Penerbit : PT Dewangga Energi Internasional
Anggota IKAPI (403/JBA/2021)
Komp. Purigading Ruko I No. 39
Pondokmelati Kota Bekasi
Tlp. 0851-6138-9537
www.dewanggapublishing.com

ISBN 978-634-233-068-5

DEEP LEARNING

BERBASIS SUARA & TEKS

Buku Referensi

DEEP LEARNING

Dr. Kristiawan Nugroho, M.Kom

DEEP LEARNING

BERBASIS SUARA & TEKS

Buku Referensi

Penulis:

Dr. Kristiawan Nugroho, M.Kom

Penerbit: PT Dewangga Energi Internasional

DEEP LEARNING BERBASIS SUARA & TEKS

Buku Referensi

Penulis:
Dr. Kristiawan Nugroho, M.Kom

ISBN: 978-634-233-068-5

Editor:
Aly Rasyid

Desain Cover & Tata Letak:
DewanggaPublishing

Proofreader:
Aly Rasyid

Penerbit:
PT Dewangga Energi Internasional
Anggota IKAPI (403/JBA/2021)

Redaksi:
Komp. Purigading Ruko I No. 39
Pondokmelati Kota Bekasi 17414
Telp/WA: 0851-6138-9537

E-mail: dewanggapublishing@gmail.com
Website: www.dewanggapublishing.com

Cetakan Pertama: Agustus 2025
Ukuran: 89 halaman, B5 18,2 x 25,7 cm

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa yang telah melimpahkan rahmat, hidayah, dan karunia-Nya sehingga buku referensi dengan judul **"RISET DEEP LEARNING BERBASIS SUARA & TEKS"** ini dapat tersusun dan diselesaikan dengan baik. Buku ini hadir sebagai salah satu bentuk kontribusi dalam bidang kecerdasan buatan *(Artificial Intelligence)*, khususnya *Deep Learning*, yang saat ini berkembang sangat pesat dan memiliki peran penting dalam pemrosesan data multimodal, terutama suara dan teks. Kajian mengenai pengolahan suara (speech processing) dan pemrosesan bahasa alami *(Natural Language Processing)* menjadi fondasi dalam berbagai aplikasi modern, mulai dari asisten virtual, sistem rekomendasi, hingga analisis data big data berbasis interaksi manusia.

Melalui buku ini, penulis berupaya memberikan gambaran yang komprehensif mengenai konsep, metode, serta implementasi riset *Deep Learning* yang dapat diaplikasikan dalam pengolahan suara dan teks. Diharapkan, buku ini tidak hanya menjadi referensi akademik, tetapi juga dapat menginspirasi para peneliti, mahasiswa, dan praktisi teknologi informasi untuk mengembangkan inovasi baru di bidang ini. Ucapan terima kasih penulis sampaikan kepada semua pihak yang telah memberikan dukungan, baik berupa moral, akademik, maupun teknis, sehingga penyusunan buku ini dapat berjalan dengan lancar. Kritik dan saran yang membangun dari pembaca sangat diharapkan demi perbaikan dan penyempurnaan buku ini di masa yang akan datang.

Akhir kata, semoga buku ini dapat memberikan manfaat yang sebesar-besarnya, menjadi sumber ilmu, dan memberikan kontribusi nyata bagi pengembangan ilmu pengetahuan, khususnya di bidang riset *Deep Learning berbasis suara dan teks*.

Semarang, 22 Agustus 2025

Penulis

DAFTAR ISI

BAB 1

PENDAHULUAN

1.1. Latar Belakang

Riset deep learning berbasis suara dan teks merupakan bidang multidisipliner yang memadukan pemrosesan bahasa alami *(Natural Language Processing/NLP)* dan pemrosesan sinyal audio untuk menciptakan sistem cerdas yang mampu memahami, mengekstraksi makna, serta merespons komunikasi manusia secara alami. Dalam beberapa tahun terakhir, kemajuan teknologi deep learning, khususnya arsitektur seperti *Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Transformer*, dan *Convolutional Neural Network (CNN),* telah merevolusi cara mesin memproses data suara dan teks. Riset ini sangat relevan dalam konteks peningkatan interaksi manusia dan mesin, mulai dari pengenalan suara (speech recognition), konversi teks ke suara (text-to-speech), penerjemahan otomatis, chatbot berbasis suara, hingga deteksi emosi dan gangguan psikologis melalui analisis suara dan teks.

Pada aspek suara, riset deep learning memungkinkan identifikasi pola-pola frekuensi, intonasi, ritme bicara, dan spektrum suara yang mengandung informasi semantik dan afektif. Misalnya, deteksi depresi atau stres dapat dilakukan dengan mengidentifikasi ciri-ciri prosodik dan artikulasi yang tidak biasa menggunakan model CNN atau LSTM. Di sisi lain, teks sebagai representasi simbolik bahasa manusia menjadi sumber data penting dalam memahami konteks, sintaksis, dan makna. Dengan teknik seperti word embedding (Word2Vec, GloVe, BERT) dan arsitektur Transformer, model deep learning mampu mengolah konteks kalimat secara menyeluruh dan melakukan klasifikasi, summarization, hingga question answering secara presisi tinggi.

Integrasi suara dan teks dalam satu kerangka riset membuka jalan bagi aplikasi yang lebih kompleks dan adaptif, seperti asisten virtual, sistem pengawasan kesehatan mental, serta pengajaran bahasa berbasis AI. Penelitian ini juga mencakup tantangan penting seperti kebutuhan data besar, labeling berkualitas tinggi, interpretabilitas model, serta bias algoritma. Oleh karena itu, pengembangan model deep learning berbasis suara dan teks tidak hanya memerlukan pendekatan komputasi yang canggih, tetapi juga pemahaman linguistik, psikologi, dan etika AI. Melalui pendekatan lintas disiplin ini, riset deep

learning berbasis suara dan teks terus berkembang menjadi fondasi penting dalam menciptakan teknologi yang lebih humanistik, inklusif, dan bermanfaat luas bagi masyarakat.

1.2. Tujuan dan Ruang Lingkup Buku

Buku ini bertujuan untuk memberikan pemahaman yang mendalam mengenai riset dalam bidang deep learning yang memanfaatkan suara dan teks sebagai data input untuk berbagai aplikasi teknologi modern. Deep learning, sebagai salah satu cabang dari machine learning, telah menunjukkan perkembangan pesat dalam beberapa tahun terakhir, terutama dalam aplikasi-aplikasi yang melibatkan analisis suara dan teks. Melalui buku ini, pembaca diharapkan dapat memahami berbagai teknik dasar dan lanjutan dalam deep learning, termasuk arsitektur jaringan saraf tiruan (ANN), jaringan saraf konvolusional (CNN), serta jaringan saraf berulang (RNN) dan Long Short-Term Memory (LSTM), yang sering diterapkan dalam tugas-tugas pengolahan suara dan teks. Buku ini juga bertujuan untuk menggali pemanfaatan model-model tersebut dalam berbagai aplikasi, seperti pengenalan suara, pemrosesan bahasa alami (NLP), serta analisis emosi atau sentimen melalui suara dan teks. Selain itu, buku ini mengulas penerapan deep learning dalam teknologi pengenalan ucapan, termasuk pengubahan suara menjadi teks, dan sebaliknya, serta dalam sistem-sistem yang memanfaatkan interaksi manusia-komputer berbasis suara. Di samping itu, buku ini akan menyajikan tantangan yang dihadapi dalam penelitian suara dan teks, seperti pemrosesan data tidak terstruktur, pengelolaan variasi dalam data suara, dan konteks linguistik dalam teks. Ruang lingkup buku ini mencakup beberapa topik utama yang meliputi pemahaman dasar deep learning, teknik-teknik pengolahan suara dan teks, serta penerapan dan evaluasi model-model deep learning dalam konteks suara dan teks. Pembaca juga akan diajak untuk mendalami cara-cara melakukan eksperimen dan penelitian dalam topik ini dengan menggunakan berbagai dataset publik, serta teknik evaluasi kinerja model yang relevan. Selain itu, buku ini juga membahas berbagai tren terkini dalam riset, termasuk penerapan model-model terbaru seperti Transformer, BERT, GPT, serta penerapan deep learning dalam pengenalan suara dan analisis sentimen. Buku ini ditujukan untuk pembaca yang memiliki pemahaman dasar tentang machine learning dan deep learning, serta mereka yang tertarik untuk memperluas wawasan mereka dalam bidang pemrosesan suara dan teks. Dengan demikian, buku ini tidak hanya memberikan teori dasar, tetapi juga menawarkan pendekatan praktis untuk pengembangan dan penerapan algoritma deep learning yang dapat diterapkan dalam riset ilmiah maupun

industri, seperti dalam bidang kesehatan, pendidikan, layanan pelanggan, dan sistem kecerdasan buatan (AI) lainnya yang mengandalkan suara dan teks sebagai input. Melalui buku ini, pembaca diharapkan dapat memperoleh wawasan yang mendalam tentang potensi dan tantangan yang ada dalam riset deep learning berbasis suara dan teks, serta bagaimana teknologi ini dapat diterapkan untuk menciptakan solusi inovatif yang bermanfaat bagi kehidupan sehari-hari.

1.3. Manfaat Buku bagi Peneliti dan Praktisi

Buku *"Riset Deep Learning Berbasis Suara & Teks"* merupakan kontribusi penting dalam pengembangan ilmu pengetahuan dan teknologi, khususnya dalam bidang kecerdasan buatan (Artificial Intelligence/AI), pembelajaran mesin (Machine Learning), dan pengolahan data multimodal. Buku ini tidak hanya memberikan landasan teoritis yang kuat, tetapi juga menyajikan penerapan praktis dan studi kasus yang relevan, sehingga sangat bermanfaat baik bagi kalangan akademisi maupun praktisi industri.

1. Manfaat bagi Peneliti

Bagi para peneliti, buku ini menjadi referensi yang kaya akan teori mutakhir dan pendekatan metodologis dalam domain *deep learning* yang berfokus pada dua jenis data utama: data suara (audio/speech) dan data teks (textual data). Kedua jenis data ini menjadi perhatian utama dalam riset-riset terbaru, terutama dalam pengembangan sistem yang bersifat interaktif seperti asisten virtual, chatbot, sistem deteksi emosi, dan pengenal suara otomatis.

Beberapa manfaat khusus bagi peneliti antara lain:

a. Referensi Ilmiah dan Teoritis: Buku ini menyajikan landasan teoretis mengenai arsitektur deep learning seperti RNN, LSTM, GRU, Transformer, dan model-model terbaru seperti BERT serta Whisper. Hal ini memungkinkan peneliti untuk mendapatkan pemahaman mendalam sebelum mengembangkan riset lanjutan.

b. Inspirasi Topik Penelitian: Dengan membahas tren riset terkini dalam pengolahan suara dan teks, buku ini membantu peneliti menemukan celah penelitian (research gap) serta memformulasikan pertanyaan riset yang relevan.

c. Panduan Eksperimen dan Replikasi: Buku ini menyertakan contoh eksperimen, pipeline data, preprocessing, dan hasil pengujian yang bisa direplikasi atau

dikembangkan untuk penelitian lanjutan, sehingga mendukung prinsip keterulangan (*reproducibility*) dalam riset ilmiah.

d. Interdisipliner: Riset yang berbasis suara dan teks sangat relevan untuk kolaborasi antar bidang seperti linguistik, psikologi, ilmu komputer, dan ilmu data. Buku ini menjadi titik temu antar disiplin untuk membangun riset kolaboratif.

2. Manfaat bagi Praktisi

Bagi praktisi di industri teknologi, pendidikan, layanan pelanggan, kesehatan, dan keamanan siber, buku ini memberikan pemahaman aplikatif terhadap bagaimana teknologi *deep learning* berbasis suara dan teks dapat dimanfaatkan untuk membangun solusi nyata di lapangan.

Beberapa manfaat langsung yang dapat dirasakan oleh praktisi antara lain:

a. Penerapan dalam Dunia Industri: Buku ini menjelaskan bagaimana sistem seperti voice recognition, speech-to-text, text-to-speech, dan analisis sentimen dapat dibangun dan diterapkan dalam sistem otomatisasi, chatbot, atau sistem monitoring kualitas layanan pelanggan.

b. Panduan Pengembangan Produk: Dengan adanya studi kasus nyata serta pembahasan teknis implementasi (misalnya menggunakan Python, TensorFlow, PyTorch), buku ini menjadi sumber panduan teknis bagi praktisi yang sedang mengembangkan produk berbasis suara dan teks.

c. Efisiensi dan Inovasi Layanan: Dalam sektor bisnis, pemanfaatan AI berbasis suara dan teks dapat meningkatkan efisiensi operasional. Buku ini memberikan wawasan bagaimana solusi berbasis AI dapat digunakan untuk otomatisasi layanan call center, asisten virtual berbasis suara, serta klasifikasi dokumen teks secara otomatis.

d. Peningkatan Kompetensi SDM: Buku ini juga dapat dimanfaatkan sebagai bahan ajar atau pelatihan untuk peningkatan kapasitas sumber daya manusia yang bekerja di bidang data science dan machine learning.

3. Kontribusi terhadap Ekosistem Teknologi

Selain manfaat langsung bagi individu peneliti dan praktisi, buku ini turut memberikan kontribusi terhadap penguatan ekosistem teknologi di Indonesia dan global. Buku ini mendorong:

a. Adopsi Teknologi AI secara Luas

b. Literasi Data dan AI di Kalangan Profesional

c. Pemanfaatan Teknologi untuk Inklusi Sosial (misalnya dalam aplikasi text-to-speech untuk disabilitas)

BAB 2

DETEKSI SUARA MENGGUNAKAN DEEP LEARNING

SINGULAR VALUE DECOMPOSITION

2.1. Deteksi Suara

Deteksi suara merupakan proses identifikasi dan pemrosesan suara yang dihasilkan dari sumber tertentu menggunakan teknologi komputer dan perangkat keras yang mendukung. Teknologi ini berfokus pada pengenalan, analisis, dan klasifikasi suara berdasarkan pola akustik dan fitur suara lainnya. Pada berbagai aplikasi, deteksi suara berperan penting, mulai dari pengenalan suara manusia *(speech recognition),* identifikasi objek melalui suara (sound recognition), hingga pengembangan sistem-sistem cerdas berbasis suara. Deteksi suara dapat diimplementasikan dalam berbagai sistem, termasuk dalam bidang keamanan, kesehatan, otomasi rumah, dan industri hiburan. Kemajuan dalam kecerdasan buatan (AI), terutama dalam deep learning dan machine learning, telah meningkatkan kemampuan sistem deteksi suara untuk mengenali pola suara dengan tingkat akurasi yang semakin tinggi. Artikel ini bertujuan untuk menjelaskan berbagai aspek terkait deteksi suara, termasuk teknik yang digunakan, aplikasi, serta tantangan yang dihadapi dalam pengembangan teknologi ini.

Teknik dalam Deteksi Suara

Teknik deteksi suara dapat dibagi menjadi beberapa kategori berdasarkan tujuan dan proses yang dilakukan dalam pengolahan suara. Beberapa teknik umum yang digunakan dalam deteksi suara antara lain:

1. Pengenalan Suara *(Speech Recognition)*

   Teknologi pengenalan suara bertujuan untuk mengonversi suara manusia menjadi teks yang dapat dipahami oleh komputer. Sistem pengenalan suara bekerja dengan menganalisis bentuk gelombang suara, mengidentifikasi frekuensi dan amplitudo, serta mencocokkan pola suara dengan model bahasa yang telah dipelajari. Sistem ini sering digunakan dalam aplikasi seperti asisten virtual (misalnya, Siri atau Google Assistant), transkripsi otomatis, dan perangkat berbasis suara lainnya.

2. Klasifikasi Suara *(Sound Classification)*

   Klasifikasi suara adalah teknik yang digunakan untuk mengidentifikasi kategori atau jenis suara berdasarkan pola akustiknya. Misalnya, sistem ini dapat mengenali suara-suara seperti gemericik air, suara pintu yang terbuka, atau suara mesin yang berjalan. Teknologi ini digunakan dalam berbagai aplikasi, seperti deteksi suara di lingkungan industri untuk pemeliharaan prediktif, serta dalam sistem keamanan untuk mendeteksi suara pecahan kaca atau alarm.

3. Deteksi Peristiwa Berdasarkan Suara *(Event Detection)*

   Deteksi peristiwa berbasis suara berfokus pada identifikasi kejadian tertentu yang dihasilkan oleh suara. Contoh penggunaan teknologi ini adalah dalam sistem pemantauan kebakaran, di mana suara alarm kebakaran dapat dideteksi untuk memicu respons otomatis. Deteksi ini juga digunakan dalam aplikasi kesehatan untuk mendeteksi perubahan dalam pola napas atau suara batuk pada pasien.

4. Pemrosesan Suara Berbasis Frekuensi *(Frequency-Based Sound Processing)*

   Pemrosesan suara berbasis frekuensi menganalisis suara berdasarkan distribusi frekuensi dan energi suara pada rentang waktu tertentu. Salah satu teknik yang digunakan dalam pemrosesan ini adalah Transformasi Fourier Cepstral (MFCC) yang sering digunakan untuk ekstraksi fitur dalam pengenalan suara. MFCC membantu dalam mengidentifikasi karakteristik suara manusia atau suara alami lainnya.

Penerapan Teknologi Deteksi Suara

Deteksi suara memiliki aplikasi yang luas di berbagai sektor, di antaranya:

1. Keamanan dan Pengawasan

   Pada sistem pengawasan, teknologi deteksi suara digunakan untuk mendeteksi peristiwa atau aktivitas yang mencurigakan. Misalnya, sistem ini dapat digunakan untuk mendeteksi suara tembakan atau perkelahian di area publik atau gedung, yang kemudian akan mengirimkan peringatan ke pihak keamanan.

2. Asisten Virtual dan Interaksi Manusia-Komputer

   Asisten virtual seperti Amazon Alexa, Google Assistant, dan Apple Siri menggunakan deteksi suara untuk memberikan layanan berbasis suara. Sistem ini mengandalkan pengenalan suara untuk memproses perintah suara pengguna, memungkinkan interaksi lebih alami dengan perangkat teknologi.

3. Aplikasi Kesehatan

   Teknologi deteksi suara juga diterapkan dalam bidang kesehatan, terutama untuk memantau kondisi pasien. Misalnya, deteksi suara batuk atau napas dapat memberikan informasi tentang kondisi pernapasan pasien, yang berguna dalam pemantauan jarak jauh. Beberapa sistem juga dapat mendeteksi suara jantung untuk mendiagnosis kelainan jantung.

4. Industri dan Otomasi Rumah

   Sistem deteksi suara digunakan dalam sektor industri untuk memonitor kondisi mesin dan peralatan. Teknologi ini dapat mendeteksi suara anomali yang mungkin menunjukkan kerusakan pada mesin atau peralatan. Selain itu, di rumah pintar (smart home), teknologi deteksi suara digunakan untuk mengaktifkan berbagai perangkat, seperti lampu, pengatur suhu, atau sistem keamanan berdasarkan suara pengguna.

Tantangan dalam Deteksi Suara

Meskipun deteksi suara memiliki banyak manfaat, ada sejumlah tantangan yang perlu diatasi untuk meningkatkan akurasi dan keandalan teknologi ini. Beberapa tantangan utama meliputi:

1. Lingkungan yang Berisik

   Suara latar belakang atau noise dapat mempengaruhi akurasi sistem deteksi suara, terutama di lingkungan yang bising. Untuk mengatasi masalah ini, teknologi noise cancellation dan filter suara digunakan untuk meningkatkan kemampuan sistem dalam mengenali suara yang relevan.
2. Variasi dalam Suara

   Suara manusia atau suara alam sangat bervariasi, tergantung pada faktor-faktor seperti aksen, usia, jenis kelamin, atau bahkan kondisi fisik individu. Hal ini menyulitkan sistem deteksi suara untuk mengenali semua variasi dengan akurat. Oleh karena itu, dibutuhkan model yang lebih fleksibel dan adaptif untuk mengenali berbagai variasi suara.
3. Pemrosesan Data dalam Waktu Nyata

   Pada aplikasi-aplikasi yang memerlukan respons cepat (seperti deteksi peristiwa), sistem deteksi suara harus mampu memproses data dalam waktu nyata. Mengelola pemrosesan data yang sangat besar dengan kecepatan tinggi menjadi tantangan besar, terutama ketika menggunakan metode yang berbasis AI dan machine learning.

2.2. Deep Learning

Deep learning merupakan cabang dari pembelajaran mesin (machine learning) yang menggunakan struktur jaringan saraf tiruan (artificial neural networks) untuk menganalisis dan memodelkan data yang sangat besar dan kompleks. Teknik ini terinspirasi oleh cara kerja otak manusia dalam memproses informasi dan mengambil keputusan. Metode deep learning memiliki kemampuan untuk secara otomatis mempelajari representasi data dari berbagai lapisan (layers), memungkinkan model untuk mengenali pola-pola yang tidak terlihat oleh metode konvensional.

Prinsip Kerja Deep Learning

Pada dasarnya, deep learning melibatkan penggunaan jaringan saraf yang terdiri dari beberapa lapisan (layers). Setiap lapisan berfungsi untuk mengubah data input menjadi representasi yang semakin kompleks, dengan lapisan terakhir memberikan output yang merupakan hasil dari pemrosesan data.

