untuk mengetahui perkembangan penelitian yang berkaitan dengan sistem penerjemahan bahasa, *Natural Language Processing* (NLP), *Machine Translation*, *Large Language Model* (LLM), *Voice Conversion*, serta pengembangan aplikasi berbasis *Artificial Intelligence* (AI). Selain itu, penelitian terdahulu berfungsi untuk mengidentifikasi kelebihan, kekurangan, dan celah penelitian (*research gap*) sehingga dapat mempertegas kebaruan (*novelty*) dari penelitian yang dilakukan.

Yang, Ma, Huang, dan Voyles (2020) mengembangkan penelitian mengenai HiFi-GAN, yaitu *model* Generative Adversarial Network (GAN) yang digunakan untuk menghasilkan sintesis suara berkualitas tinggi dengan waktu inferensi yang cepat. Hasil penelitian menunjukkan bahwa HiFi-GAN mampu menghasilkan suara yang lebih natural dibandingkan beberapa metode *vocoder* sebelumnya. Namun, penelitian tersebut hanya berfokus pada proses sintesis suara dan belum mengintegrasikan teknologi penerjemahan bahasa maupun personalisasi identitas suara.

Kong, Kim, dan Bae (2020) memperkenalkan arsitektur HiFi-GAN sebagai *neural vocoder* yang mampu menghasilkan kualitas audio mendekati suara manusia dengan efisiensi komputasi yang tinggi. Penelitian tersebut menjadi salah satu dasar perkembangan sistem *speech synthesis* modern, tetapi belum diterapkan pada sistem penerjemahan bahasa daerah maupun sistem yang memanfaatkan *Voice Conversion*.

Hsu, Bolte, Tsai, Lakhotia, Salakhutdinov, dan Mohamed (2021) mengembangkan *model* HuBERT (*Hidden Unit BERT*) sebagai metode *self-supervised learning* untuk memperoleh representasi suara. Hasil penelitian menunjukkan bahwa HuBERT mampu mengekstraksi informasi linguistik dari sinyal suara secara efektif sehingga banyak dimanfaatkan pada sistem Speech Recognition, *Speech Processing*, dan *Voice Conversion*. Meskipun demikian, penelitian tersebut belum membahas implementasinya pada aplikasi penerjemahan bahasa yang terintegrasi.

Russell dan Norvig (2021) menjelaskan bahwa perkembangan *Artificial Intelligence* telah mendorong munculnya berbagai aplikasi cerdas, seperti *Machine Translation*, *Speech Processing*, dan sistem berbasis *Machine Learning*. Konsep tersebut menjadi dasar dalam pengembangan sistem penerjemahan berbasis AI yang mampu memahami bahasa, melakukan penalaran, serta menghasilkan keluaran yang lebih kontekstual.

Jurafsky dan Martin (2023) menjelaskan bahwa perkembangan *Natural Language Processing* (NLP) memungkinkan komputer memahami struktur bahasa, hubungan semantik, serta konteks kalimat melalui pendekatan *Machine Learning*, *Deep Learning*, dan *Large Language Model* (LLM). Konsep tersebut menjadi dasar dalam pengembangan *AI Translation Engine* serta mekanisme *Hybrid AI Translation* yang diterapkan pada penelitian ini.

Susanto, Diandaru, Krisnadhi, Purwarianti, dan Wijaya (2023) melakukan penelitian mengenai *Neural Machine Translation* pada bahasa daerah di Indonesia. Hasil penelitian menunjukkan bahwa pengembangan sistem penerjemahan bahasa daerah masih menghadapi keterbatasan jumlah dataset sehingga kualitas hasil terjemahan sangat dipengaruhi oleh kelengkapan data pelatihan dan kemampuan *model* dalam memahami konteks bahasa.

Wang, Qian, Chen, dan rekan-rekan (2023) menjelaskan perkembangan teknologi *Retrieval-Based Voice Conversion* (RVC) yang memanfaatkan mekanisme *feature retrieval* untuk meningkatkan kemiripan identitas suara hasil konversi. Penelitian tersebut menunjukkan bahwa RVC mampu menghasilkan kualitas suara yang lebih natural dibandingkan metode *Voice Conversion* sebelumnya, namun belum diintegrasikan dengan sistem penerjemahan bahasa daerah berbasis aplikasi *mobile*.