Jaringan saraf dalam deep learning dapat dibagi menjadi tiga bagian utama:

1. Lapisan Input (Input Layer): Menerima data mentah yang akan diproses. Data ini bisa berupa gambar, teks, suara, atau jenis data lainnya.
2. Lapisan Tersembunyi (Hidden Layers): Merupakan lapisan-lapisan yang ada di antara input dan output. Lapisan-lapisan ini melakukan transformasi kompleks terhadap data dengan memanfaatkan bobot (weights) yang dipelajari selama proses pelatihan.

3. Lapisan Output (Output Layer): Menghasilkan hasil akhir yang sesuai dengan tujuan dari model, seperti klasifikasi, prediksi, atau pengenalan objek.

Jenis-Jenis Arsitektur Deep Learning

Terdapat beberapa jenis arsitektur deep learning yang digunakan sesuai dengan jenis data dan aplikasi yang dihadapi, antara lain:

1. Jaringan Saraf Feedforward (Feedforward Neural Networks - FNN)
   Merupakan jenis jaringan saraf yang paling dasar, di mana data mengalir dalam satu arah, dari input hingga output, tanpa adanya umpan balik (feedback).
2. Jaringan Saraf Konvolusional (Convolutional Neural Networks - CNN)
   CNN sering digunakan untuk pengolahan gambar dan video. Mereka menggunakan lapisan konvolusi untuk mengekstraksi fitur dari data gambar, diikuti oleh lapisan pooling untuk mereduksi dimensi, dan akhirnya lapisan fully connected untuk menghasilkan prediksi.
3. Jaringan Saraf Berulang (Recurrent Neural Networks - RNN)
   RNN digunakan untuk data yang bersifat urutan, seperti teks atau data waktu. Mereka memiliki umpan balik, yang memungkinkan informasi untuk dipertahankan dan digunakan dalam waktu yang lebih lama, cocok untuk aplikasi seperti analisis teks dan prediksi urutan.
4. Long Short-Term Memory (LSTM)
   LSTM adalah jenis RNN yang dirancang untuk mengatasi masalah vanishing gradient dalam jaringan saraf berulang. LSTM dapat mempelajari ketergantungan jangka panjang dalam data sekuensial, sangat berguna dalam analisis teks atau prediksi yang memerlukan ingatan jangka panjang.
5. Generative Adversarial Networks (GAN)
   GAN terdiri dari dua model jaringan yang bersaing satu sama lain: generator yang mencoba membuat data yang realistis, dan discriminator yang mencoba membedakan data asli dari yang dihasilkan. GAN sering digunakan dalam pembuatan gambar atau video yang realistis.

Pelatihan model deep learning melibatkan beberapa langkah kunci:

1. Pengumpulan Data: Data yang relevan dan berkualitas tinggi sangat penting untuk pelatihan model deep learning. Data harus cukup besar dan beragam untuk mencakup berbagai variasi yang dapat ditemukan dalam dunia nyata.
2. Preprocessing Data: Sebelum dimasukkan ke dalam model, data perlu diproses terlebih dahulu. Untuk gambar, ini bisa melibatkan normalisasi atau augmentasi gambar. Untuk teks, ini bisa melibatkan tokenisasi dan penghapusan kata-kata tidak penting.
3. Pembagian Data: Data umumnya dibagi menjadi tiga set: data pelatihan (training), data validasi (validation), dan data pengujian (test). Data pelatihan digunakan untuk melatih

model, data validasi digunakan untuk memilih parameter terbaik, dan data pengujian digunakan untuk mengevaluasi kinerja model setelah pelatihan.

4. Pelatihan: Model deep learning dilatih dengan menggunakan algoritma optimasi seperti stochastic gradient descent (SGD) atau Adam, yang berusaha meminimalkan fungsi kerugian (loss function). Bobot-bobot dalam jaringan saraf diperbarui berdasarkan hasil error (selisih antara prediksi dan nilai sebenarnya).
5. Evaluasi Model: Setelah pelatihan selesai, model dievaluasi menggunakan data pengujian. Kinerja model dapat diukur menggunakan berbagai metrik seperti akurasi, presisi, recall, dan F1-score.

Tantangan dalam Deep Learning

Walaupun deep learning telah membawa kemajuan besar dalam berbagai aplikasi, terdapat beberapa tantangan yang perlu diperhatikan, di antaranya:

1) Kebutuhan Data Besar: Deep learning membutuhkan jumlah data yang sangat besar untuk pelatihan. Tanpa data yang cukup, model tidak akan dapat belajar dengan baik.
2) Kebutuhan Komputasi Tinggi: Pelatihan model deep learning memerlukan perangkat keras yang kuat, seperti GPU atau TPU, yang dapat meningkatkan biaya dan waktu pelatihan.
3) Interpretabilitas Model: Model deep learning sering dianggap sebagai "kotak hitam" karena kesulitan dalam menjelaskan bagaimana keputusan dibuat. Ini menjadi tantangan besar dalam aplikasi yang memerlukan transparansi, seperti di bidang medis atau hukum.

Aplikasi Deep Learning Deep learning telah diaplikasikan dalam berbagai bidang, antara lain:

1) Pengenalan Wajah dan Objek: CNN banyak digunakan dalam aplikasi pengenalan wajah, deteksi objek dalam gambar, dan pengolahan video.
2) Pengenalan Suara dan Teks: RNN dan LSTM digunakan dalam pengenalan suara, seperti di asisten virtual (contohnya, Google Assistant dan Siri), serta dalam analisis sentimen atau terjemahan otomatis.
3) Mobil Otonom: Deep learning digunakan untuk mengendalikan mobil tanpa pengemudi dengan mengenali objek di sekitar mobil dan mengambil keputusan berdasarkan pengolahan data visual dan sensor lainnya.
4) Medis: Dalam bidang medis, deep learning digunakan untuk diagnosis gambar medis, prediksi penyakit, dan perawatan yang dipersonalisasi.

2.3 Singular Value Decomposition

Singular Value Decomposition (SVD) adalah teknik aljabar linier yang sangat berguna dalam berbagai aplikasi ilmiah dan teknik, termasuk dalam deteksi suara. Metode ini menguraikan matriks menjadi tiga matriks terpisah, yang memiliki sifat penting dan digunakan untuk ekstraksi informasi dari data, termasuk data suara. SVD memungkinkan representasi data dalam bentuk yang lebih sederhana dan komprehensif, yang memudahkan deteksi pola atau fitur penting dalam data suara.

1. Pengertian Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) adalah suatu metode dalam aljabar linier yang digunakan untuk memfaktorkan sebuah matriks menjadi tiga matriks lain. Dengan kata lain, SVD memecah sebuah matriks besar menjadi bentuk yang lebih sederhana namun tetap menyimpan informasi penting di dalamnya.

ecara matematis, jika ada matriks AAA berukuran m×nm \times nm×n, maka SVD dinyatakan sebagai:

$$A = U \Sigma V^T$$

U = matriks ortogonal berukuran $m \times m$ (kolomnya disebut *left singular vectors*).

Σ = matriks diagonal berukuran $m \times n$ yang berisi *singular values* (nilai-nilai non-negatif).

V^T = transpose dari matriks ortogonal V berukuran $n \times n$ (barisnya disebut *right singular vectors*).

SVD dapat dianggap sebagai cara untuk:

- ✓ Menguraikan informasi utama dari sebuah matriks.
- ✓ Mengidentifikasi arah dominan dalam data (mirip dengan *Principal Component Analysis / PCA*).
- ✓ Mengurangi dimensi data dengan tetap mempertahankan informasi terpenting.
- ✓ Ibaratnya, SVD seperti membongkar sebuah sinyal/gambar menjadi komponen-komponen inti yang paling berpengaruh, sambil mengabaikan "noise" atau detail yang kurang penting.

2. Penerapan SVD dalam Deteksi Suara

Pada deteksi suara, terutama dalam analisis sinyal suara dan pengenalan suara, data suara biasanya direpresentasikan dalam bentuk matriks. Suara dapat dikonversi menjadi representasi digital, seperti spektrum frekuensi atau mel-spectrogram, yang kemudian diubah menjadi matriks dua dimensi. Dalam hal ini, SVD berfungsi untuk mengurangi dimensi data, memisahkan informasi penting, dan mengidentifikasi pola yang relevan untuk klasifikasi atau deteksi.

Langkah-langkah penggunaan SVD pada deteksi suara:

1) Preprocessing Sinyal Suara: Langkah pertama adalah merekam dan mengubah sinyal suara menjadi bentuk digital, seperti bentuk gelombang atau spektrogram. Sebuah spektrogram adalah representasi grafis dari intensitas frekuensi dalam sinyal suara sepanjang waktu.
2) Transformasi Matriks: Setelah memperoleh sinyal suara yang telah diubah menjadi spektrogram, langkah berikutnya adalah mengubah data tersebut menjadi matriks. Di sini, setiap kolom atau baris matriks mewakili waktu atau frekuensi dalam sinyal suara.
3) Decomposisi Matriks dengan SVD: Matriks spektrogram kemudian didekomposisi menggunakan SVD. Ini menghasilkan tiga komponen utama: matriks UUU, matriks Σ\SigmaΣ, dan matriks VTV^TVT. Nilai singular yang ada di dalam Σ\SigmaΣ mewakili komponen penting dalam data, yang mengindikasikan fitur suara dominan.
4) Penyaringan dan Reduksi Dimensi: Dalam beberapa aplikasi, nilai singular yang lebih kecil dianggap noise atau informasi yang kurang penting. Oleh karena itu, komponen-komponen tersebut bisa dihapus, sehingga menghasilkan representasi yang lebih sederhana namun tetap mengandung informasi yang relevan. Dengan cara ini, kita dapat mengurangi kompleksitas data suara tanpa kehilangan kualitas informasi penting.
5) Ekstraksi Fitur untuk Deteksi: Fitur yang diperoleh dari hasil dekomposisi SVD, seperti vektor-vektor eigen yang ada di dalam UUU dan VTV^TVT, digunakan untuk menganalisis atau mendeteksi suara tertentu. Ini memungkinkan identifikasi pola, seperti pengenalan kata, deteksi emosi, atau identifikasi suara.

3. Keuntungan Menggunakan SVD untuk Deteksi Suara

a. Pengurangan Dimensi: SVD memungkinkan pengurangan dimensi yang signifikan, yang mengurangi kompleksitas data suara yang sangat besar, terutama pada aplikasi pengenalan suara dalam waktu nyata.
b. Peningkatan Akurasi: Dengan menghapus komponen yang tidak relevan atau noise, SVD meningkatkan akurasi dalam deteksi suara, karena model fokus pada informasi penting saja.
c. Pemrosesan Cepat: Dalam aplikasi yang memerlukan pengenalan suara dalam waktu nyata, pengurangan dimensi dengan SVD mempercepat proses deteksi suara, karena informasi yang lebih sedikit harus diproses.
d. Menangani Data Tidak Terstruktur: Suara adalah bentuk data yang sangat tidak terstruktur, yang dapat dipengaruhi oleh banyak variabel, seperti kecepatan bicara, akustik ruangan, dan gangguan latar belakang. SVD membantu mengekstrak fitur utama dari data suara yang rumit dan tidak terstruktur ini.

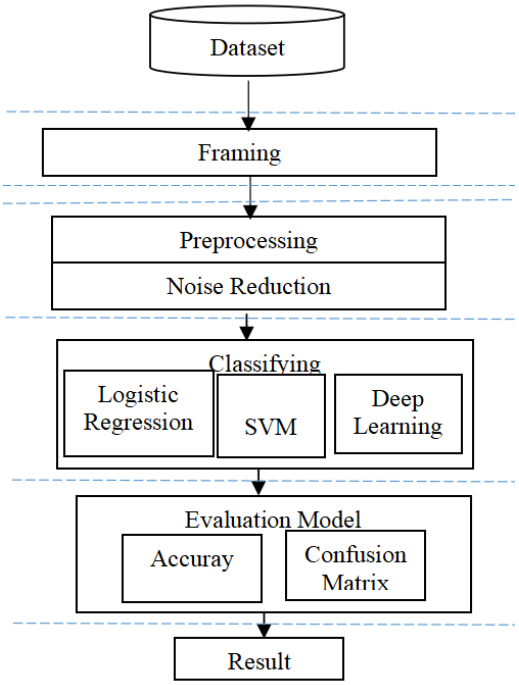
4. Aplikasi SVD dalam Deteksi Suara

a. Pengenalan Pembicara: SVD digunakan untuk memisahkan fitur-fitur unik dari suara individu, memungkinkan sistem untuk mengenali pembicara berdasarkan karakteristik suara mereka.
b. Pengenalan Ucapan: Dalam pengenalan ucapan otomatis (ASR - Automatic Speech Recognition), SVD membantu mengidentifikasi kata atau frasa yang diucapkan dengan mengurangi noise dan meningkatkan akurasi pengenalan suara.
c. Deteksi Emosi: SVD juga digunakan dalam analisis ekspresi emosional dalam suara, di mana fitur yang dihasilkan membantu membedakan antara berbagai emosi berdasarkan intonasi dan pola suara.
d. Pendeteksian Gangguan atau Kebisingan: Dalam aplikasi keamanan atau pengawasan, SVD dapat digunakan untuk mendeteksi suara-suara tertentu yang menunjukkan kejadian mencurigakan, seperti pecahnya kaca atau suara teriakan.

Singular Value Decomposition adalah alat yang sangat kuat dalam deteksi suara, memungkinkan analisis suara yang lebih efisien dan efektif dengan mereduksi dimensi data dan menghilangkan noise. Penggunaan SVD dalam deteksi suara, baik itu untuk pengenalan suara, deteksi emosi, atau pengenalan ucapan, memberikan berbagai keuntungan yang signifikan dalam meningkatkan kecepatan dan akurasi sistem berbasis suara. Dengan demikian, SVD telah menjadi metode yang tidak hanya berguna dalam analisis suara, tetapi juga dalam berbagai aplikasi terkait kecerdasan buatan dan pemrosesan sinyal digital.

2.4. Metode Penelitian

Penelitian mengenai deteksi suara etnis jawa menggunakan deep learning dan singular value decomposition dilakukan melaui serangkaian percobaan / riset yang bisa dilihat pada gambar 1 sebagai berikut :



Gambar 1 : Tahapan Penelitian Deep Learning SVD

Tahapan penelitian yang dilakukan adalah sebagai berikut :

1) Perancangan Dataset

Perancangan dataset suara merupakan tahapan krusial dalam pengembangan sistem berbasis pengenalan suara, pemrosesan bahasa alami, maupun kecerdasan buatan yang berbasis audio, karena kualitas dan keragaman data suara sangat memengaruhi akurasi serta kemampuan generalisasi model yang dibangun. Dataset suara yang dirancang dengan baik harus mencakup berbagai variasi aksen, intonasi, bahasa, jenis kelamin, umur, lingkungan rekaman (seperti latar belakang bising atau sunyi), dan perangkat perekam, sehingga mampu merepresentasikan kondisi dunia nyata secara menyeluruh. Hal ini penting untuk memastikan bahwa sistem yang dikembangkan tidak hanya bekerja dalam kondisi laboratorium yang ideal, tetapi juga mampu menangani kompleksitas penggunaan di lapangan. Selain itu, perancangan dataset yang memperhatikan aspek etika, privasi, dan legalitas, seperti mendapatkan persetujuan dari subjek suara dan memastikan anonimitas, juga menjadi landasan penting agar proses pembangunan sistem suara berlangsung secara bertanggung jawab. Pada riset ini digunakan dataset rekaman suara gender berbahasa jawa dengan berjenis kelamin pria dan wanita, ada 5 pria dan 5 wanita yang terlibat dalam perekaman suara ini. Masing-masing gender mengatakan 3 kata yaitu “Makan”,”Minum”,”Tidur” masing-masing diucapkan sebanyak 10 kali.Perekaman suara dilakukan dengan menggunakan software Adobe Audition dengan tipe suara Mono,Sample Rate sebesar 48000 Hz

2) Framing

Framing adalah proses yang sangat penting dalam sistem pengenalan suara, di mana sinyal suara yang kontinu dibagi menjadi segmen-segmen kecil yang disebut frame. Proses ini bertujuan untuk menangani data suara yang berubah secara dinamis dan memungkinkan pemrosesan lebih lanjut, seperti ekstraksi fitur atau analisis karakteristik suara dalam jangka waktu tertentu. Framing membantu membagi sinyal audio yang sangat panjang menjadi bagian-bagian yang lebih kecil dan mudah dianalisis. Dalam konteks pengenalan suara, framing memberikan dasar untuk berbagai teknik analisis, seperti ekstraksi mel-frequency cepstral coefficients (MFCCs), yang digunakan untuk membedakan antara berbagai jenis suara.

1. Pengertian Framing dalam Pengenalan Suara

Framing dalam pengenalan suara adalah proses membagi sinyal suara yang bersifat kontinu menjadi segmen-segmen pendek, yang biasanya disebut *frame*. Setiap frame memiliki durasi waktu tertentu yang dianggap cukup untuk mencakup informasi relevan dari sinyal suara. Biasanya, durasi frame berkisar antara 20 hingga 40 milidetik. Dalam rentang waktu ini, karakteristik suara yang dihasilkan oleh sumber suara (seperti manusia) dianggap cukup stabil dan tidak berubah secara signifikan. Setelah frame dibentuk, setiap frame ini dianalisis untuk mengekstrak fitur yang dapat digunakan untuk pengenalan suara lebih lanjut.

2. Proses Framing

Framing dilakukan dalam beberapa langkah, yakni:

a. Pengambilan Sampel: Proses pertama dalam framing adalah pengambilan sampel dari sinyal suara analog untuk mengubahnya menjadi sinyal digital. Hal ini dilakukan dengan cara mengukur amplitudo suara pada interval waktu tertentu.
b. Pembentukan Frame: Setelah pengambilan sampel, sinyal digital dibagi menjadi frame-frame kecil. Setiap frame biasanya terdiri dari sejumlah sampel yang diambil dalam jangka waktu yang sangat singkat. Dalam kebanyakan aplikasi pengenalan suara, frame berdurasi sekitar 20 hingga 25 milidetik, dengan overlap (tumpang tindih) antara satu frame dengan frame lainnya sekitar 50%.
c. Windowing: Sinyal dalam setiap frame sering kali diperlakukan dengan menggunakan fungsi window (misalnya, Hamming atau Hanning window) untuk meminimalkan efek "discontinuity" pada batas-batas frame. Fungsi window ini membantu mengurangi efek samping yang bisa mengganggu kualitas analisis sinyal.

3. Mengapa Framing Diperlukan?

a. Stabilitas Sinyal: Suara manusia atau sinyal suara alami bersifat sangat dinamis dan berubah seiring waktu. Dengan membagi sinyal suara menjadi frame-frame pendek, sistem pengenalan suara dapat menangani fluktuasi tersebut dan menganalisis sinyal dalam jangka waktu yang relatif stabil.

b. Fitur yang Lebih Terukur: Dengan framing, sinyal suara dapat diperlakukan dalam blok-blok waktu yang lebih kecil, yang memungkinkan untuk ekstraksi fitur yang lebih terukur dan lebih relevan, seperti Mel Frequency Cepstral Coefficients (MFCCs), Linear Predictive Coding (LPC), dan Zero Crossing Rate (ZCR).
c. Penyederhanaan Proses Komputasi: Proses framing menyederhanakan pengolahan data besar yang dapat lebih mudah ditangani dalam unit yang lebih kecil. Hal ini mengurangi beban komputasi yang diperlukan untuk menganalisis sinyal suara secara keseluruhan.

4. Overlapping pada Framing

Pada banyak sistem pengenalan suara, frame tidak diambil secara terpisah tanpa tumpang tindih, tetapi frame-frame tersebut memiliki overlap (tumpang tindih). Ini berarti bahwa setiap frame berbagi sejumlah sampel dengan frame sebelumnya dan setelahnya. Tujuan dari overlapping ini adalah untuk memastikan bahwa informasi yang penting dari batas antar frame tetap tercapture, menghindari hilangnya detail penting saat frame dibentuk. Overlap biasanya berkisar antara 50% sampai 75% dari durasi setiap frame.

5. Pentingnya Ukuran Frame dan Overlap

Ukuran frame dan tingkat overlap sangat mempengaruhi kinerja sistem pengenalan suara.

a. Ukuran Frame: Jika frame terlalu besar, maka perubahan cepat dalam sinyal suara yang terjadi dalam waktu singkat bisa terlewat. Di sisi lain, jika frame terlalu kecil, informasi yang relevan mungkin tercacah atau hilang. Ukuran frame yang ideal umumnya ditemukan di kisaran 20-25 milidetik.
b. Overlapping: Dengan overlap yang tepat, sinyal suara dapat dianalisis dengan lebih cermat, terutama ketika perbedaan antara satu frame dengan lainnya sangat kecil. Overlap memastikan bahwa perubahan yang lebih halus dalam suara tidak akan terlewatkan.
c. Teknik Analisis Pasca Framing

Setelah framing dilakukan, berbagai teknik analisis dapat diterapkan pada setiap frame untuk mengekstrak fitur yang relevan. Beberapa teknik yang digunakan meliputi:

a. MFCC (Mel Frequency Cepstral Coefficients): MFCC adalah fitur yang paling umum digunakan dalam pengenalan suara. Teknik ini melibatkan analisis spektrum suara dalam setiap frame untuk menangkap informasi yang penting mengenai karakteristik suara manusia.
b. LPC (Linear Predictive Coding): LPC digunakan untuk memperkirakan parameter suara dalam setiap frame dan sering digunakan dalam pengolahan sinyal suara untuk kompresi atau pengenalan suara.
c. Formant Analysis: Formant analisis mengidentifikasi frekuensi suara utama dalam sinyal yang berhubungan dengan kualitas suara manusia, seperti vokal.

7. Aplikasi Framing dalam Pengenalan Suara

Framing sangat penting dalam berbagai aplikasi pengenalan suara, seperti:

a. Speech Recognition: Proses pengenalan ucapan memerlukan framing untuk menganalisis segmen-segmen pendek dari suara manusia, memungkinkan sistem untuk mengenali kata dan kalimat.
b. Speaker Identification: Framing memungkinkan sistem untuk menganalisis pola suara yang unik dari setiap pembicara, membantu dalam identifikasi pembicara.
c. Emotion Detection: Dengan framing, perubahan dalam intonasi atau emosi dalam suara dapat dianalisis dan dikenali.

3) Preprocessing

Preprocessing pada dataset suara merupakan tahap krusial dalam sistem pengenalan suara yang bertujuan untuk meningkatkan kualitas data dan mempersiapkannya agar optimal digunakan dalam proses pelatihan model. Tahapan ini mencakup berbagai proses seperti penghapusan noise latar belakang (noise reduction), normalisasi volume, segmentasi sinyal suara menjadi frame-frame kecil, serta ekstraksi fitur akustik seperti Mel-Frequency Cepstral Coefficients (MFCC), Spectrogram, atau Mel-spectrogram. Selain itu, proses preprocessing juga dapat melibatkan konversi stereo ke mono, resampling agar memiliki frekuensi sampling yang konsisten, dan padding atau trimming untuk menyamakan durasi tiap klip suara. Dengan preprocessing yang tepat, kualitas data meningkat sehingga memungkinkan algoritma pembelajaran mesin maupun deep learning mempelajari pola suara secara lebih efektif dan akurat. Suara yang telah direkam kemudian diolah menggunakan Adobe Audition, sinyal suara kemudian dipotong dengan durasi waktu yang sama yaitu sebesar 31586.Setelah dilakukan pemotongan suara dan dimasukkan dalam excel diperoleh 150 kata yang akan diolah pada proses selanjutnya. Preprocessing merupakan proses yang dilakukan untuk mengolah dataset dengan menghilangkan beberapa bagian yang tidak diperlukan sehingga diharapkan menghasilkan tingkat akurasi yang lebih tinggi.

4) Noise Reduction

Riset ini menggunakan fasilitas Noise Reduction dalam Adobe Audition sehingga suara yang dihasilkan akan lebih jernih. Noise reduction adalah proses yang digunakan untuk mengurangi atau menghilangkan suara-suara yang tidak diinginkan dalam rekaman audio. Suara yang tidak diinginkan, atau sering disebut sebagai "noise," bisa berasal dari berbagai sumber, seperti suara latar belakang (misalnya, dengungan perangkat elektronik), kebisingan lingkungan, atau rekaman audio yang buruk. Adobe Audition, sebagai salah satu perangkat lunak pengeditan audio profesional, menyediakan berbagai fitur untuk mengatasi masalah ini dan meningkatkan

kualitas rekaman audio. Langkah-langkah lengkap dan komprehensif dalam melakukan noise reduction pada data suara menggunakan Adobe Audition:

1. Mempersiapkan Data Suara

Sebelum memulai proses pengurangan noise, penting untuk memastikan bahwa data suara yang digunakan dalam keadaan siap untuk diproses. Langkah pertama adalah membuka file audio yang ingin diproses di Adobe Audition. Pengguna dapat melakukannya dengan memilih menu *File* dan kemudian memilih *Open* untuk memuat file audio ke dalam proyek Adobe Audition.

Jika suara yang ingin diproses mengandung banyak noise latar belakang, pastikan untuk memilih segmen audio yang benar-benar berisi noise (tanpa suara utama atau dialog). Bagian ini akan digunakan untuk menganalisis profil noise.

2. Menganalisis Profil Noise

Langkah pertama dalam noise reduction adalah menganalisis profil noise, yang akan digunakan Adobe Audition untuk memahami karakteristik noise tersebut. Ikuti langkah-langkah berikut:

- ✓ Pilih segmen dari rekaman yang hanya mengandung noise (tanpa suara penting). Pastikan bahwa hanya suara latar belakang yang terdengar di segmen ini.
- ✓ Pilih bagian tersebut dengan alat seleksi di timeline.
- ✓ Masuk ke menu *Effects* dan pilih *Noise Reduction/Restoration* > *Capture Noise Print*.
- ✓ Adobe Audition akan mempelajari karakteristik suara yang ada pada segmen ini dan membuat "profil noise."

3. Melakukan Noise Reduction

Setelah Adobe Audition berhasil menangkap profil noise, langkah berikutnya adalah mengaplikasikan proses noise reduction pada seluruh atau sebagian dari file audio. Berikut caranya:

- ✓ Pilih seluruh file audio yang ingin diproses, atau jika hanya bagian tertentu yang ingin diubah, seleksi bagian tersebut.
- ✓ Masuk ke menu *Effects* dan pilih *Noise Reduction/Restoration* > *Noise Reduction (Process)*.
- ✓ Di dalam jendela *Noise Reduction (Process)*, pengguna akan melihat dua komponen utama:

1) Noise Reduction: Ini adalah pengaturan untuk menentukan seberapa banyak noise yang akan dikurangi. Angka yang lebih tinggi berarti lebih banyak noise yang akan dihapus, namun harus hati-hati agar suara asli tidak terdistorsi.
2) Reduce by: Mengatur seberapa keras pengurangan noise akan dilakukan. Angka ini mengontrol seberapa banyak volume noise yang dikurangi, dan semakin tinggi angkanya, semakin banyak noise yang dihilangkan.

- ✓ Pengguna juga dapat mengatur slider *FFT Size* dan *Precision Factor* untuk menyesuaikan tingkat presisi dan kualitas pengurangan noise. Semakin tinggi angka pada kedua parameter ini, semakin akurat proses pengurangan noise yang dilakukan.

Setelah pengaturan selesai, klik *Apply* untuk mengaplikasikan noise reduction pada data suara.

4. Mendengarkan Hasilnya

Setelah pengurangan noise diterapkan, penting untuk mendengarkan hasilnya. Dengarkan apakah suara asli masih terdengar jelas atau ada distorsi yang muncul. Jika terlalu banyak noise yang dihapus, dan suara utama menjadi teredam atau tidak alami, pengguna bisa melakukan penyesuaian lebih lanjut. Jika perlu, pengguna dapat mengulang langkah-langkah di atas dengan pengaturan yang lebih rendah untuk mendapatkan hasil yang lebih alami. Gunakan fitur *Preview* di jendela *Noise Reduction (Process)* untuk mendengarkan perubahan sebelum mengaplikasikannya secara permanen.

5. Menggunakan Fitur *Click/Pop Eliminator*

Selain noise umum seperti dengungan atau suara latar, terkadang data suara dapat mengandung *clicks* atau *pops* yang tidak diinginkan, biasanya berasal dari gangguan sinyal saat rekaman. Adobe Audition menyediakan fitur khusus untuk menghilangkan suara tersebut:

- ✓ Pilih bagian rekaman yang mengandung click atau pop.
- ✓ Masuk ke menu *Effects* > *Noise Reduction/Restoration* > *Click/Pop Eliminator*.
- ✓ Sesuaikan pengaturan untuk menghilangkan klik atau pop sesuai kebutuhan.
- ✓ Terapkan efek ini pada segmen yang relevan.

6. Finalisasi dan Penyimpanan

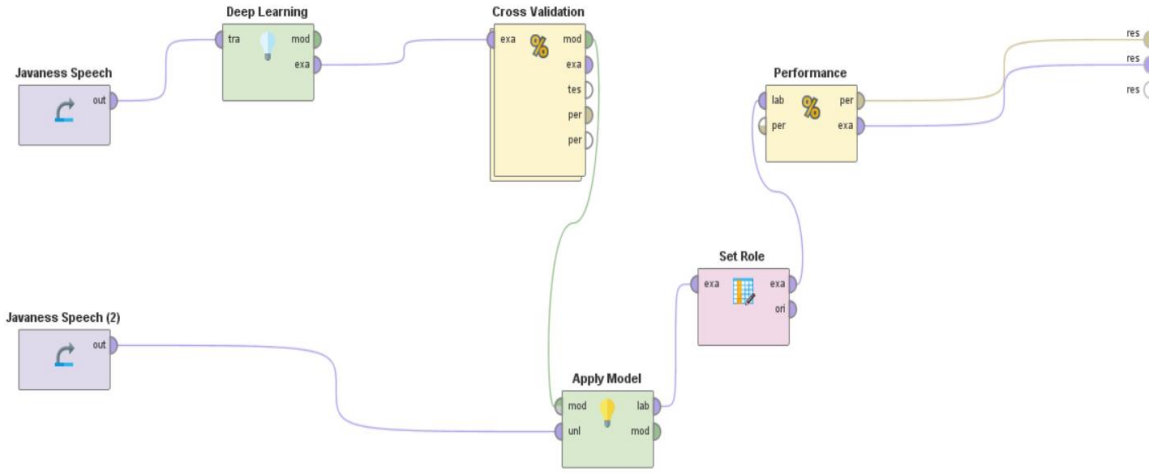
Setelah noise reduction diterapkan dan hasilnya telah diperiksa, langkah terakhir adalah melakukan pengolahan akhir dan menyimpan file audio. Jika ada bagian lain dari rekaman yang membutuhkan pengolahan lebih lanjut, seperti normalisasi volume atau equalization, lakukan proses tersebut.

Untuk menyimpan file, pilih *File* > *Save As* dan pilih format yang diinginkan (misalnya, WAV, MP3, atau format lainnya).

2.5. Klasifikasi Menggunakan Rapid Miner

RapidMiner adalah sebuah platform analitik data yang powerful dan user-friendly yang mendukung pemodelan berbasis *deep learning* melalui antarmuka *drag-and-drop* yang intuitif, menjadikannya sangat cocok digunakan oleh peneliti, praktisi data, dan akademisi dalam membangun model kecerdasan buatan tanpa perlu menulis banyak kode pemrograman. Dalam konteks *deep learning*, RapidMiner menyediakan integrasi dengan *Deep Learning Extension* yang memungkinkan pengguna untuk membangun dan melatih jaringan saraf tiruan (neural networks) berlapis-lapis untuk menyelesaikan berbagai permasalahan kompleks seperti klasifikasi gambar,

analisis sentimen teks, prediksi deret waktu, dan deteksi anomali. Platform ini mendukung berbagai arsitektur seperti deep neural networks dan convolutional neural networks, serta memungkinkan tuning parameter secara visual dan otomatis dengan fitur *auto-modeling*. Selain itu, kemampuan RapidMiner untuk mengintegrasikan *pipeline* data dari tahap praproses, transformasi fitur, pelatihan model, hingga evaluasi dan visualisasi hasil membuatnya menjadi solusi end-to-end yang efisien untuk pengembangan model *deep learning* yang andal dan dapat direproduksi. Pada penelitian dipergunakan aplikasi Rapid Miner untuk mengklasifikasi data suara gender pada suku Jawa dengan desain tampilan seperti pada gambar 1 sebagai berikut :

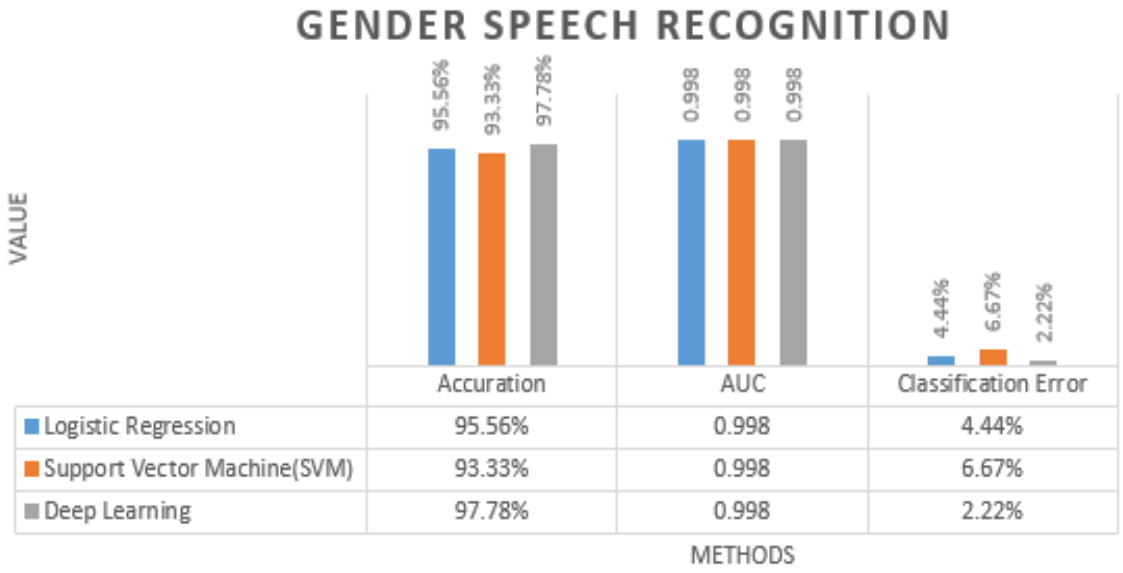


Gambar 1 : Desain Klasifikasi Gender Suku Jawa

Pada gambar 1 terlihat bahwa dataset Javaness Speech dipergunakan pada riset ini sebagai data training dan data testing dengan pengolahan menggunakan metode Deep Learning. Setelah masuk proses Cross Validation maka mode diaplikasikan untuk mendapatkan hasil akhir evaluasi dalam rangka mengetahui kinerja model Deep Learning pada pengolahan suara.

2.6. Hasil Penelitian

Pada penelitian yang telah dilakukan untuk mengelola data suara etnis jawa menggunakan aplikasi Rapid Miner didapatkan hasil pengukuran kinerja model sebagai berikut :



Gambar 2 : Hasil Pengukuran Model

Pada gambar 2 dapat terlihat bahwa model pengenalan suara menggunakan Deep Learning mencapai kinerja akurasi tertinggi sebesar 97.78 % lebih baik kinerjanyya dibandingkan dengan metode Logistic Regression maupun Support Vector Machine dalam pengenalan suara gender berbahasa Jawa. Selain itu model Deep Learning lebih rendah dalam kinerja Classification Error sebesar 2.22 % lebih baik dari 2 model lain.

BAB 3

PENGENALAN SUARA MENGGUNAKAN DATA AUGMENTATION DEEP NEURAL NETWORK

3.1 Data Augmentation

Data augmentation merupakan teknik penting dalam pemrosesan data audio, khususnya dalam pengembangan sistem pengenalan suara otomatis (Automatic Speech Recognition/ASR). Teknik ini bertujuan untuk meningkatkan keragaman dan jumlah data pelatihan tanpa harus mengumpulkan data baru dari dunia nyata, yang sering kali mahal dan memakan waktu. Dalam konteks pengenalan suara, data augmentation membantu model machine learning atau deep learning untuk lebih robust dan mampu mengenali variasi ucapan dalam berbagai kondisi akustik. Secara umum, data augmentation pada pengenalan suara dilakukan dengan memodifikasi sinyal suara asli agar menyerupai berbagai kemungkinan situasi nyata yang akan dihadapi oleh sistem saat digunakan.

Penerapan data augmentation sangat krusial, terutama ketika dataset pelatihan terbatas, tidak seimbang, atau tidak cukup representatif terhadap populasi nyata. Dalam pengenalan suara, masalah seperti aksen, dialek, noise lingkungan, dan kualitas mikrofon sering kali menurunkan performa sistem jika tidak ditangani dengan baik melalui augmentasi. Dalam praktiknya, data augmentation diimplementasikan secara dinamis saat pelatihan model (on-the-fly) atau dilakukan sebelumnya dalam preprocessing. Framework seperti Kaldi, ESPnet, dan torchaudio di PyTorch telah menyediakan pustaka atau fungsi bawaan untuk memfasilitasi augmentasi data audio.

Secara keseluruhan, penggunaan teknik data augmentation dalam pengenalan suara bukan hanya meningkatkan akurasi model, tetapi juga memperkuat ketahanannya (robustness) terhadap kondisi nyata yang kompleks dan beragam. Dengan kata lain, data augmentation merupakan strategi vital dalam menjembatani kesenjangan antara data pelatihan yang terbatas dan kompleksitas data dunia nyata.

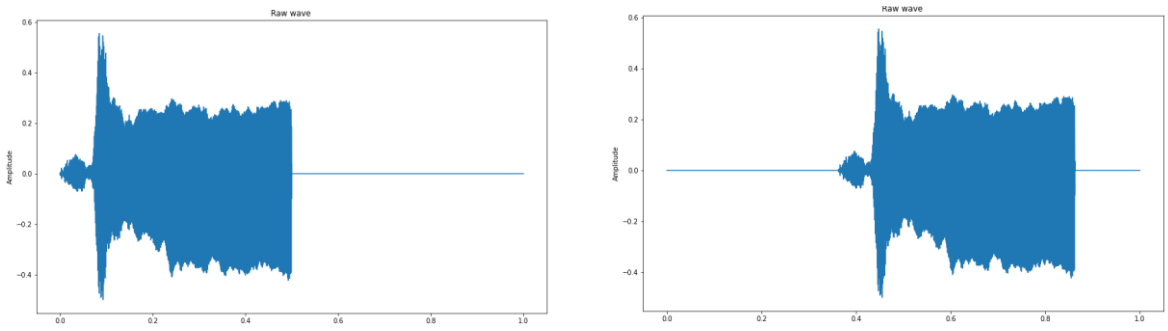
3.2 Pitch Shifting

Pitch Shifting adalah salah satu teknik *data augmentation* yang umum digunakan dalam pemrosesan sinyal audio, khususnya pada bidang seperti pengenalan suara (*speech recognition*), deteksi emosi dari suara, dan klasifikasi musik atau bunyi. Teknik ini bekerja dengan cara mengubah tinggi nada (*pitch*) dari sinyal audio tanpa mengubah durasi atau kecepatan suara aslinya. Proses ini dilakukan melalui algoritma digital seperti *phase vocoder*, *resampling dengan time-stretching*, atau metode berbasis *Fourier Transform* yang mempertahankan konten temporal suara.

Pada konteks *data augmentation*, Pitch Shifting bertujuan untuk menciptakan variasi buatan dari dataset audio yang terbatas. Misalnya, jika kita memiliki satu rekaman suara seorang pembicara yang mengatakan "Halo", maka dengan pitch shifting ±2 semitone, kita bisa menghasilkan dua versi baru dari suara tersebut dengan nada yang lebih tinggi atau lebih rendah, namun tetap menyampaikan kata yang sama. Hal ini sangat berguna dalam meningkatkan *generalization* model pembelajaran mesin atau *deep learning*, karena model akan belajar dari berbagai variasi vokal dan karakteristik suara tanpa perlu merekam data tambahan secara manual.

Kelebihan utama dari pitch shifting adalah kemampuannya untuk memperkaya dataset secara cepat, efisien, dan realistis, tanpa menimbulkan artefak yang merusak kualitas semantik dari audio asli. Teknik ini juga menjaga *label* atau anotasi tetap relevan (misalnya, label "anjing menggonggong" tetap valid meskipun pitch-nya dinaikkan sedikit), sehingga sangat ideal untuk digunakan dalam sistem klasifikasi berbasis suara.

Namun demikian, perlu diperhatikan bahwa penggunaan pitch shifting secara berlebihan dapat menyebabkan model menjadi bias terhadap pitch tertentu, atau memperkenalkan distorsi yang tidak alami jika tidak disesuaikan dengan parameter yang optimal. Oleh karena itu, pemilihan rentang pitch shift (biasanya antara ±1 hingga ±3 semitone) harus disesuaikan dengan jenis dataset dan tujuan aplikasi model. Visualisasi sinyal suara metode pitch shifting dapat dilihat pada gambar 3 sebagai berikut :



sinyal suara asli sinyal suara pitch shifting

Gambar 3 : Sinyal suara metode Pitch Shifting

Secara keseluruhan, pitch shifting adalah metode augmentasi yang sederhana namun efektif dalam memperbesar keragaman data audio, memperbaiki kinerja model pembelajaran suara, serta mengurangi risiko overfitting pada dataset yang terbatas. Teknik ini kini menjadi bagian penting dalam *toolkit* preprocessing audio modern, bersanding dengan metode lain seperti *time stretching*, *additive noise*, *random cropping*, dan *reverberation simulation*.

3.3 Time Stretching

Time Stretching adalah teknik dalam pengolahan sinyal audio yang memungkinkan perubahan durasi sinyal tanpa mengubah nada atau frekuensinya. Dalam konteks pengenalan suara (*speech recognition*), *time stretching* digunakan untuk memodifikasi kecepatan bicara—baik mempercepat maupun memperlambat—tanpa menyebabkan distorsi tonal. Ini berbeda dengan *pitch shifting* yang mengubah nada suara.

Teknik ini sangat penting dalam *data augmentation* untuk memperkaya keragaman data latih model, meningkatkan ketahanan (*robustness*) model terhadap variasi tempo bicara dari berbagai pembicara.

Tujuan dan Manfaat dalam Pengenalan Suara

1) Meningkatkan generalisasi model: Dengan menambahkan variasi kecepatan suara tanpa mengubah intonasi, model pengenalan suara dapat belajar mengenali ucapan dari berbagai pembicara dengan tempo bicara berbeda.