Google Developers (2024) menjelaskan bahwa *Flutter* merupakan *framework* lintas *platform* yang mampu membangun aplikasi Android dan iOS menggunakan satu basis kode (*single codebase*). *Flutter* juga mendukung komunikasi dengan backend melalui REST API sehingga sesuai digunakan sebagai antarmuka aplikasi berbasis *Artificial Intelligence*.

Google AI (2024) memperkenalkan *Google Gemini* sebagai *Large Language Model* (LLM) yang mampu memahami konteks bahasa, menghasilkan teks, serta mendukung berbagai

tugas *Natural Language Processing*, termasuk penerjemahan bahasa. Gemini menyediakan layanan berbasis *Application Programming Interface* (API) sehingga dapat diintegrasikan dengan aplikasi untuk menghasilkan terjemahan yang lebih alami dan kontekstual.

Applio Team (2024) mengembangkan *platform Applio* sebagai antarmuka untuk proses pelatihan (*training*) dan inferensi (*inference*) *model Retrieval-Based Voice Conversion* (RVC). *Platform* tersebut mempermudah pengelolaan *model* suara, pembangunan berkas indeks, serta proses inferensi sehingga implementasi teknologi *Voice Conversion* menjadi lebih mudah diintegrasikan dengan aplikasi berbasis Python.

Berdasarkan berbagai penelitian tersebut dapat disimpulkan bahwa sebagian besar penelitian sebelumnya masih berfokus pada pengembangan *Machine Translation*, *Natural Language Processing*, *Speech Processing*, *Large Language Model*, maupun *Voice Conversion* secara terpisah. Belum banyak penelitian yang mengintegrasikan seluruh komponen tersebut ke dalam satu aplikasi *mobile* yang mampu melakukan penerjemahan Bahasa Indonesia ke Bahasa Sunda sekaligus menghasilkan keluaran suara dengan karakteristik vokal pengguna.

Oleh karena itu, penelitian ini menawarkan kebaruan (*novelty*) berupa pengembangan aplikasi LingoNusantara AI yang mengintegrasikan *AI Translation Engine* berbasis dataset lokal, *Google Gemini* melalui mekanisme *Hybrid AI Translation*, *Flutter* sebagai *frontend*, *Retrieval-Based Voice Conversion* REST API sebagai *backend*, *Edge Text-to-Speech* (Edge TTS), serta *Retrieval-Based Voice Conversion* (RVC) menggunakan *platform Applio*. Integrasi tersebut memungkinkan sistem menghasilkan terjemahan Bahasa Sunda yang tetap dapat digunakan secara *offline* melalui *AI Translation Engine* dan menghasilkan terjemahan yang lebih alami ketika *online* menggunakan *Google Gemini*. Selanjutnya, hasil terjemahan dikonversi menjadi suara yang mempertahankan karakteristik vokal pengguna melalui teknologi *Retrieval-Based Voice Conversion* (RVC).

2.3 Alasan Pemilihan Metode Prototyping

Metode Prototyping dipilih pada penelitian ini karena sesuai dengan karakteristik pengembangan aplikasi LingoNusantara AI yang memerlukan proses pengembangan secara bertahap serta penyempurnaan berulang berdasarkan hasil evaluasi. Menurut Pressman dan Maxim (2020), metode Prototyping sangat sesuai digunakan pada pengembangan perangkat lunak yang masih memerlukan penyesuaian kebutuhan selama proses implementasi sehingga pengembang dapat memperoleh umpan balik (*feedback*) secara berkelanjutan.

Selama proses penelitian, aplikasi mengalami beberapa perubahan dan penyempurnaan, baik pada sisi antarmuka (*User Interface*), arsitektur sistem, maupun fitur utama aplikasi. Pengembangan diawali dengan implementasi *AI Translation Engine* berbasis dataset lokal, kemudian dilanjutkan dengan integrasi *Google Gemini* melalui mekanisme *Hybrid AI Translation* untuk meningkatkan kualitas hasil terjemahan ketika perangkat terhubung ke internet. Selain itu, proses sintesis suara menggunakan *Edge Text-to-Speech* (Edge TTS) serta konversi suara menggunakan *Retrieval-Based Voice Conversion* (RVC) juga dikembangkan dan disempurnakan secara bertahap hingga menghasilkan sistem yang terintegrasi.