2) Simulasi kondisi dunia nyata: Dalam percakapan sehari-hari, orang berbicara dengan kecepatan yang berbeda-beda. Time stretching membantu menciptakan data yang mencerminkan keragaman ini.

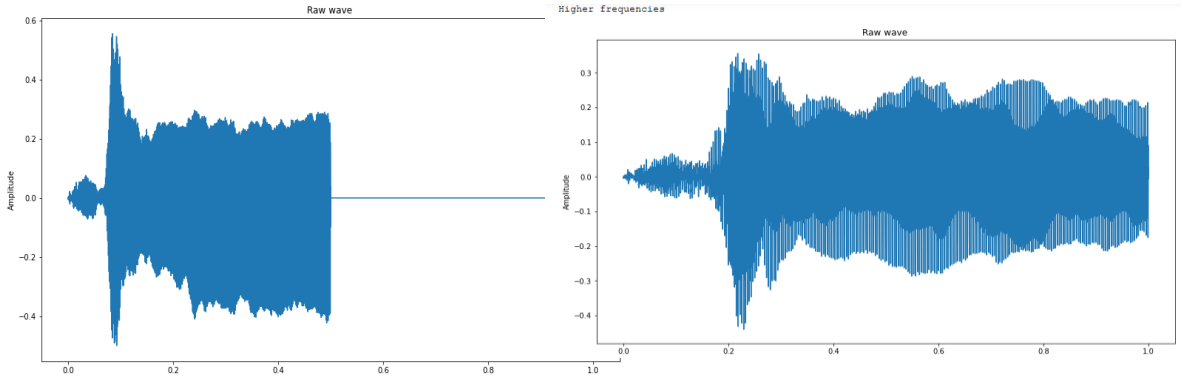
3) Memperbesar dataset: Dalam skenario di mana jumlah data asli terbatas, time stretching dapat digunakan sebagai teknik augmentasi untuk menghasilkan data baru tanpa perlu rekaman ulang.

Cara Kerja Time Stretching

Metode ini umumnya diimplementasikan dengan menggunakan transformasi sinyal, seperti:

- ✓ Short-Time Fourier Transform (STFT): Audio dibagi menjadi frame-frame kecil, kemudian dilakukan perubahan pada durasi frame sambil mempertahankan fase spektrum frekuensi.
- ✓ Phase Vocoder: Salah satu pendekatan paling umum yang memanfaatkan STFT untuk mempertahankan karakteristik spektral sambil memperlambat atau mempercepat sinyal.
- ✓ Time-Domain Techniques: Seperti *Synchronized Overlap-Add* (SOLA), yang menggeser dan menjahit segmen audio secara cerdas untuk menghindari artefak.

Visualisasi sinyal suara metode pitch shifting dapat dilihat pada gambar 4 sebagai berikut :

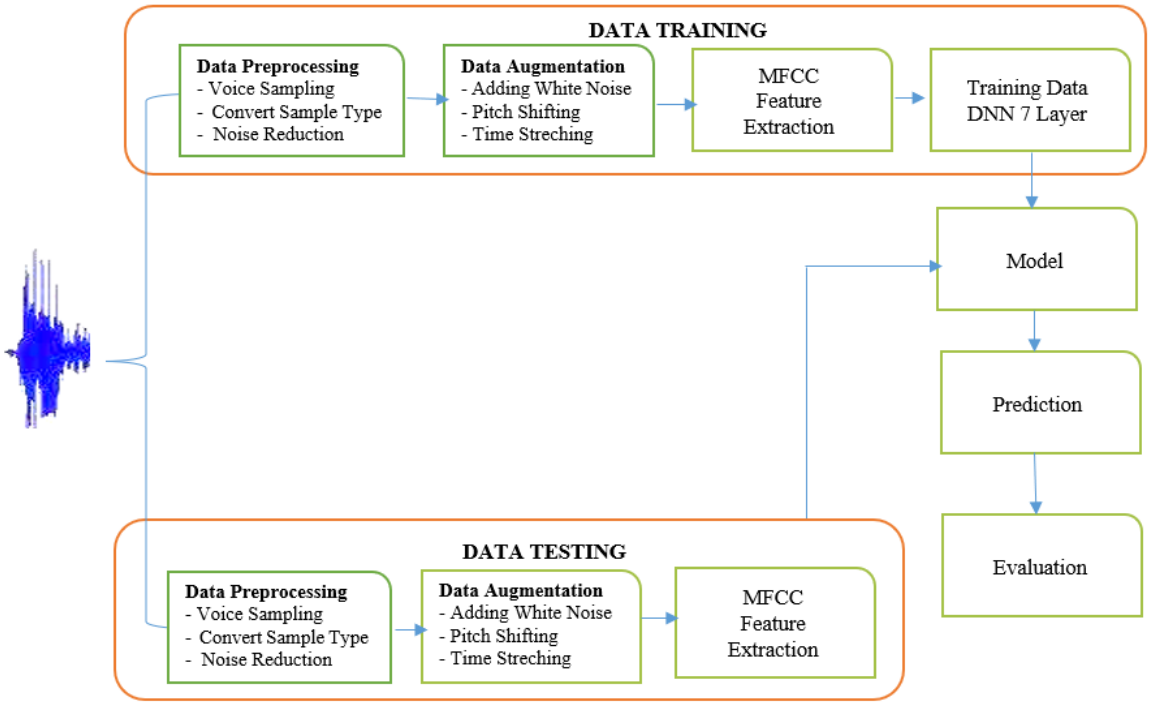


sinyal suara asli sinyal suara time stretching

Gambar 4 : Sinyal suara metode Time Stretching

3.4 Metode Penelitian

Penelitian yang dilakukan mengenai deteksi suara pembicara multietnis menggunakan penelittian experimental dengan desain penelitian yang terlihat pada gambar 5 sebagai berikut :



Gambar 5 : Tahapan Penelitian

A. Dataset dan Preprocessing

Penelitian ini menggunakan dataset “801 Languages Spoken In Indonesian” part 2 yang berisi pembicara dari berbagai etnics di Indonesia yang mempunyai durasi waktu 34 menit 44 detik yang diambil dari Youtube dengan alamat https://www.youtube.com/watch?v=FkwXbCY1rWg&t=370s. Setiap pembicara suku diambil sampel suara berdurasi 1 detik menggunakan aplikasi Adobe Audition CS6 dengan sampel rate 44.100 Hz Bit Depth 32 bit(float) Mono channel WAV setting 32-bit Floating Point(IEEE). Suara tersebut dimasukkan dalam folder data trainning dengan masing-masing pembicara suku akan berisi 10 sampel suara. Pada beberapa sampel suara yang kurang dari 10 akan ditambahkan suara dengan teknik data augmentation berupa adding noise, time streching dan pitch shifting. Hasil suara yang ditambahkan menggunakan data augmentation akan disesuaikan dengan setting suara

sampel data sebelumnya yaitu Mono channel WAV setting 32-bit Floating Point(IEEE) dengan sampel rate 44.100 Hz Bit Depth 32 bit(float).

Setiap file sampel suara yang dimasukkan ke dalam folder baik data training maupun data testing sebelumnya dilakukan segmentasi dengan pola sebagai berikut:

1. Modality (01 = full-AV, 02 = video-only, 03 = audio-only).
2. Vocal (01 = speech, 02 = song).
3. Emosi (01 = neutral, 02 = calm, 03 = happy, 04 = sad, 05 = angry, 06 = fearful, 07 = disgust, 08 = surprised).
4. Intensitas Emosi (01 = normal, 02 = strong).
5. Pengulangan (01 = repeataion 1, 02 = repeataion 2).
6. Nomor urutan pembicara per suku/daerah dari 01 sampai dengan 10
7. Suku bangsa pembicara sebagai kelas dari 01 sampai dengan 100

B. Data Augmentation

Pada penelitian ini dipergunakan 3 pendekatan pada proses data augmentation yaitu :

1) Adding White Noise

Adding White Noise merupakan salah satu teknik data augmentation dalam pemrosesan sinyal suara yang bertujuan untuk meningkatkan jumlah dan variasi data pelatihan tanpa perlu merekam ulang. Teknik ini dilakukan dengan menambahkan suara bising acak (white noise) ke dalam sinyal audio asli. White noise adalah jenis suara yang memiliki spektrum frekuensi yang rata, menyerupai suara mendesis atau statis. Penambahan white noise dapat membantu model pembelajaran mesin atau deep learning menjadi lebih robust terhadap variasi lingkungan nyata yang bising, seperti suara latar di jalanan, ruang kelas, atau tempat umum. Teknik ini juga efektif dalam mengurangi overfitting karena memperkenalkan variasi kecil namun signifikan ke dalam data. Penelitian ini menggunakan script pada bahasa python dengan listing program sebagai berikut :

```
import os
import numpy as np
import soundfile as sf
def add_noise_to_wav(input_file, output_file, noise_factor):
    # Membaca file WAV
    wav_data, sample_rate = sf.read(input_file)
    # Menghasilkan noise dengan amplitudo yang sesuai
    noise = np.random.randn(len(wav_data))
    noise /= np.max(np.abs(noise))
    noise *= noise_factor
    # Menambahkan noise ke data WAV
    augmented_data = wav_data + noise
    # Menyimpan file WAV yang sudah diaugmentasi
    sf.write(output_file, augmented_data, sample_rate)
# Menentukan path file input dan output
input_file = '/content/drive/My Drive/Colab Notebooks/IndonesianSpeakers/data/03-01-01-01-01-01-01.wav'
output_file = '/content/drive/My Drive/Colab Notebooks/IndonesianSpeakers/output/generated_03-01-01-01-01-01-01.wav'
# Menentukan faktor noise (0.0 - 1.0)
noise_factor = 0.1
# Melakukan data augmentation dengan menambahkan noise
add_noise_to_wav(input_file, output_file, noise_factor)
```

2) Pitch Shifting

Pitch Shifting merupakan salah satu teknik *data augmentation* dalam pemrosesan suara yang bertujuan untuk memperbanyak dan memvariasikan data suara tanpa mengubah isi atau maknanya. Teknik ini bekerja dengan mengubah tinggi nada (pitch) suatu sinyal audio, baik dinaikkan maupun diturunkan, tanpa mempengaruhi durasi suara. Pada konteks pengenalan suara (*speech recognition*), Pitch Shifting membantu model agar lebih toleran terhadap variasi suara pengguna, seperti perbedaan jenis kelamin, intonasi, atau gaya bicara. Misalnya, suara asli seseorang dapat digeser naik beberapa *semitone* untuk mensimulasikan suara anak-anak, atau diturunkan untuk meniru suara yang lebih berat, seperti suara pria dewasa. Dengan demikian, model menjadi lebih robust terhadap kondisi nyata yang beragam. Secara teknis, Pitch

Shifting dapat dilakukan menggunakan algoritma seperti phase vocoder atau time-domain harmonic scaling, yang mempertahankan durasi suara tetap stabil sambil memodifikasi spektrum frekuensi. Hasil augmentasi ini kemudian digunakan sebagai tambahan pada dataset pelatihan untuk meningkatkan akurasi dan generalisasi model dalam tugas-tugas seperti klasifikasi emosi, identifikasi pembicara, dan pengenalan kata kunci. Script untuk pendekatan pitch shifting adalah sebagai berikut:

```
from pydub import AudioSegment

def pitch_shift(audio_file, output_file, semitones):
    sound = AudioSegment.from_file(audio_file)
    shifted_sound = sound._spawn(sound.raw_data, overrides={
        "frame_rate": int(sound.frame_rate * (2.0 ** (semitones / 42.0)))
    })
    shifted_sound.export(output_file, format="wav")

# Contoh penggunaan
audio_file = "/content/drive/My Drive/Colab Notebooks/IndonesianSpeakers/data/03-01-01-01-01-02-02.wav"  # Ganti dengan nama file audio input Anda
output_file = "/content/drive/My Drive/Colab Notebooks/IndonesianSpeakers/output/generated_03-01-01-01-01-02-02.wav"  # Ganti dengan nama file audio output yang dihasilkan
semitones = 3  # Jumlah semitone yang ingin di-shift (positif untuk naik, negatif untuk turun)
pitch_shift(audio_file, output_file, semitones)
print("File audio berhasil diubah dengan pitch shift dan disimpan sebagai", output_file)
```

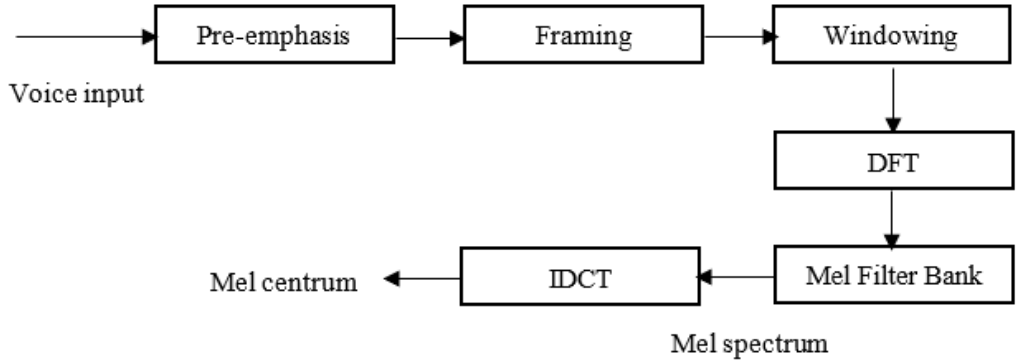
3) Time Stretching

```
import librosa
import numpy as np
import matplotlib.pyplot as plt
class AudioAugmentation:
    def read_audio_file(self, file_path):
        input_length = 16000
        data = librosa.core.load(file_path)[0]
        if len(data) > input_length:
            data = data[:input_length]
        else:
            data = np.pad(data, (0, max(0, input_length - len(data))), "constant")
        return data
```

```
def write_audio_file(self, file, data, sample_rate=16000):
        librosa.output.write_wav(file, data, sample_rate)
    def plot_time_series(self, data):
        fig = plt.figure(figsize=(14, 8))
        plt.title('Raw wave ')
        plt.ylabel('Amplitude')
        plt.plot(np.linspace(0, 1, len(data)), data)
        plt.show()
    def stretch(self, data, rate=1):
        input_length = 16000
        data = librosa.effects.time_stretch(data, rate)
        if len(data) > input_length:
            data = data[:input_length]
        else:
            data = np.pad(data, (0, max(0, input_length - len(data))), "constant")
        return data
# Create a new instance from AudioAugmentation class
aa = AudioAugmentation()
```

C. Feature

Pada penelitian ini kami mempergunakan Mel Frequency Cepstral Coeffisients (*MFCC*) yang merupakan salah satu metode yang robust untuk dipergunakan dalam melakukan ektraksi ciri pada speaker recognition, Selain itu proses extraksi fitur menggunakan Librosa pada bahasa pemrograman Python yang mempunyai fungsi untuk membaca file audio dan membantu proses ekstraksi menggunakan MFCC.



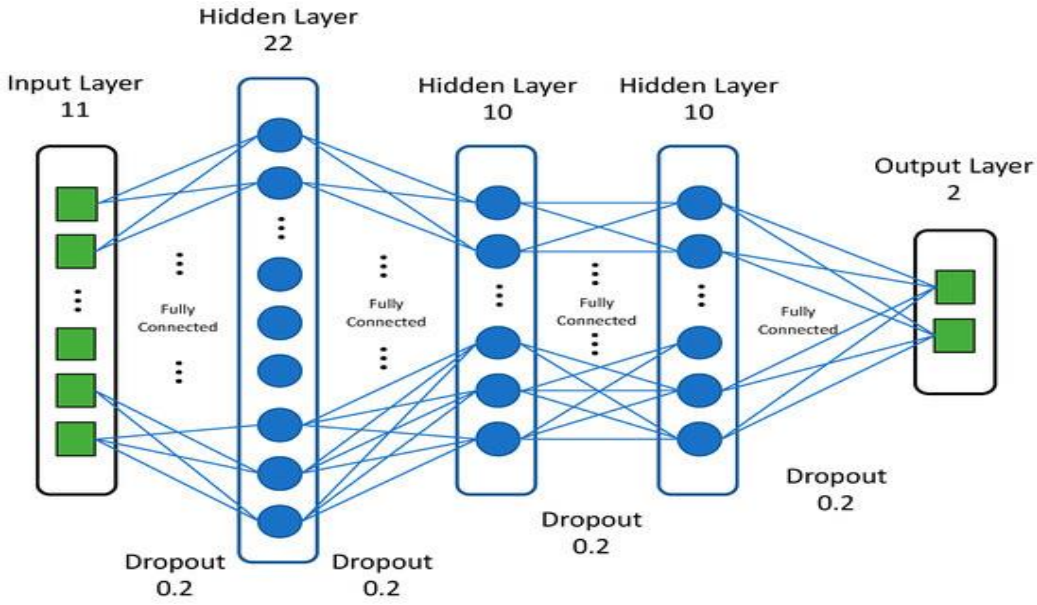
Gambar 6 : Gambar proses kerja metode MFCC

Pada metode MFCC input suara audio diperkuat pada proses pre-emphasis untuk meningkatkan signal to noise ratio agar audio tidak terpengaruh oleh noise, Selanjutnya dilakukan proses framing dengan cara membagi suara audio menjadi beberap frame dengan jumlah sinyal sama. Windowing merupakan proses pembobotan dari frame yang dihasilkan menggunakan fungsi window, Proses berikutnya DFT (Discrete Fourier Transform) yang merupakan proses yang bertujuan untuk melakukan analisis sinyal frekuensi dari sinyal waktu diskrit kemudian Mel Filter Bank digunakan untuk menghitung MFCC yang berasal dari ujaran asli. IDCT (Inverse Discrete Cosine Transform) merupakan proses yang dipergunakan dalam mendapatkan spektrum magnitudo yang diinterpolasi dalam menghasilkan mel centrum.

D. Deep Neural Network (DNN)

Metode deep learning telah berkembang menghasilkan berbagai metode yang robust dalam speaker recognition, Salah satu metode DL yang sering dipergunakan adalah Deep Neural Network. DNN dipergunakan dalam berbagai riset pengenalan suara karena memiliki banyak kelebihan dibandingkan metode tradisional dalam machine learning,

Metode ini banyak dipergunakan karena memiliki keunggulan dalam berbagai bidang riset antara lain pengenalan objek visual, geografi dan pengenalan suara(Seifert et al., 2017). Pada penelitian lain DNN mencapai tingkat kinerja yang sigfnificant seperti penelitian yang dilakukan oleh (Mclaren et al., 2015) mengenai telephone speech, (Seki et al., 2016) mengenai ujaran pendek menggunakan DNN based accoustic model, (Snyder et al., 2018) tentang DNN dengan data augmentation, (Novotný et al., 2019) juga melakukan riset dengan DNN berbasis autoencorder dan (Saleem & Irfan Khattak, 2020) yang melakukan penelitian mengenai kegunaan DNN dalam single channel speaker independent multi-talker speech separation.



Gambar 7 : Arsitektur Deep Neural Network (DNN)

Arsitektur model DNN terdiri dari input layer, beberapa lapisan hidden layer,dropout dan output layer(Rajyaguru et al., 2020), Model DNN merupakan pengembangan dari Neural Network yang pada dasarnya merupakan fungsi dalam model matematika $f : X \rightarrow Y$ yang dapat dijelaskan sebagai berikut :

1. Input Layer

   Merupakan lapisan yang terdiri dari *neuron* yang menerima data masukan dari variabel X, Neuron dari lapisan ini terhubung secara langsung dengan hidden layer.

2. Hidden Layer

   Merupakan suatu lapisan yang bisa terdiri dari beberapa lapisan neuron yang menerima data dari input layer.

3. Dropout

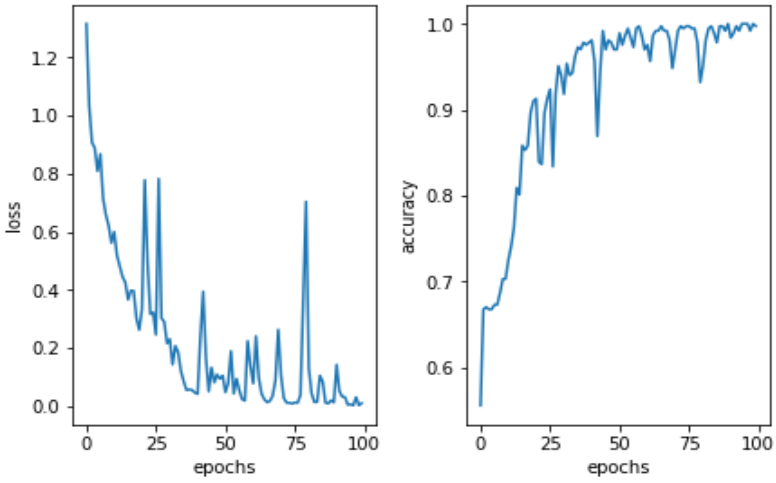
   Merupakan model tunggal yang dipergunakan untuk mensimulasikan yang memiliki sejumlah besar aarsitektur jaringan yang berbeda yang dapat dipergunakan untuk mengurangi masalah overfitting pada model Neural Network.

4. Output Layer

   Merupakan suatu lapisan yang terdiri dari neuron yang menerima data dari hidden layer atau langsung dari input layer yang nilai luarannya merepresentasikan hasil perhitungan dari X menjadi nilai Y.

3.5 Hasil Penelitian

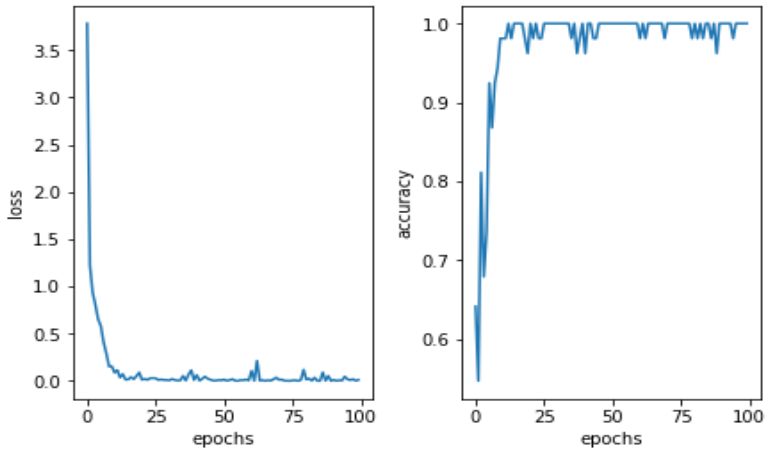
Pada riset ini kami mempergunakan teknik data augmentation dengan adding white noise,time streching dan pitch shifting dengan setting sample rate 44100 mono dalam menguji 700 suara pembicara etnics di Indonesia. Dengan model Data Augmentation DNN 7 Layer (DA-DNN7L) pada epoch 100 batch size 4 menghasilkan kinerja data training yang ditunjukkan dalam gambar dibawah ini:



Gambar 8 : Kinerja DA-DNN7L pada data training

Pengujian dataset training menunjukkan hasil akurasi 95.28 % dengan tingkat loss sebesar loss 0.0102, Jika kita melihat pada gambar 8 diatas tingkat loss semakin menurun mulai

dari epoch 1 sampai dengan 100 namun cenderung tidak stabil pada epoch ke 22,27,36,62,70,79 loss meningkat yang otomatis diikuti oleh kinerja akurasi yang menurun pada epoch ke 22,27,36,62,70,79 meskipun pada akhirnya stabil di atas angka 95% pada epoch ke 90.



Gambar 9 : Kinerja DA-DNN7L pada data testing

Pengujian pada model yang telah dihasilkan yang di uji menggunakan data testing kami menghasilkan tingkat akurasi yang tinggi tsebesar 99.76% dan tingkat loss yang rendah di angka 0.052, Pada model DA-DNN7L terbukti menghasilkan metode yang robust dalam pengenalan pembicara dibuktikan dengan tidak adanya overfitting pada pengujian model ini, Justru akurasi yang ditunjukkan model yang disusulkan pada data testing semakin meningkat dengan tingkat loss yang juga semakin menurun seperti ditunjukkan pada gambar 9 diatas. Jika pekerjaan ini dibandingkan dengan hasil penelitian lain mengenai pengenalan pembicara dengan menggunakan machine learning maka hasil kinerjanya dapat dilihat seperti pada tabel 1 sebagai berikut :

Tabel 1

Perbandingan kinerja Speaker Recognition dengan Machine Learning

Metode	Etnics	Dataset	Accuracy (%)
Gaussian Mixture Models (Chakroun & Frikha, 2020)	English	TIMIT	98.44%

Hidden Markov Model(Mouaz et al., 2019)	Arab	Modern Standard Arabic (MSA) and Moroccan Dialect Speakers	90%
Support Vector Machine(Hanifa et al., 2020)	Malaysian	The speaker's ethnicity based on four categories: Malay, Chinese, Indian and Bumiputera	57.7%
Ours (Riset Ini))	**Indonesian**	**Indonesian Etnics Speakers**	**99.76%**

Tabel 1 menunjukkan bahwa metode yang kami usulkan (DA-DNN7L) merupakan metode yang paling tinggi tingkat akurasinya dalam pengenalan pembicara dibandingkan dengan berbagai metode lain yang mempergunakan pendekatan machine learning konvensional , Metode kami mengungguli GMM dan HMM yang dikenal sebagai metode yang paling robust dalam penelitian pengenalan pembicara. Pada penelitian lain dengan menggunakan Deep Learning perbandingan kinerja beberapa modelnya ditampilkan dalam tabel 2 sebagai berikut:

Tabel 2

Perbandingan kinerja Speaker Recognition dengan Deep Learning

Dataset	Feature Extraction	Model	Accuracy (%)
OC16 MixASR-CHEN(Long et al., 2020)	FBank	DNN code-switching acoustic data augmentation	86.10 %

Hindi speech sample(Maurya et al., 2018)	MFCC	MFCC-VQ	94.12 %
Voxceleb(An et al., 2019)	FBank	CNN ResNet-18+Self-Attention	96.50 %
Urban- Sound8K dataset(Salamon & Bello, 2017)		SB-CNN	85.00 %
Indonesian Etnics Speaker(Riset Ini)	**MFCC**	**DA-DNN7L**	**99.76%**

Perbandingan kinerja speaker recognition diatas menunjukkan metode DA-DNN7L memiliki kinerja yang lebih baik jika dibandingkan dengan beberapa penelitian sejenis dengan menggunakan DNN maupun CNN (Convolutional Neural Network). Dalam tabel 1 bisa dilihat metode yang diusulkan dengan metode fitur extrksi MFCC memiliki tingkat akurasi yang melebihi kinerja model lain seperti MFCC-VQ maupun CNN yang mempergunakan Fbank sehingga metode yang diusulkan termasuk metode yang Robust dalam speaker recognition.

BAB 4

PENGENALAN UJARAN PALSU MENGGUNAKAN DEEP NEURAL NETWORK

4.1 Ujaran Palsu (Fake Speech)

Deteksi ujaran palsu adalah proses mengidentifikasi informasi yang salah, menyesatkan, atau dibuat-buat untuk disebarkan kepada publik dengan tujuan tertentu, seperti memanipulasi opini atau menciptakan kebingungan. Dalam konteks media sosial dan platform daring, ujaran palsu dapat berupa berita palsu (*fake news*), rumor, atau hoaks yang disebarkan melalui berbagai saluran.

Berikut adalah penjelasan lengkap dan komprehensif mengenai deteksi ujaran palsu:

1. Definisi Ujaran Palsu (Fake News)

   Ujaran palsu (fake news) merujuk pada informasi yang sengaja disebarkan dengan tujuan menyesatkan, mengecoh, atau memanipulasi audiens. Berita palsu ini dapat berbentuk artikel berita, gambar, video, atau meme yang dirancang untuk menarik perhatian dan mempengaruhi opini publik. Biasanya, ujaran palsu beredar dengan cepat melalui media sosial dan platform berita online.

2. Jenis-Jenis Ujaran Palsu

   a. Hoaks: Informasi palsu yang disebarkan untuk menciptakan kepanikan atau ketakutan di masyarakat.

   b. Clickbait: Artikel atau judul yang dirancang untuk menarik perhatian pembaca tanpa mempertimbangkan kebenaran kontennya.

   c. Misinformasi: Informasi yang salah namun disebarkan tanpa niat untuk menipu atau merugikan orang lain.

   d. Disinformasi: Informasi palsu yang sengaja disebarkan dengan niat untuk menyesatkan atau memanipulasi.

   e. Deepfakes: Video atau audio yang dimanipulasi menggunakan teknologi AI untuk meniru suara atau wajah seseorang, sehingga sulit dibedakan antara yang asli dan palsu.

3. Penyebab dan Dampak Penyebaran Ujaran Palsu

Penyebab:

1) Keinginan untuk memperoleh keuntungan ekonomi: Banyak situs web yang menghasilkan pendapatan melalui klik dan iklan yang terkait dengan artikel berita palsu.
2) Manipulasi politik: Ujaran palsu sering digunakan untuk mempengaruhi opini publik dalam konteks pemilu atau keputusan politik.
3) Sensasi dan kontroversi: Isu yang kontroversial cenderung lebih menarik bagi pembaca, dan ujaran palsu yang menyentuh tema sensitif seperti agama, ras, atau kebijakan pemerintah sering kali lebih viral.

Dampak:

1) Menurunkan kredibilitas media: Ketika berita palsu tersebar luas, kepercayaan publik terhadap media konvensional bisa terganggu.
2) Mengarah pada perpecahan sosial: Ujaran palsu dapat memicu ketegangan sosial, diskriminasi, atau kebencian antar kelompok.
3) Mengganggu proses demokrasi: Manipulasi opini publik dapat merusak hasil pemilu atau keputusan politik yang penting.
4) Penyebaran ketakutan atau kepanikan: Hoaks tentang bencana alam atau krisis kesehatan dapat menyebabkan kepanikan yang tidak perlu di masyarakat.

4. Teknik Deteksi Ujaran Palsu

Deteksi ujaran palsu melibatkan berbagai pendekatan yang menggunakan analisis data, pemrosesan bahasa alami (NLP), dan teknik kecerdasan buatan (AI). Beberapa metode yang digunakan untuk mendeteksi ujaran palsu antara lain:

a. Pendekatan Berbasis Aturan:

   Menggunakan aturan atau template untuk mencari pola yang sering muncul dalam berita palsu, seperti penggunaan bahasa yang sensasional, sumber yang tidak kredibel, atau ketidakcocokan informasi.

b. Pendekatan Berbasis Pembelajaran Mesin (Machine Learning):

1) Klasifikasi Teks: Menggunakan algoritma seperti Naive Bayes, SVM (Support Vector Machine), atau Random Forest untuk mengklasifikasikan berita sebagai palsu atau asli berdasarkan fitur-fitur seperti kata kunci, struktur kalimat, atau sumber informasi.

2) Pembelajaran Dalam (Deep Learning): Model seperti Convolutional Neural Networks (CNN) atau Recurrent Neural Networks (RNN), termasuk Long Short-Term Memory (LSTM), digunakan untuk mengenali pola dalam data teks atau video dan mengidentifikasi apakah informasi tersebut dapat dianggap sebagai ujaran palsu.

3) Model Pembelajaran Terbimbing (Supervised Learning): Melatih model dengan dataset yang telah diberi label (berita asli atau palsu) untuk mengajarkan model mengenali pola dalam data.

c. Analisis Jaringan Sosial:

1) Menggunakan analisis graf atau jaringan untuk melihat bagaimana berita palsu menyebar di media sosial. Analisis ini melibatkan identifikasi penyebar utama dan pola penyebaran berita palsu.

2) Pendeteksian Bot: Mengidentifikasi akun otomatis atau bot yang mungkin terlibat dalam menyebarkan berita palsu secara massal.

d. Verifikasi Sumber dan Fakta:

1) Pencocokan Fakta (Fact-checking): Platform seperti Snopes, FactCheck.org, atau PolitiFact digunakan untuk memverifikasi klaim-klaim yang muncul di berita atau media sosial.

2) Pendeteksian Sumber yang Tidak Terpercaya: Mengidentifikasi apakah berita tersebut berasal dari sumber yang terverifikasi atau memiliki reputasi buruk.

4.2 Deep Neural Network

Metode Deep Neural Network (DNN) dalam pengenalan suara adalah pendekatan berbasis *deep learning* yang menggunakan jaringan saraf tiruan dengan banyak lapisan tersembunyi (*hidden layers*) untuk mempelajari representasi kompleks dari sinyal audio. DNN bekerja dengan meniru cara otak manusia memproses informasi, di mana setiap lapisan mempelajari fitur yang semakin abstrak dan relevan untuk tugas pengenalan suara.

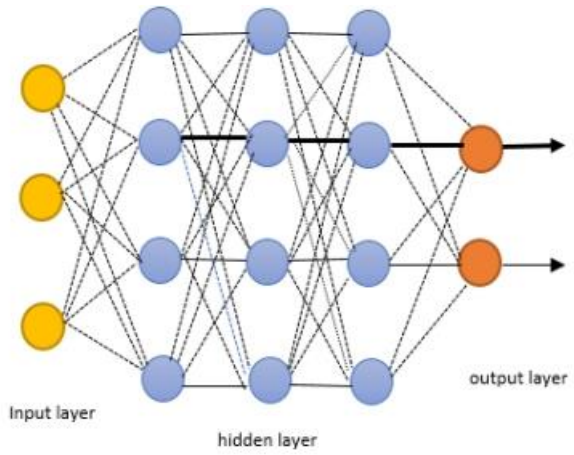
DNN dalam pengenalan suara umumnya terdiri dari:

a. Input Layer – Menerima vektor fitur (misalnya MFCC) dari sinyal suara.

b. Hidden Layers – Beberapa lapisan yang saling terhubung, dengan fungsi aktivasi non-linear seperti ReLU atau Sigmoid untuk mempelajari representasi tingkat tinggi.

c. Output Layer – Menghasilkan probabilitas untuk setiap unit suara (misalnya *phoneme*), biasanya menggunakan fungsi *softmax*.

Arsitektur DNN selengkapnya bsa dilihat pada gambar 10 sebagai berikut :

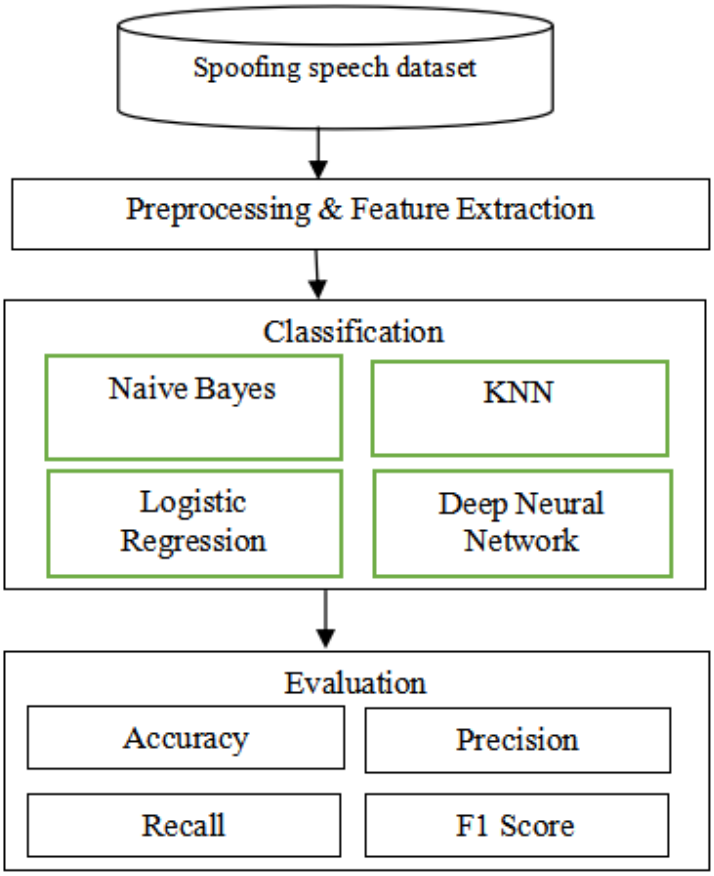


Gambar. 10 Arsiitektur DNN

Pada arsitektur DNN pada gambar 10 terdapat lapisan tersembunyi yang kedalamannya dapat mencapai n lapisan. DNN merupakan metode yang robust untuk diimplementasikan dalam berbagai permasalahan dalam visi komputer, termasuk di bidang klasifikasi citra, pengenalan suara, video, dan deteksi objek karena memiliki keunggulan dalam arsitektur lapisan pada jaringan saraf tiruan yang tersusun menjadi beberapa lapisan. Hal ini memungkinkan model menjadi lebih efisien dalam mempelajari fitur-fitur kompleks dan melakukan komputasi secara efektif pada beberapa operasi bersamaan.

4.3 Metode Penelitian

Penelitian yang dilakukan adalah experimental dimana tahapan penelitian tersaji pada gambar 11 sebagai berikut:



Gambar. 11 : Metode Penelitian

Tahapan penelitian yang ditunjukkan pada Gambar 11 menunjukkan proses pengenalan ucapan palsu dengan mengolah data melalui kegiatan preprocessing kemudian data terstruktur tersebut akan diklasifikasikan dengan empat pendekatan dimana 3 pendekatan tersebut merupakan metode Machine Learning klasik yaitu Naive Bayes, KNN, dan Logistic Regression sedangkan satu metode Deep Learning dengan DNN juga akan digunakan dalam proses klasifikasi penelitian ini.

4.4 Hasil Penelitian

Pada penelitian berbasis *machine learning* (ML), pengukuran akurasi merupakan langkah krusial dalam mengevaluasi performa model prediksi atau klasifikasi. Akurasi menjadi salah satu metrik evaluasi yang paling umum digunakan karena memberikan informasi tentang seberapa banyak prediksi model yang benar dibandingkan dengan total keseluruhan prediksi yang dilakukan.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}\ x\ 100\%$$

Dimana :

TP=True Positive FP=False Positive

TN=True Negative FN=False Negative

Pada penelitian yang sudah dilakukan diperoleh hasil pengukuran kinerja akurasi menggunakan algoritma Deep Neural Network yang dibandingkan dengan beberapa pendekatan metode machine learning lainnya seperti pada tabel 3 sebagai berikut:

Tabel 3. Perbandingan Kinerja Akurasi pada deteksi Fake Speaker

Metofr	Akurasi (%)
Naive Bayes(NB)	87.5
KNN	89.1
Logistic Regression(LR)	87.5
DNN	**96.5**

Tabel 3 menunjukkan bahwa metode DNN merupakan metode yang memberikan akurasi tertinggi dalam mengenali penutur dengan ucapan palsu dibandingkan metode lain seperti NB, KNN, dan LR. DNN memberikan hasil akurasi tinggi dengan penataan bentuk arsitektur model yang melibatkan 100 neuron pada lapisan tersembunyi menggunakan fungsi aktivasi Relu dan Adam Optimizer yang menyusun jaringan DNN.

Presisi adalah ukuran yang menunjukkan seberapa akurat suatu model atau sistem dalam mengidentifikasi prediksi positif yang benar dibandingkan dengan seluruh prediksi positif yang dihasilkan. Dalam konteks evaluasi kinerja, khususnya pada machine learning, presisi menjadi indikator penting ketika kesalahan memberikan prediksi positif palsu (false positive) memiliki dampak signifikan Presisi bisa dihitung menggunakan rumus sebagai berikut :

$$\text{Precision} = \frac{TP}{(TP+FP)} \; x \; 100\%$$

Pada penelitian yang dilakukan perhitungan pressisi dapat dilihat hasilnya seperti pada tabel 4

Tabel 4. Perbandingan Presisi deteksi ujaran palsu

Metode	Presisi (%)
Naive Bayes(NB)	76.5
KNN	94.2
Logistic Regression(LR)	76.5
DNN	**97.3**

Pada penelitian ini juga dihitung Recall untuk mengetahui sejauh mana model mampu menangkap semua data positif yang ada, sehingga membantu peneliti atau praktisi melakukan penyesuaian, seperti mengatur *threshold* atau memilih algoritma yang lebih sensitif terhadap data positif. Perhitungan ini memeprgunakan rumus sebagai berikut :

$$\text{Recall} = \frac{\text{TP}}{(\text{FN}+\text{TP})} * 100\%$$

Hasil perhitungan Recall pada penelitian ini disampaikan pada tabel 5 sebagai berikut :

Tabel 5. Perbandingan Recall metode deteksi ujaran palsu

Methods	Recall (%)
Naive Bayes(NB)	87.5
KNN	89.1
Logistic Regression(LR)	87.5
DNN	**96.5**

Kinerja model pengenalan pembicara dengan ucapan palsu juga diukur menggunakan skala F1 yang membandingkan rata-rata presisi dan perolehan yang telah dibobot sebelumnya. Hasil perhitungan F1 dapat menggunakan rumus berikut:

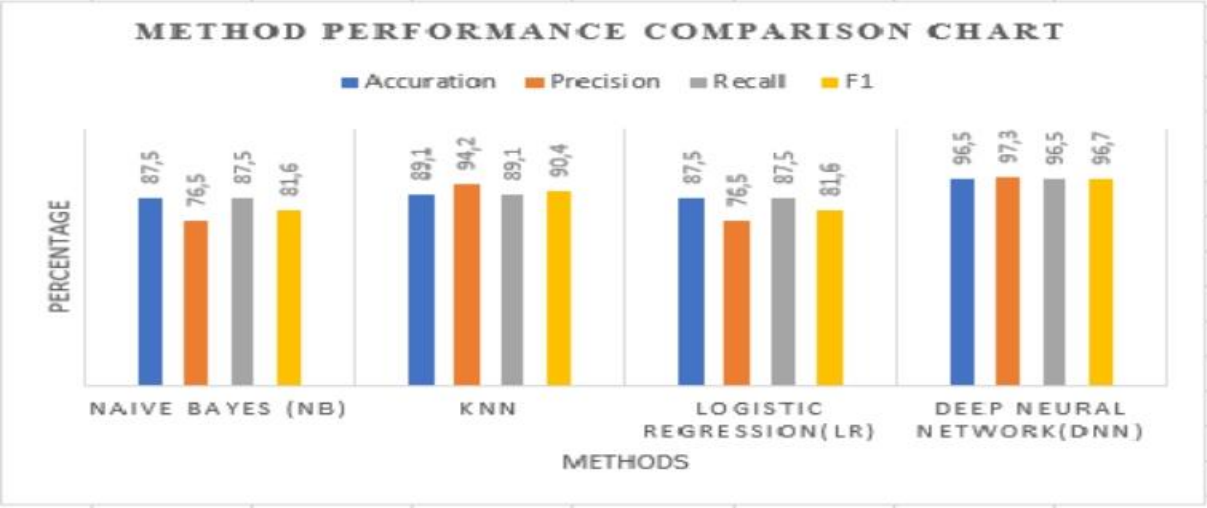
$$\text{F1 Score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

Hasil pengukuran kinerja F1 Score bisa dilihat selengkapnya pada tabel 6 sebagai berikut :

Tabel 6 Perbandingan F1 metode deteksi ujaran palsu

Methods	F1 Measure (%)
Naive Bayes(NB)	81.6
KNN	90.4
Logistic Regression(LR)	81.6
DNN	**96.7**

Pada tabel 6 terlihat jelas metode DNN merupakan metode dengan kinerj F1 score terbaik dibandingkan pendekatan lainnya, Secara keseluruhan kinerja model DNN pada deteksi ujaran palsu dapat dilihat pada gambar 12 seperti dibawah ini.