Metode Prototyping memberikan fleksibilitas dalam melakukan evaluasi terhadap setiap modul yang dikembangkan. Setiap perubahan dapat langsung diuji menggunakan *Black Box Testing* untuk memastikan seluruh fungsi sistem berjalan sesuai kebutuhan. Selanjutnya, kualitas suara hasil konversi dievaluasi menggunakan metode *Mean Opinion Score* (MOS) sehingga pengembangan sistem tidak hanya berfokus pada aspek fungsional, tetapi juga pada kualitas keluaran suara yang dihasilkan.

Dibandingkan dengan metode Waterfall, metode Prototyping lebih sesuai digunakan pada penelitian ini karena kebutuhan sistem mengalami perkembangan selama proses penelitian. Penambahan fitur *Hybrid AI Translation*, integrasi *Google Gemini*, penyempurnaan *AI Translation Engine*, serta implementasi *Retrieval-Based Voice Conversion* (RVC) merupakan beberapa perubahan yang dilakukan berdasarkan hasil evaluasi pada setiap tahapan pengembangan. Pendekatan tersebut memungkinkan sistem berkembang secara lebih fleksibel tanpa harus mengulang seluruh proses pengembangan dari awal.

Dengan demikian, metode Prototyping dinilai paling sesuai untuk digunakan dalam penelitian ini karena mampu mendukung proses pengembangan aplikasi secara bertahap, mempermudah proses evaluasi dan penyempurnaan sistem, serta menghasilkan aplikasi LingoNusantara AI yang sesuai dengan kebutuhan pengguna dan tujuan penelitian.

Perbandingan metode tersebut disajikan pada Tabel 2.3.

Tabel 2. 7 Perbandingan Metode Pengembangan Perangkat Lunak

Metode	Kelebihan	Kekurangan	Alasan Tidak Dipilih / Dipilih
Waterfall	Tahapan pengembangan terstruktur dan dokumentasi lengkap.	Sulit melakukan perubahan apabila kebutuhan sistem berubah di tengah proses pengembangan.	Tidak dipilih karena pengembangan aplikasi LingoNusantara AI masih memerlukan banyak revisi dan penyempurnaan selama proses penelitian.
Agile	Fleksibel terhadap perubahan kebutuhan dan mendukung pengembangan bertahap.	Membutuhkan tim pengembang yang aktif berkolaborasi serta pembagian sprint secara berkelanjutan.	Tidak dipilih karena penelitian dilakukan secara individu sehingga penerapan Agile kurang efektif.
Spiral	Memiliki analisis risiko yang baik dan cocok untuk proyek berskala besar.	Proses pengembangan lebih kompleks, membutuhkan waktu dan biaya yang lebih besar.	Tidak dipilih karena ruang lingkup penelitian tidak memerlukan analisis risiko yang kompleks.
Prototyping	Memungkinkan pembuatan prototype yang dapat dievaluasi dan disempurnakan secara bertahap berdasarkan masukan pengguna.	Memerlukan beberapa kali iterasi hingga diperoleh sistem akhir yang sesuai kebutuhan.	Dipilih karena paling sesuai dengan karakteristik penelitian yang memerlukan pengembangan bertahap, integrasi AI, serta evaluasi berulang pada setiap proses pengembangan.

Sumber: Diadaptasi dari Pressman dan Maxim (2020); Sommerville (2021); Dennis, Wixom, & Tegarden (2020).

Berdasarkan perbandingan pada Tabel 2.3, metode Prototyping dipilih karena memiliki karakteristik yang paling sesuai dengan penelitian ini. Pengembangan aplikasi LingoNusantara AI memerlukan proses evaluasi dan penyempurnaan secara berulang, baik pada modul penerjemahan, proses konversi suara, maupun antarmuka aplikasi. Karakteristik tersebut sulit

dicapai apabila menggunakan metode Waterfall yang bersifat linear, sedangkan metode Agile dan Spiral dinilai kurang sesuai dengan ruang lingkup penelitian yang dilakukan secara individual.