Gambar. 12 Grafik perbandingan kinerja model deteksi ujaran palsu

Jika diamati pada gambar 12 terlihat bahwa metode Deep neural Network (DNN) memiliki kinerja terbaik baik dari sisi Akurasi,Presisi,Recall maupun F1 Score jika dibandingkan dengan metode Machine Learning sederhana seperti Bayes,Logistic Regression maupun k-Nearest Neighbor (KNN) pada bidang riset pengenalan suara pada ujaran palsu.

BAB 5

DETEKSI PEMBICARA MENGGUNAKAN GENERATIVE ADVERSARIAL NETWORK

5.1 Generative Adversarial Network (GAN)

Generative Adversarial Networks (GANs) adalah salah satu inovasi terpenting dalam dunia kecerdasan buatan (AI) dan pembelajaran mesin (machine learning) yang pertama kali diperkenalkan oleh Ian Goodfellow dan tim pada tahun 2014. GANs adalah model pembelajaran yang digunakan untuk menghasilkan data baru yang menyerupai data asli, dengan cara yang sangat mirip dengan bagaimana manusia dapat mempelajari dan menghasilkan informasi berdasarkan pengalaman yang ada. Model ini telah mengubah cara kita berinteraksi dengan teknologi di berbagai bidang, mulai dari penciptaan gambar, musik, hingga teks.

Prinsip Kerja GAN :

GANs bekerja berdasarkan prinsip kompetisi antara dua jaringan saraf tiruan: *Generator* dan *Discriminator*. Keduanya dilatih secara bersamaan, dengan tujuan untuk meningkatkan kinerja masing-masing melalui proses yang disebut *adversarial training*.

1. Generator: Jaringan ini bertanggung jawab untuk menghasilkan data baru. Pada awalnya, generator membuat data yang acak (misalnya gambar) berdasarkan input acak (noise) yang diberikan kepadanya. Generator berusaha untuk menciptakan data yang semakin mirip dengan data asli yang ada dalam dataset.
2. Discriminator: Tugas discriminator adalah membedakan antara data nyata (yang berasal dari dataset) dan data palsu (yang dihasilkan oleh generator). Discriminator diberikan data asli dan data palsu dan dilatih untuk mengklasifikasikan apakah data tersebut asli atau palsu.

Proses Adversarial Training :

Proses pelatihan GAN adalah bentuk "permainan" antara generator dan discriminator. Generator berusaha untuk menipu discriminator dengan menghasilkan data yang sangat mirip dengan data asli, sedangkan discriminator berusaha untuk menjadi semakin baik

dalam membedakan antara data asli dan palsu. Dalam setiap iterasi, kedua jaringan ini saling melatih untuk memperbaiki kekurangan masing-masing.

1) Generator: Meningkatkan kemampuannya dalam menghasilkan data yang lebih realistis.

2) Discriminator: Meningkatkan kemampuannya dalam mengidentifikasi apakah data tersebut asli atau palsu.

   Melalui proses ini, keduanya "berkompetisi" sehingga kemampuan mereka terus meningkat. Seiring waktu, generator menjadi sangat terampil dalam menghasilkan data yang hampir tidak dapat dibedakan dari data asli.

GAN terdiri dari dua komponen utama:

1. Generator: Menghasilkan data baru berdasarkan input noise acak.
2. Discriminator: Menilai apakah data yang diberikan adalah data asli atau data yang dihasilkan oleh generator.

Sejak diperkenalkan, berbagai varian GAN telah dikembangkan untuk menangani berbagai jenis masalah dan aplikasi. Beberapa di antaranya adalah:

1. DCGAN (Deep Convolutional GAN): Merupakan modifikasi GAN yang menggunakan *convolutional neural networks* (CNN) untuk meningkatkan kualitas gambar yang dihasilkan.
2. CGAN (Conditional GAN): Memperkenalkan kondisi atau label pada input untuk menghasilkan data berdasarkan kondisi tertentu, seperti menghasilkan gambar berdasarkan deskripsi teks.
3. WGAN (Wasserstein GAN): Menggunakan fungsi kerugian yang lebih stabil dan mengatasi beberapa masalah yang ada pada GAN tradisional, seperti mode collapse.
4. StyleGAN: Dikembangkan untuk menghasilkan gambar wajah manusia yang sangat realistis dan banyak digunakan dalam pembuatan gambar wajah palsu (deepfakes).
5. CycleGAN: Mengizinkan konversi citra antara dua domain yang berbeda tanpa membutuhkan pasangan data yang terlabelkan.

Keunggulan GAN

1. Kreativitas dalam Penciptaan Konten: GAN dapat menghasilkan gambar, musik, atau teks yang tidak hanya baru, tetapi juga kreatif. Misalnya, GAN dapat digunakan untuk menghasilkan karya seni atau desain baru yang menyerupai karya terkenal.
2. Peningkatan Kualitas Data Sintetis: GAN dapat digunakan untuk menghasilkan data sintetis yang digunakan dalam pelatihan model pembelajaran mesin, khususnya ketika data asli sulit diperoleh.
3. Aplikasi dalam Industri: GAN banyak digunakan dalam industri seperti hiburan, medis, otomotif, dan e-commerce untuk meningkatkan pengalaman pengguna atau menciptakan konten visual yang menarik.

5.2 Arsitektur GAN

Generative Adversarial Networks (GAN) adalah salah satu metode dalam pembelajaran mesin yang digunakan untuk menghasilkan data yang sangat realistis dan mirip dengan data asli, baik dalam bentuk gambar, suara, atau teks. GAN diperkenalkan oleh Ian Goodfellow dan timnya pada tahun 2014. Metode ini mengandalkan dua model neural network yang saling berkompetisi, yaitu **Generator** dan **Discriminator**. Kombinasi dari dua jaringan ini memungkinkan GAN untuk belajar menghasilkan data yang tidak hanya baru, tetapi juga berkualitas tinggi, seringkali sulit dibedakan dari data asli.

Proses Pelatihan GAN :

Pelatihan GAN dilakukan dalam bentuk proses yang disebut *adversarial training*, yang melibatkan dua langkah kompetitif yang saling berinteraksi:

1) Generator berusaha untuk memperbaiki kemampuannya dalam menghasilkan data palsu yang semakin menyerupai data asli. Tujuannya adalah agar data yang dihasilkannya sulit dibedakan dari data asli.
2) Discriminator, di sisi lain, berusaha untuk lebih baik dalam membedakan antara data asli dan palsu yang dihasilkan oleh generator.

Kedua model ini saling bersaing dalam permainan minimax, di mana generator berusaha untuk mengelabui discriminator, sementara discriminator berusaha menjadi semakin pintar dalam membedakan data palsu dan asli. Proses ini dapat digambarkan dengan dua fungsi biaya yang saling bertujuan:

- ✓ Discriminator berusaha memaksimalkan kemampuannya dalam memisahkan data asli dan palsu.
- ✓ Generator berusaha memaksimalkan kemampuannya dalam menghasilkan data yang berhasil melewati discriminator.

Perhitungan GAN

Matematika di balik GAN dapat dijelaskan menggunakan konsep *game theory*, di mana dua agen (generator dan discriminator) memiliki tujuan yang berlawanan:

- ✓ Fungsi objektif untuk generator adalah meminimalkan kemungkinan bahwa discriminator dapat mengenali bahwa outputnya adalah palsu.
- ✓ Fungsi objektif untuk discriminator adalah memaksimalkan kemampuannya untuk membedakan antara data nyata dan data palsu.

Fungsi yang digunakan dalam GAN adalah:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

di mana:

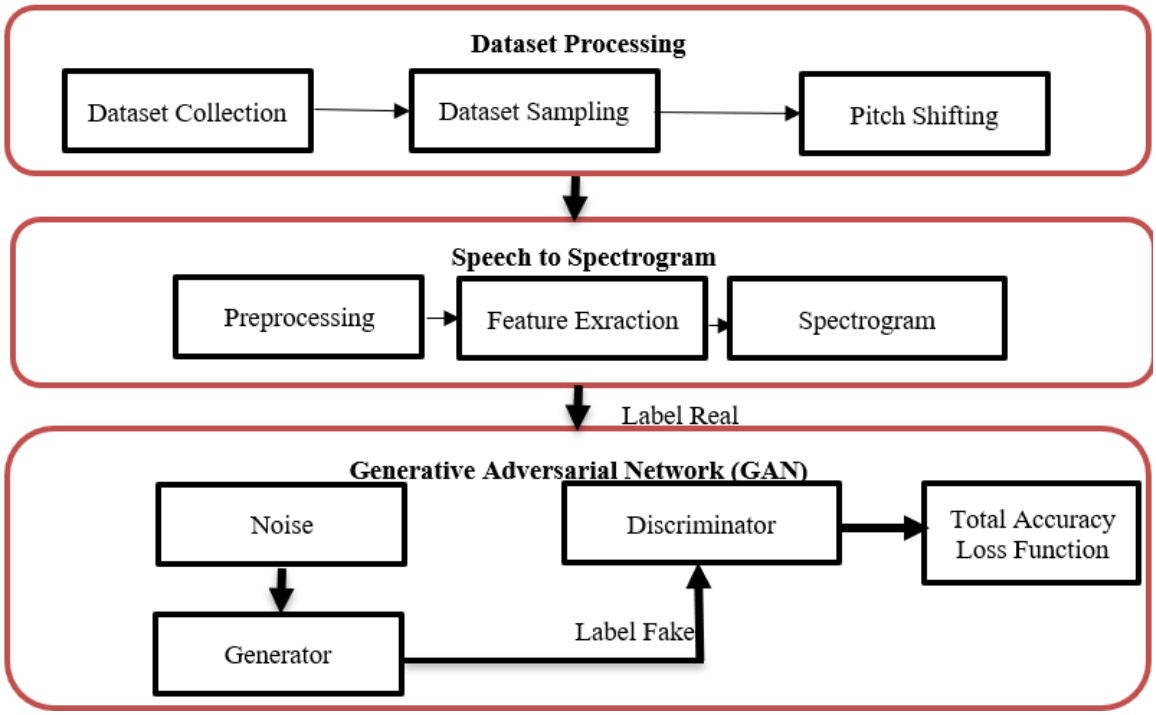
$D(x)$ adalah probabilitas bahwa data xxx adalah asli.

$G(z)$ adalah data yang dihasilkan oleh generator G dari vektor acak z

$p_{data}(x)$ adalah distribusi data asli.

$p_z(z)$ adalah distribusi vektor acak yang digunakan untuk menghasilkan data palsu.

5.3 Metode Penelitian



Gambar. 13 Metode penelitian pitch shifting GAN

Tahapan penelitian yang dilakukan antara lain adalah :

1. Pengumpulan dataset (dataset collection) dan data sampling

   Penelitian mengenai deteksi suara ini menggunakan dataset ujaran (suara) pada penelitian pengenalan penutur multietnis di Indonesia, yaitu dataset dari puluhan ujaran etnis di Indonesia. Namun, penelitian ini akan mengambil dataset dari 10 etnis yang mewakili suku di Jawa dan luar Jawa. Kemudian, sinyal ujaran multietnis tersebut harus dilakukan proses sampling ujaran selama 2 detik untuk memilih penutur yang terdiri dari suku Banda Aceh, Padang, Ambon, Bali Tabanan, Banten, Banyumas, Betawi, Jawa Timur, Yogyakarta, dan Madura. Setelah dilakukan pengumpulan hasil sampling ujaran tersebut ternyata memiliki jumlah suara yang bervariasi, sehingga ditentukan setiap suku akan diisi dengan sepuluh sampel suara.

2. Pitch shifting

   Pitch shifting adalah teknik pengubahan tinggi nada (*pitch*) dari suatu sinyal audio tanpa mengubah durasi temporalnya. Dalam konteks *data augmentation* suara, pitch shifting

digunakan untuk menghasilkan variasi data pelatihan dengan memanipulasi frekuensi fundamental (F0) sehingga suara terdengar lebih tinggi atau lebih rendah, namun tetap mempertahankan kecepatan dan panjang durasi aslinya. Teknik ini berbeda dengan **time** stretching, yang mengubah kecepatan suara tetapi mempertahankan pitch. Pitch shifting fokus pada pergeseran frekuensi tanpa mempercepat atau memperlambat audio.

Algoritma pseudocode pitch shifting adalah sebagai berikut :

1. Impor pustaka pemrosesan audio yang diperlukan
2. shifted_audio = audio_processing_library.pitch_shift(audio_sample, semitone_shift)
3. Kembalikan shifted_audio
4. min_semitone_shift = -2
5. max_semitone_shift = 2
6. audio_samples = load_audio_dataset()
7. augmented_data = []
8. Untuk setiap audio_sample di audio_samples:
9. random_semitone_shift = random_choice(min_semitone_shift, max_semitone_shift)
10. shifted_audio = pitch_shift(audio_sample, random_semitone_shift)
11. augmented_data.append(shifted_audio)
12. Simpan_atau_gunakan_augmented_data(augmented_data)

Sedangkan jika diimplementasikan pada pemrograman python bisa dipergunakan script sebagai berikut :

```
import librosa

import soundfile as sf

# Load audio

y, sr = librosa.load('audio.wav', sr=None)

# Pitch shift naik 2 semitone

y_shifted = librosa.effects.pitch_shift(y, sr, n_steps=2)
```

 # Simpan hasil

sf.write('audio_pitch_up.wav', y_shifted, sr)

3. Konversi sinyal (Speech to spectreogram)

Konversi sinyal suara menjadi spektrum adalah proses yang sangat penting dalam analisis sinyal, terutama dalam berbagai aplikasi seperti pengolahan bahasa alami (NLP), pengenalan suara, dan bahkan musik. Proses ini mengubah sinyal suara yang bersifat waktu menjadi representasi frekuensi yang menggambarkan bagaimana frekuensi suara tersebut berubah seiring waktu.

✓ Definisi Spektrogram

Spektrogram adalah representasi visual dari spektrum frekuensi suatu sinyal suara seiring waktu. Pada spektrogram, sumbu horizontal mewakili waktu, sumbu vertikal mewakili frekuensi, dan intensitas warna atau kecerahan menggambarkan amplitudo atau kekuatan sinyal pada frekuensi tertentu pada waktu tertentu.

✓ Langkah-Langkah Konversi Sinyal Suara ke Spektrogram

Proses konversi dimulai dengan menangkap sinyal suara, biasanya dalam bentuk sinyal analog yang kemudian diubah menjadi sinyal digital, dan selanjutnya diproses menggunakan transformasi matematika yang kompleks. Berikut adalah langkah-langkah utama dalam proses tersebut:

1) Pengambilan Sinyal Suara

Sinyal suara pertama-tama direkam menggunakan mikrofon yang menghasilkan sinyal analog. Sinyal ini kemudian dikonversi menjadi format digital menggunakan proses yang dikenal dengan *sampling*. Pada tahap ini, suara direkam dengan frekuensi tertentu (misalnya 44.1 kHz) untuk memperoleh representasi digital dari gelombang suara.

2) Pembagian Sinyal Menjadi Jendela-Jendela Kecil

Karena sinyal suara bersifat kontinu dan berubah-ubah, kita perlu memecah sinyal tersebut menjadi bagian-bagian kecil yang lebih mudah dianalisis. Proses ini dikenal dengan *windowing*. Pada tahap ini, sinyal dibagi menjadi beberapa *frame* atau

jendela waktu yang lebih kecil, biasanya antara 10 ms hingga 50 ms. Setiap jendela ini akan dianalisis untuk mengetahui spektrum frekuensinya.

3) Penerapan Transformasi Fourier

Setiap frame yang telah dibagi akan dianalisis menggunakan *Transformasi Fourier* (biasanya *Fast Fourier Transform*, FFT), yang mengubah sinyal dari domain waktu menjadi domain frekuensi. FFT memecah sinyal suara menjadi komponen-komponen frekuensinya, menghasilkan informasi tentang amplitudo dan fasa dari berbagai frekuensi yang ada pada setiap titik waktu dalam jendela tersebut.

Hasil dari proses FFT ini adalah serangkaian nilai kompleks yang dapat dihitung magnitudonya untuk menggambarkan kekuatan atau intensitas dari setiap frekuensi dalam jendela waktu yang bersangkutan. Nilai magnitudo ini akan menjadi elemen utama dalam membangun spektrogram.

4. Metode GAN

Generative Adversarial Networks (GANs) telah merevolusi berbagai bidang dalam kecerdasan buatan, termasuk pengolahan suara. Pada dasarnya, GAN adalah jaringan syaraf tiruan yang terdiri dari dua komponen utama: generator dan discriminator. Kedua komponen ini berinteraksi dalam suatu proses permainan zero-sum, di mana generator berusaha untuk menghasilkan data yang mirip dengan data asli, sedangkan discriminator berusaha membedakan data yang dihasilkan oleh generator dengan data asli. Dalam konteks pengenalan suara, GAN memiliki potensi besar untuk meningkatkan kualitas dan akurasi sistem pengenalan suara melalui berbagai pendekatan inovatif.

Penerapan GAN dalam Pengenalan Suara

Pengenalan suara melibatkan konversi suara menjadi bentuk teks atau data yang dapat dimengerti oleh sistem komputer. Salah satu tantangan utama dalam pengenalan suara adalah mengatasi variabilitas dalam suara manusia, seperti aksen, intonasi, atau kualitas rekaman suara. GAN dapat digunakan untuk mengatasi tantangan ini dengan cara menghasilkan data suara sintetis yang memperkaya variasi suara yang digunakan untuk melatih model pengenalan suara.

Beberapa aplikasi GAN dalam pengenalan suara antara lain:

1. Peningkatan Kualitas Suara

Salah satu aplikasi GAN yang paling umum dalam pengenalan suara adalah peningkatan kualitas suara. GAN dapat digunakan untuk menghilangkan noise pada rekaman suara, memperbaiki distorsi, atau meningkatkan kualitas rekaman yang buruk, sehingga sistem pengenalan suara dapat lebih mudah mengenali kata-kata yang diucapkan.

2. Data Augmentasi untuk Pelatihan Model

   GAN dapat digunakan untuk menghasilkan data suara sintetis yang menyerupai suara manusia dalam berbagai variasi. Data ini dapat digunakan untuk melatih model pengenalan suara, yang dapat meningkatkan kinerja model dalam menghadapi berbagai aksen, kebisingan latar belakang, dan variasi lainnya. Augmentasi data menggunakan GAN membantu meningkatkan robustitas dan generalisasi model pengenalan suara.

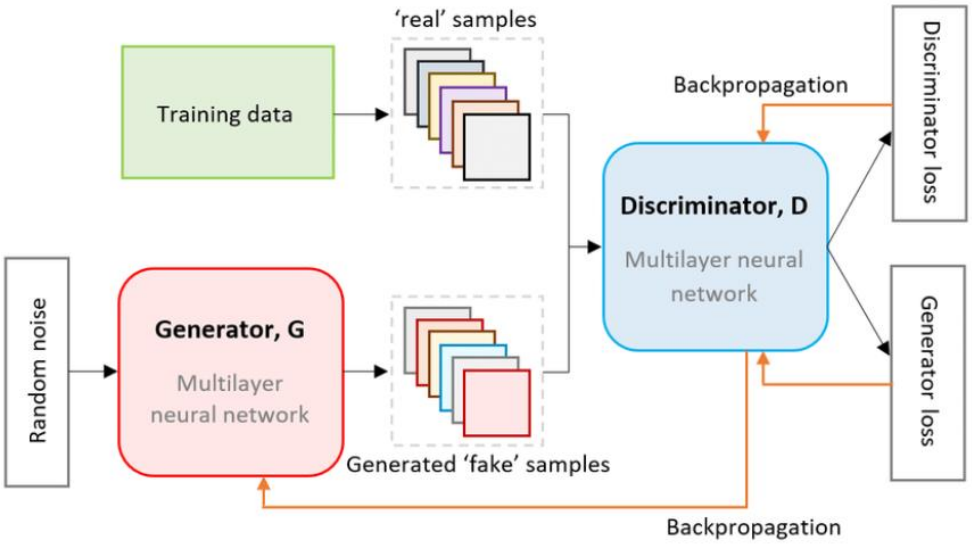
3. Pemodelan dan Sintesis Suara

   GAN juga digunakan untuk sintesis suara, di mana suara sintetis yang dihasilkan oleh generator digunakan untuk melatih model pengenalan suara. Teknologi ini dapat menghasilkan suara yang menyerupai suara manusia secara alami, bahkan mampu meniru berbagai aksen dan gaya bicara yang berbeda.

4. Pengenalan Suara pada Latar Belakang Bising

   GAN dapat digunakan untuk mempelajari pola suara dalam kondisi bising dan meningkatkan akurasi sistem pengenalan suara dalam lingkungan yang penuh dengan gangguan. Dengan menghasilkan data suara sintetis yang mencakup latar belakang bising, model GAN membantu sistem pengenalan suara menjadi lebih tangguh terhadap suara-suara yang mengganggu.

Secara umum arsitektur GAN(Little et al., 2021) bisa terlihat pada gambar 14 sebagai berikut:



Gambar. 14 Arsitektur umum metode GAN

Pada gambar 14 terlihat bahwa Generative Adversarial Network (GAN) bekerja berdasarkan arsitektur yang terdiri dari dua model neural network yang saling berlawanan, yaitu **Generator** dan **Discriminator**, yang dilatih secara bersamaan dalam proses kompetitif. Generator bertugas menghasilkan data sintetis (misalnya gambar, suara, atau teks) yang meniru distribusi data asli, dimulai dari input acak (noise vector) yang diubah menjadi output menyerupai data nyata. Sementara itu, Discriminator berfungsi sebagai pengklasifikasi yang membedakan antara data asli dari dataset dan data palsu hasil Generator. Proses pelatihan mengikuti pendekatan minimax game: Generator berusaha meminimalkan kemampuan Discriminator untuk membedakan data palsu, sedangkan Discriminator berusaha memaksimalkan akurasi deteksinya. Pelatihan berlangsung iteratif—Generator mengirim data palsu, Discriminator mengevaluasi, lalu error dikembalikan (backpropagation) ke masing-masing model untuk memperbaiki parameter. Seiring waktu, Generator menjadi semakin mahir menghasilkan data yang realistis, sementara Discriminator menjadi semakin sulit membedakan data asli dan palsu, hingga tercapai

5. Evaluasi Model

Evaluasi model machine learning adalah langkah krusial untuk menentukan seberapa baik model yang dibangun dalam memecahkan masalah yang dihadapi. Dua metrik evaluasi yang

paling umum digunakan adalah accuracy dan loss function. Kedua metrik ini membantu dalam menilai kinerja model, meskipun keduanya mengukur hal yang sedikit berbeda.

1. Accuracy: Mengukur Proporsi Prediksi yang Benar

Accuracy adalah metrik evaluasi yang paling sederhana dan paling sering digunakan untuk model klasifikasi. Metrik ini mengukur proporsi prediksi yang benar dibandingkan dengan total prediksi yang dilakukan oleh model.

Rumus:

$$\text{Accuracy} = \frac{\text{Jumlah Prediksi Benar}}{\text{Jumlah Total Data}}$$

Misalnya, jika model klasifikasi memprediksi 80 dari 100 data dengan benar, maka accuracy-nya adalah 80%. Accuracy memberikan gambaran umum mengenai seberapa banyak prediksi model yang benar dibandingkan dengan seluruh data.

Kelebihan:

- ✓ Mudah dipahami dan diinterpretasikan.
- ✓ Cocok untuk dataset yang seimbang, di mana jumlah data pada setiap kelas relatif sama.

Kekurangan:

Accuracy bisa menyesatkan ketika dataset tidak seimbang (imbalanced dataset). Misalnya, dalam kasus klasifikasi biner di mana satu kelas jauh lebih banyak daripada kelas lainnya, model yang selalu memprediksi kelas mayoritas bisa mendapatkan accuracy yang tinggi meskipun model tersebut tidak berguna. Dalam situasi seperti ini, metrik lain seperti precision, recall, dan F1-score lebih informatif.

2. Loss Function: Mengukur Kualitas Prediksi

Loss function adalah ukuran yang digunakan untuk mengevaluasi kesalahan model dalam membuat prediksi. Berbeda dengan accuracy yang hanya fokus pada hasil prediksi yang benar atau salah, loss function menghitung seberapa besar perbedaan antara prediksi model dan nilai sebenarnya (ground truth). Loss function memberikan penalti yang lebih besar untuk prediksi yang jauh dari nilai yang benar, memungkinkan model untuk mengoptimalkan diri dan meminimalkan kesalahan.

Jenis Loss Function:

Terdapat berbagai jenis loss function yang digunakan bergantung pada tipe masalah yang ingin diselesaikan, seperti:

A. Mean Squared Error (MSE): Digunakan dalam masalah regresi. MSE menghitung rata-rata kuadrat selisih antara nilai prediksi dan nilai sebenarnya.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Di mana y_i adalah nilai sebenarnya dan $\hat{y}_i$ adalah nilai prediksi.

B. Binary Cross-Entropy Loss: Digunakan dalam masalah klasifikasi biner. Cross-entropy menghitung seberapa besar perbedaan antara distribusi probabilitas yang diprediksi oleh model dan distribusi probabilitas yang sebenarnya.

$$\text{Binary Cross-Entropy Loss} = -\frac{1}{n}\sum_{i=1}^{n}(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i))$$

Di mana y_i adalah label sebenarnya (0 atau 1), dan $\hat{y}_i$ adalah probabilitas prediksi.

C. Categorical Cross-Entropy Loss: Digunakan dalam masalah klasifikasi multikelas, mengukur kesalahan antara distribusi probabilitas dari model dan nilai sebenarnya dalam bentuk one-hot encoding.

$$\text{Categorical Cross-Entropy Loss} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C} y_{ic}\log(\hat{y}_{ic})$$

Di mana C adalah jumlah kelas y_{ic} adalah label sebenarnya (1 untuk kelas yang benar dan 0 untuk kelas lainnya), dan $\hat{y}_{ic}$ adalah probabilitas prediksi untuk kelas c.

Kelebihan:

- ✓ Memberikan wawasan lebih mendalam mengenai bagaimana model berperforma pada setiap prediksi, bahkan ketika model membuat kesalahan kecil.
- ✓ Dapat digunakan untuk mengoptimalkan model dengan lebih baik, karena model dapat belajar untuk mengurangi loss selama proses pelatihan.

Kekurangan:

- ✓ Tidak mudah dipahami oleh orang yang tidak berpengalaman dalam machine learning karena sifatnya yang lebih matematis.
- ✓ Dapat menjadi lebih besar dan lebih sensitif terhadap outlier atau nilai yang jauh dari prediksi model, terutama pada MSE.

5.4 Hasil Penelitian

1. Pemrosesan Dataset

Penelitian ini memanfaatkan dataset yang mencakup 10 etnis, mewakili suku-suku di Jawa maupun di luar Jawa. Proses pengambilan sampel dilakukan pada sinyal ujaran multietnis dengan durasi 2 detik untuk memilih penutur yang berasal dari Banda Aceh, Padang, Ambon, Bali, Tabanan, Banten, Banyumas, Betawi, Jawa Timur, Yogyakarta, dan Madura. Setelah pengumpulan data selesai, ditemukan bahwa jumlah sampel suara yang diperoleh bervariasi. Oleh karena itu, setiap kelompok etnis ditetapkan memiliki sepuluh sampel suara, dan apabila terdapat kekurangan, jumlahnya akan dilengkapi menggunakan teknik augmentasi data dengan metode pergeseran nada.

2.Pitch Shifing

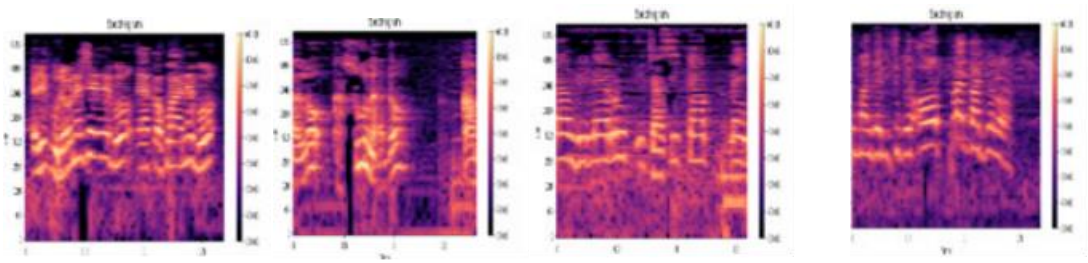
Pitch Shifting adalah salah satu teknik dalam pengolahan sinyal suara yang digunakan untuk mengubah tinggi nada (pitch) suatu sinyal audio tanpa mempengaruhi kecepatan atau durasi ucapannya. Dalam konteks pengenalan suara, metode ini sering dimanfaatkan sebagai bagian dari data augmentation untuk memperbanyak variasi data latih. Secara teknis, Pitch Shifting bekerja dengan memodifikasi frekuensi fundamental dan harmonik dari gelombang suara. Misalnya, suara penutur dapat dinaikkan nadanya sehingga terdengar seperti memiliki intonasi lebih tinggi, atau diturunkan nadanya agar terdengar lebih rendah. Perubahan ini memberikan variasi alami pada karakteristik suara tanpa mengubah kata atau kalimat yang diucapkan. Penerapan Pitch Shifting dalam pengenalan suara sangat bermanfaat untuk mengatasi keterbatasan jumlah dataset. Dengan menambahkan variasi pitch pada sampel suara, sistem pengenalan dapat belajar mengenali pola fonetik dari berbagai kemungkinan intonasi penutur. Hal ini membantu model menjadi lebih robust

terhadap perbedaan suara antar individu, baik dari segi gender, usia, maupun aksen. Algoritma pitch shifting sendiri bisa dilihat pada tampilan sebagai berikut :

1. Import the necessary audio processing library
2. shifted_audio = audio_processing_library.pitch_shift(audio_sample, semitone_shift)
3. Return shifted_audio
4. min_semitone_shift = -2
5. max_semitone_shift = 2
6. audio_samples = load_audio_dataset()
7. augmented_data = []
8. For each audio_sample in audio_samples:
9. random_semitone_shift = random_choice(min_semitone_shift, max_semitone_shift)
10. shifted_audio = pitch_shift(audio_sample, random_semitone_shift)
11. augmented_data.append(shifted_audio)
12. Save_or_use_augmented_data(augmented_data)

4. Speech to Spectrogram

   Tahapan selanjutnya Adalah melakukan konversi sinyal ujaran (speech) ke bentuk spectrogram suara. Sinyal suara yang telah dihasilkan kemudian dikonversi dalam bentuk spectrogram suara yang secara visual contoh tampilannya terlihat pada gambar 15 sebagai berikut :

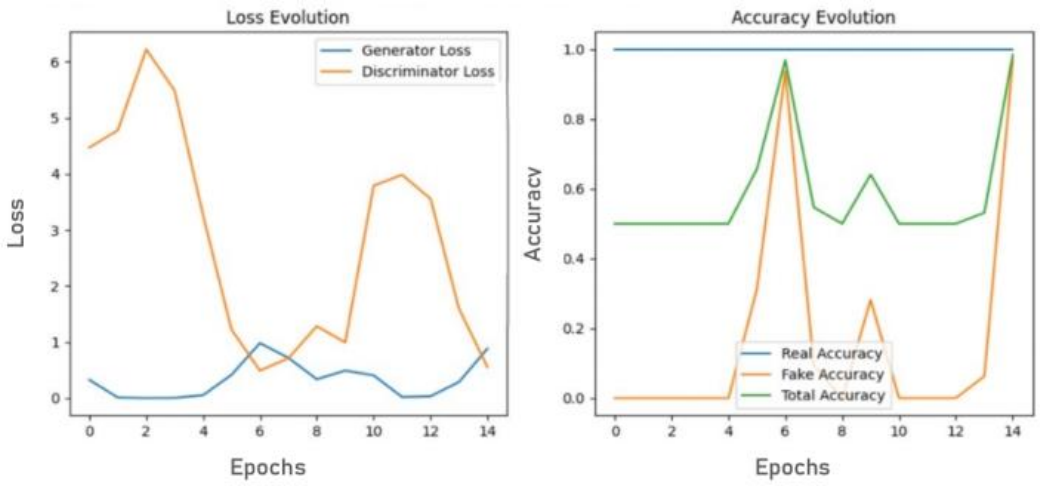


Bali Tabanan Padang Jawa Timur Betawi

Gambar. 15 Sinyal spectrogram suara

5. Hasil Uji Model

Penelitian yang telah dilakukan mengahsilkan suatu pemodelan untuk mendeteksi ujaran suara etnis berbasis spectrogram. Model tersebut kemudian diuji coba kinerjanya dengan hasil pengukuran yang bisa dilihat pada gambar 16 sebagai berikut :



Gambar. 16 Hasil uji coba kinerja model pengenalan suara

Jika kita melihat gambar 16 ditunjukkan bahwa nilai pada generator loss dan discriminator loss mendekati 0, yang mengindikasikan bahwa generator mampu menghasilkan model dengan kualitas lebih baik. Dari sisi akurasi keseluruhan, model ini juga menunjukkan kinerja yang sangat tinggi dengan capaian hampir 100%. Berdasarkan penelitian yang dilakukan, metode yang diusulkan terbukti mampu mencapai tingkat akurasi tinggi dalam mengenali spektrogram ucapan dari penutur multietnis. Pencapaian ini melampaui hasil penelitian lain di bidang pengenalan ucapan yang turut menggunakan pendekatan GAN. Perbandingan akurasi antara penelitian ini dan penelitian lain dapat dilihat pada Tabel 7.

Tabel 7. Perbandingan Kinerja Model

Metode	Dataset	Akurasi (%)
Power spectral density-generative adversarial network (PSD-GAN) [22]	MAHNOB-HCI	70.34%
Convolutional neural networks spectrogram image features (CNN-SIFs) [27]	Traffic sound datasets (TSD)	97.18%

Spectrogram deep convolutional generative adversarial network (S-DCGAN) [28]	Power Spectral Density	91.25%
Proposed Method (PS-GAN)	Indonesian Multi-etnics Speech	98.43%

BAB 6

RNN DAN LSTM PADA DATA BERBASIS TEKS

6.1 Reccurent Neural Network (RNN)

Recurrent Neural Network (RNN) adalah jenis jaringan saraf tiruan yang dirancang untuk memproses data berurutan (sequential data), seperti teks, urutan waktu (time series), dan, yang relevan dengan topik ini, pengenalan suara. RNN memiliki kemampuan untuk memori internal, yang memungkinkan model ini untuk "menghafal" informasi dari urutan input sebelumnya. Ini menjadikannya sangat efektif dalam menangani tugas-tugas yang melibatkan data yang memiliki dependensi temporal, seperti pengolahan bahasa alami, analisis urutan sinyal, dan pengenalan suara.

RNN berbeda dengan model jaringan saraf feedforward karena ia memiliki loop yang memungkinkan informasi dari langkah waktu sebelumnya untuk digunakan dalam langkah berikutnya. Sebagai hasilnya, RNN mampu memproses urutan data, menyimpan informasi tentang urutan sebelumnya dalam "memori" jangka pendek, dan menggunakan informasi ini untuk memprediksi elemen berikutnya dalam urutan tersebut. Hal ini sangat penting dalam pengenalan suara, di mana suara yang diucapkan bersifat temporal dan tergantung pada urutan suara sebelumnya.

Pengenalan suara adalah proses konversi sinyal suara (umumnya dalam bentuk gelombang audio) menjadi bentuk teks atau perintah yang dapat dipahami oleh mesin. Dalam tugas pengenalan suara, sinyal suara sering kali diubah menjadi fitur-fitur berbasis waktu, seperti Mel-Frequency Cepstral Coefficients (MFCC), yang merepresentasikan informasi penting dari spektrum suara.

Langkah-langkah dasar dalam pengenalan suara dengan RNN adalah sebagai berikut:

1) Ekstraksi Fitur: Suara yang diterima dikonversi menjadi representasi fitur berbasis waktu, misalnya menggunakan teknik seperti FFT (Fast Fourier Transform) atau MFCC. Fitur-fitur ini menggambarkan sifat dasar dari suara dalam bentuk numerik.

2) Proses RNN: Fitur-fitur yang dihasilkan diproses oleh RNN. Di setiap langkah waktu, RNN menerima input yang merepresentasikan satu potongan kecil dari suara dan memperbarui "status" internalnya (memori) berdasarkan input tersebut.

3) Prediksi Output: Setelah RNN memproses seluruh urutan suara, output dihasilkan, yang biasanya berupa prediksi kata atau perintah yang sesuai dengan suara yang telah dikenali.

RNN memiliki sejumlah keunggulan dalam pengenalan suara, terutama karena kemampuannya dalam menangani urutan data:

1) Memori Jangka Pendek: RNN dapat mengingat informasi dari langkah waktu sebelumnya dan mempertimbangkan konteks temporal dalam data suara.
2) Penerapan pada Data Berurutan: Pengenalan suara memerlukan model yang dapat memahami hubungan temporal antar suara, misalnya bagaimana suara sebelumnya memengaruhi suara berikutnya dalam kalimat atau kata.
3) Kemampuan untuk Belajar dari Data Waktu Nyata: RNN sangat efisien dalam memproses urutan waktu yang panjang, yang diperlukan dalam aplikasi seperti pengenalan ucapan real-time.

Meskipun RNN sangat kuat dalam menangani urutan data, ada beberapa tantangan dalam penggunaannya, terutama untuk pengenalan suara:

1) Vanishing and Exploding Gradient: Salah satu masalah utama RNN adalah kesulitan dalam mempelajari hubungan jangka panjang karena fenomena yang disebut *vanishing gradient* atau *exploding gradient*. Hal ini mengurangi kemampuan model untuk mengingat informasi dalam urutan yang sangat panjang.
2) Keterbatasan Kapasitas Memori: RNN tradisional hanya mampu menangani urutan dengan jangka waktu terbatas. Untuk menangani masalah ini, dikembangkanlah varian seperti Long Short-Term Memory (LSTM) dan Gated Recurrent Units (GRU), yang dirancang untuk mengatasi masalah vanishing gradient dengan cara menjaga memori lebih lama dan lebih stabil.

6.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) adalah jenis jaringan saraf tiruan yang digunakan untuk memodelkan urutan data, dan sangat efektif dalam menangani masalah yang

berhubungan dengan data berbasis waktu, seperti pengenalan suara. LSTM adalah bentuk dari Recurrent Neural Network (RNN) yang dirancang untuk mengatasi masalah vanishing gradient pada RNN standar, yang membuat LSTM lebih unggul dalam menangani data urutan panjang.

LSTM bekerja dengan mengingat informasi dalam urutan input selama jangka waktu yang panjang. Ini sangat penting dalam pengenalan suara, karena pengolahan sinyal suara melibatkan data sekuensial, seperti suara berbicara yang berkesinambungan dan dapat berlangsung dalam durasi panjang. Dalam pengenalan suara, LSTM digunakan untuk mengenali pola dalam suara dan menghubungkannya dengan teks atau perintah suara.

LSTM memiliki tiga gerbang utama yang mengatur aliran informasi:

- ✓ Forget Gate: Menentukan informasi mana yang akan dibuang atau diingat dari unit memori.
- ✓ Input Gate: Menentukan informasi mana yang akan diperbarui atau dimasukkan ke dalam memori.
- ✓ Output Gate: Menentukan informasi apa yang akan diteruskan ke unit selanjutnya dan menjadi bagian dari hasil keluaran.

Ketiga komponen ini bekerja secara bersamaan untuk menjaga informasi penting sepanjang urutan sementara mengabaikan data yang tidak relevan. Ini memungkinkan LSTM untuk mempertahankan konteks jangka panjang, yang sangat penting dalam pengenalan suara, di mana konteks temporal dan pola suara sebelumnya bisa mempengaruhi interpretasi suara berikutnya.

Pada dasarnya, pengenalan suara adalah proses mengubah sinyal suara yang diterima menjadi teks atau perintah yang dapat dimengerti oleh mesin. LSTM digunakan untuk memproses sinyal suara ini dalam beberapa tahap berikut:

- ✓ Preprocessing Sinyal Suara: Sinyal suara yang masuk diubah menjadi representasi numerik, seperti fitur Mel-frequency cepstral coefficients (MFCC), yang menggambarkan spektrum frekuensi dari sinyal suara.
- ✓ Proses Urutan dengan LSTM: Data yang telah diproses dimasukkan ke dalam jaringan LSTM. Jaringan LSTM akan menganalisis urutan data suara ini,

mengingat pola suara dalam konteks temporal, dan menghubungkannya dengan data pelatihan yang sesuai (misalnya, kata atau frasa yang sesuai).

- ✓ Prediksi Teks atau Perintah: Setelah memproses urutan data suara, LSTM memberikan output berupa prediksi teks atau perintah yang dimaksudkan, yang kemudian diterjemahkan menjadi tindakan yang sesuai dalam sistem.

LSTM dapat digunakan dalam berbagai aplikasi pengenalan suara, seperti pengenalan ucapan otomatis, asisten virtual (misalnya, Siri atau Google Assistant), dan sistem otomatisasi berbasis suara. Keunggulan LSTM dibandingkan metode lain adalah kemampuannya dalam menangani konteks suara jangka panjang, yang sangat penting dalam percakapan alami dan pemahaman suara dalam kalimat yang panjang.

LSTM memiliki beberapa keuntungan utama dalam pengenalan suara:

- ✓ Mengatasi Masalah Vanishing Gradient: Dalam RNN tradisional, gradien cenderung menghilang saat melibatkan urutan panjang, namun LSTM mengatasi masalah ini dengan menggunakan memori jangka panjang.
- ✓ Pemahaman Konteks Temporal: LSTM dapat mempertahankan dan memproses informasi yang terjadi dalam jangka panjang, yang esensial dalam pengenalan suara yang melibatkan percakapan dan perubahan suara sepanjang waktu.
- ✓ Fleksibilitas: LSTM dapat digunakan dalam berbagai macam pengenalan suara, mulai dari pengenalan ucapan hingga analisis suara dalam aplikasi seperti analisis sentimen atau emosi dalam suara.

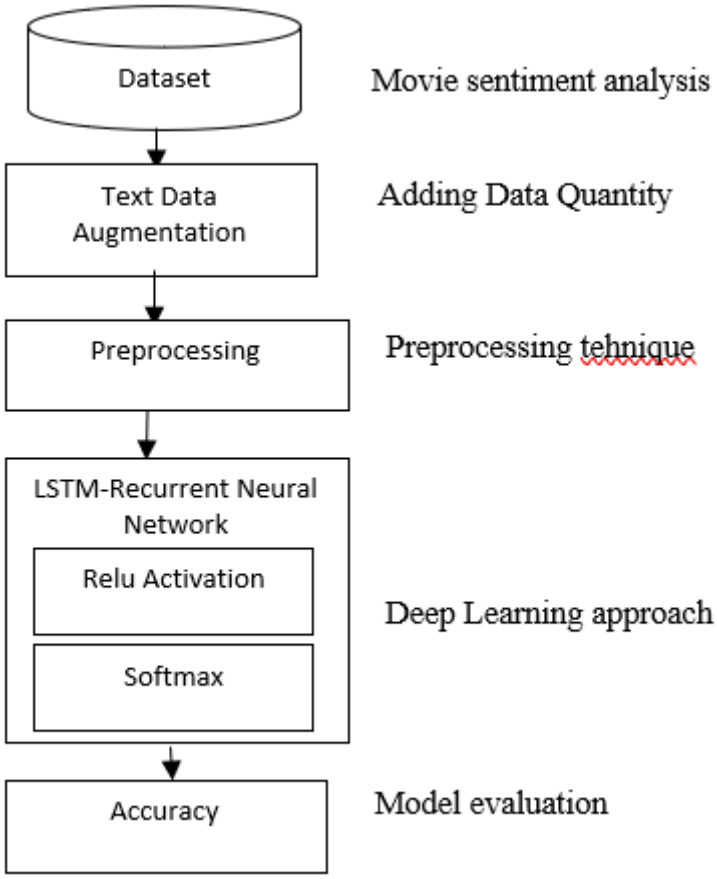
Walaupun LSTM sangat efektif dalam pengenalan suara, ada beberapa tantangan yang perlu diatasi:

- ✓ Kebutuhan Data Pelatihan yang Besar: Untuk melatih model LSTM yang efektif dalam pengenalan suara, diperlukan kumpulan data yang besar dan bervariasi agar model dapat memahami berbagai variasi suara, aksen, dan kontekstualisasi.
- ✓ Kompleksitas Komputasi: LSTM memerlukan sumber daya komputasi yang tinggi karena melibatkan banyak parameter yang perlu dioptimalkan, terutama saat memproses data suara dalam jumlah besar atau real-time.

Penelitian dalam penerapan LSTM pada pengenalan suara terus berkembang, dengan inovasi terbaru yang melibatkan kombinasi LSTM dengan model lain seperti Convolutional Neural Networks (CNN) atau Transformer untuk meningkatkan akurasi dan efisiensi. Salah satu aplikasi terbaru adalah penggunaan LSTM dalam pengenalan suara di lingkungan yang bising, yang dapat mengidentifikasi suara manusia meskipun ada gangguan eksternal.

6.3 Metode Penelitian

Penelitian yang telah dilakukan melalui serangkaian tahapan yang terlihat pada gambar 17 sebegai berikut :



Gambar 17 : Tahapan Penelitian

Jika melihat pada tahapan riset pada gambar 17 maka dapat dijelaskan sebagai berikut :

1. Pengumpulan Dataset

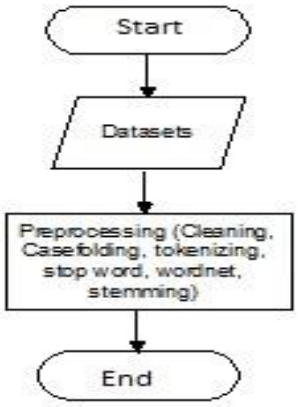
Langkah pertama yang dilakukan adalah pengunpulan dataset dimana dataset mengenai sentimen analisis didapatkan dari sitis Github dengan detail alamat sebagai berikut :

https://github.com/clairett/pytorch-sentiment-classification/tree/master/data/SST2

Kumpulan data ini berisi opini dari beberapa orang mengenai ulasan berbagai film di media sosial Twitter. Kumpulan data yang diambil awalnya berjumlah 6920 dalam format csv, dengan 3310 merupakan kelas negatif dan 3610 merupakan kelas positif.

2. Data Preprocessing

Preprocessing pada sentimen analisis merupakan tahap awal yang sangat penting untuk memastikan data teks siap diolah dan menghasilkan analisis yang akurat. Proses ini meliputi pembersihan data dari karakter yang tidak relevan seperti tanda baca, angka, atau simbol, serta normalisasi teks seperti mengubah semua kata menjadi huruf kecil. Selain itu, dilakukan juga tokenisasi (memecah kalimat menjadi kata-kata), stopword removal (menghapus kata umum yang tidak memiliki makna penting, misalnya "dan", "yang", "atau"), serta stemming atau lemmatization untuk mengembalikan kata ke bentuk dasarnya. Dengan tahapan ini, teks yang semula tidak terstruktur dapat diubah menjadi data yang lebih bersih, seragam, dan siap digunakan untuk proses ekstraksi fitur serta klasifikasi sentimen.Tahapan preprocessing bisa dilihat apda gambar 18 sebagai berikut :

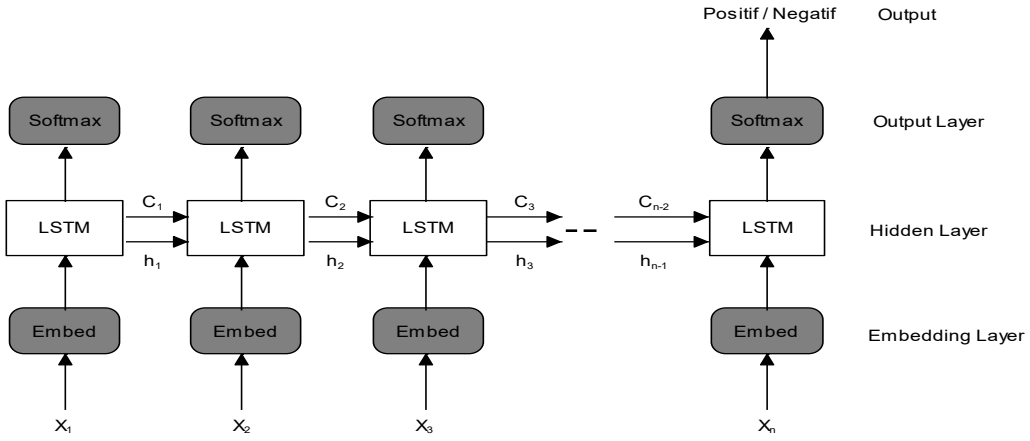


Gambar 18 : Tahapan preprocessing

3. Ekteaksi Fitur

Ekstraksi fitur pada sentimen analisis adalah proses mengubah teks yang sudah melalui tahap preprocessing menjadi representasi numerik agar dapat dipahami oleh algoritma pembelajaran mesin. Pada tahap ini, informasi penting dari teks seperti kata, frasa, atau pola tertentu dipetakan menjadi fitur yang merepresentasikan karakteristik data. Beberapa teknik umum yang digunakan antara lain Bag of Words (BoW) yang menghitung frekuensi kemunculan kata, Term Frequency-Inverse Document Frequency (TF-IDF) yang memberikan bobot pada kata berdasarkan tingkat kepentingannya, serta word embeddings seperti Word2Vec atau GloVe yang memodelkan makna kata dalam bentuk vektor berdimensi rendah. Ekstraksi fitur ini berperan penting dalam menangkap konteks dan nuansa bahasa sehingga sistem dapat lebih akurat dalam membedakan sentimen positif, negatif, maupun netral.

4. Model LSTM RNN



Gambar 19 : Arsitektur model LSTM RNN

Arsitektur Long Short-Term Memory (LSTM) yang dikombinasikan dengan Recurrent Neural Network (RNN) merupakan pendekatan yang dirancang untuk memaksimalkan kemampuan model dalam memproses data sekuensial, khususnya teks. RNN berperan dalam menangkap keterkaitan antarurutan data dengan cara menyimpan informasi dari langkah sebelumnya, namun sering terkendala masalah vanishing gradient ketika berhadapan dengan dependensi jangka panjang. Di sinilah LSTM memperkuat kinerja RNN melalui mekanisme khusus berupa *memory cell* dan *gates* (input, forget, dan

output gate) yang mampu mengatur aliran informasi secara lebih selektif, sehingga model dapat mempertahankan atau melupakan informasi penting sesuai kebutuhan. Dengan kombinasi ini, arsitektur LSTM-RNN tidak hanya mampu memahami hubungan jangka pendek antarurutan data seperti pada RNN standar, tetapi juga efektif dalam menangani konteks jangka panjang yang krusial pada analisis sentimen, penerjemahan bahasa, maupun pemrosesan teks alami lainnya.

5. Evaluasi

Evaluasi model menggunakan akurasi adalah metode yang paling umum digunakan untuk menilai performa sebuah model klasifikasi, termasuk pada sentimen analisis. Akurasi dihitung dengan membandingkan jumlah prediksi yang benar dengan total jumlah data uji, sehingga memberikan gambaran seberapa baik model mampu mengklasifikasikan data sesuai dengan label sebenarnya. Semakin tinggi nilai akurasi, semakin baik model dalam mengenali pola pada data. Namun, akurasi juga memiliki keterbatasan, terutama jika data bersifat tidak seimbang (imbalanced dataset), karena model dapat tampak memiliki kinerja tinggi hanya dengan memprediksi kelas mayoritas. Oleh karena itu, meskipun akurasi mudah dipahami dan digunakan, Rumus menghiung akurasi adalah sebagai berikut :

$$Accuracy = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

TP adalah True positive FP adalah Fales Positive

FN adalah False Negative TN adalah True Negative

6.4 Hasil Penelitian

Penelitian yang dilakukan menerapkan rasio pembagian data sebesar 70% untuk pelatihan dan 30% untuk pengujian, setiap input pada tiap langkah waktu diproses oleh LSTM untuk menghasilkan output sekaligus memperbarui status internalnya. Status ini terus diperbarui di setiap langkah waktu agar dapat menyimpan informasi dari langkah sebelumnya. Pada tahap pelatihan, LSTM RNN memanfaatkan algoritma backpropagation through time (BPTT) untuk menyesuaikan bobot dan bias sehingga diperoleh prediksi yang lebih akurat. Sementara itu, pada tahap inferensi, model LSTM RNN menggunakan bobot dan bias yang telah dilatih untuk memprediksi keluaran dari data masukan baru.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 text_vectorization (TextVec  (None, None)             0
 torization)

 embedding (Embedding)       (None, None, 64)          320000

 bidirectional (Bidirectiona  (None, None, 128)        66048
 l)

 dropout (Dropout)           (None, None, 128)         0

 bidirectional_1 (Bidirectio  (None, 64)               41216
 nal)

 dropout_1 (Dropout)         (None, 64)                0

 dense (Dense)               (None, 20)                1300

 dense_1 (Dense)             (None, 1)                 21

=================================================================
Total params: 428,585
Trainable params: 428,585
Non-trainable params: 0
_________________________________________________________________
```

Gambar 20 : Model LSTM RNN

Pada Gambar 20 ditunjukkan arsitektur LSTM RNN yang memanfaatkan lapisan tersembunyi dua arah dengan 64 unit. Model ini menggunakan dropout dengan nilai p=0,5, kemudian dilanjutkan dengan lapisan bidirectional yang memiliki 32 unit, disertai kembali dengan penerapan dropout p=0,5. Selanjutnya, ditambahkan lapisan dense dengan 20 unit tersembunyi yang menggunakan fungsi aktivasi ReLU, serta lapisan output dengan aktivasi softmax. Hasil pelatihan model selama 5 epoch ditampilkan pada Gambar 21 berikut.

```
Epoch 1/50
532/532 [==============================] - 66s 83ms/step - loss: 0.1664 - accuracy: 0.9312 - val_loss: 1.3277 - val_accuracy: 0.7287
Epoch 2/50
532/532 [==============================] - 22s 41ms/step - loss: 0.0187 - accuracy: 0.9962 - val_loss: 1.9216 - val_accuracy: 0.6969
Epoch 3/50
532/532 [==============================] - 18s 33ms/step - loss: 0.0178 - accuracy: 0.9960 - val_loss: 1.8094 - val_accuracy: 0.6974
Epoch 4/50
532/532 [==============================] - 18s 34ms/step - loss: 0.0140 - accuracy: 0.9977 - val_loss: 1.7824 - val_accuracy: 0.7024
Epoch 5/50
532/532 [==============================] - 21s 38ms/step - loss: 0.0070 - accuracy: 0.9986 - val_loss: 2.2170 - val_accuracy: 0.7051
```

Gambar 21 : Training model LSTM RNN

Hasil selengkapnya pengujian model bisa dilihat pada tampilan tabel sebagai berikut :

Tabel 8 : Perbandingan kinerja metode deteksi sentiment analisis

Dataset	**Metode**	**Hasil Akurasi**
Arabic Sentiment Tweets Dataset (ASTD)	CNN-LSTM	65.05%
Sentimen Analisis Multi-label Twitter	RNN-LSTM	67,22%
Movie Sentiment Analysis (Metode Diusulkan)	**LSTM RNN**	**70.45%**

Pada tabel 8 terlihat bahwa merode LSTM RNN lebih unggul dari sisi kinerja akurasi menggunakan dataset Movie Sentiment Analysis dengan capain akurasi sebesar 70,45 % lebih baik jika dibandingkan dengan metode CNN-LSTM maupun RNN-LSTM pada penggunaan dataset lainnya.

BAB 7

SENTIMEN ANALYSIS MENGGUNAAN BERT

7.1 Bidirectional Encoder Representations from Transformers (BERT)

Bidirectional Encoder Representations from Transformers (BERT) adalah salah satu model deep learning yang paling terkenal dalam pengolahan bahasa alami (Natural Language Processing, NLP). Meskipun BERT awalnya dikembangkan untuk tugas-tugas NLP, metode ini dapat diterapkan dalam pengenalan suara dengan melakukan beberapa penyesuaian dan transformasi. Pengenalan suara sendiri merupakan cabang dari pemrosesan sinyal suara yang bertujuan untuk mengubah sinyal audio atau suara manusia menjadi teks atau perintah yang dapat dipahami oleh sistem.

BERT adalah model berbasis arsitektur transformer yang memperkenalkan konsep *masked language modeling* (MLM). Dalam teknik ini, beberapa kata dalam sebuah kalimat di-"masking", dan tugas BERT adalah untuk memprediksi kata yang hilang tersebut berdasarkan konteks dari kata-kata sekitarnya. Kelebihan utama dari BERT adalah kemampuannya dalam memahami konteks dua arah (bidirectional), yang membedakannya dari model sebelumnya yang hanya memproses data secara unidirectional (dari kiri ke kanan atau sebaliknya). Ini memungkinkan BERT untuk menangkap makna yang lebih dalam dan kompleks dalam kalimat, baik yang bersifat sintaksis maupun semantik.

Pada pengenalan suara, tugas utama adalah mentranskripsikan ucapan menjadi teks. Secara tradisional, model-model pengenalan suara seperti *Hidden Markov Models* (HMM) dan *Deep Neural Networks* (DNN) digunakan untuk mengidentifikasi pola suara. Namun, dengan meningkatnya kemampuan pemrosesan bahasa alami melalui model-model seperti BERT, kini ada peluang untuk meningkatkan kinerja pengenalan suara dengan menggunakan BERT sebagai komponen utama.

Langkah-langkah penerapan BERT dalam pengenalan suara:

1) Preprocessing Data Suara: Data suara perlu diproses terlebih dahulu menjadi representasi yang lebih mudah dipahami oleh model. Biasanya, ini dilakukan dengan mengubah sinyal audio menjadi fitur-fitur numerik seperti *Mel-frequency cepstral coefficients* (MFCCs), yang merepresentasikan aspek-aspek penting dari suara.

2) Pengenalan Ucapan dan Representasi Teks: Setelah suara dikonversi menjadi teks awal (menggunakan model pengenalan suara tradisional), teks tersebut dapat dimasukkan ke dalam model BERT untuk diproses lebih lanjut. BERT akan memanfaatkan konteks dua arah dari teks tersebut untuk memperbaiki kesalahan pengenalan yang mungkin terjadi, seperti kata-kata yang salah transkripsi akibat noise atau aksen yang tidak dikenal.

3) Fine-Tuning untuk Pengenalan Suara: Salah satu pendekatan penting adalah *fine-tuning* BERT pada data pengenalan suara spesifik. Model BERT yang telah dilatih pada korpus bahasa alami bisa diadaptasi lebih lanjut dengan melatihnya menggunakan data pengenalan suara yang sesuai. Ini dapat mencakup data ucapan dari berbagai sumber dan aksen, yang memungkinkan model BERT lebih efektif dalam menangani variasi dalam data suara.

4) Transformasi Teks menjadi Perintah atau Tindakan: Dalam aplikasi praktis, setelah pengenalan suara berhasil dilakukan, teks yang dihasilkan bisa langsung diproses oleh model BERT untuk mengidentifikasi perintah atau tindakan spesifik. Misalnya, dalam sistem suara pintar, teks "turn on the lights" dapat diproses lebih lanjut oleh model untuk memahami bahwa itu adalah perintah untuk menyalakan lampu.

Keuntungan Menggunakan BERT dalam Pengenalan Suara

1) Pemahaman Konteks: Dengan kemampuan BERT untuk memahami konteks dua arah, model ini dapat lebih akurat dalam mengenali kata atau frasa yang memiliki makna ambigu atau memiliki variasi dalam pengucapan.

2) Peningkatan Akurasi: Pengenalan suara sering kali terhambat oleh noise atau variasi dalam cara orang berbicara (misalnya aksen atau kecepatan bicara). BERT dapat membantu mengurangi kesalahan dengan memahami konteks di sekitar kata yang terucap, sehingga meningkatkan akurasi hasil transkripsi.

3) Fleksibilitas pada Berbagai Bahasa: BERT mendukung penerapan pada berbagai bahasa, yang membuatnya berguna untuk sistem pengenalan suara multibahasa. Dengan menyesuaikan model dengan data ucapan dari berbagai bahasa, BERT bisa menangani pengenalan suara dalam konteks internasional dengan lebih baik.

Tantangan Penggunaan BERT dalam Pengenalan Suara

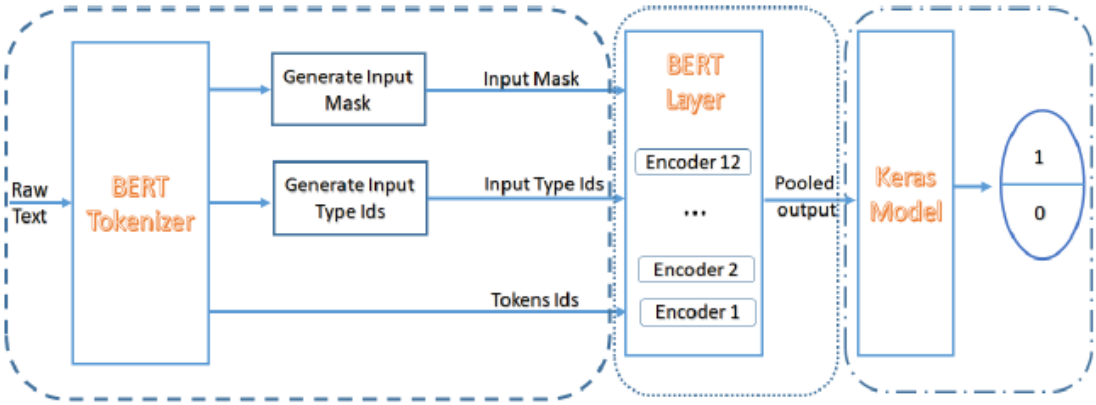
Meskipun BERT menawarkan banyak keuntungan, ada beberapa tantangan yang harus diatasi:

1) Kompleksitas dan Sumber Daya Komputasi: BERT adalah model yang sangat besar dan memerlukan sumber daya komputasi yang signifikan untuk pelatihan dan inferensi. Oleh karena itu, penerapannya dalam sistem pengenalan suara yang real-time bisa menjadi sulit tanpa optimasi yang tepat.
2) Ketersediaan Data Berkualitas Tinggi: Agar model BERT dapat di-*fine-tune* dengan baik untuk pengenalan suara, dibutuhkan data ucapan yang berkualitas tinggi dan bervariasi, yang mencakup banyak variasi dalam aksen, bahasa, dan konteks.
3) Integrasi dengan Sistem Lain: Penggunaan BERT dalam pengenalan suara seringkali memerlukan integrasi dengan sistem lain, seperti pengenalan suara berbasis deep learning, yang memerlukan pendekatan hibrida yang dapat menggabungkan kekuatan BERT dan model-model tradisional.

BERT membawa potensi besar dalam meningkatkan kinerja sistem pengenalan suara, terutama dalam hal mengurangi kesalahan pengenalan yang disebabkan oleh variabilitas bahasa dan noise. Dengan menerapkan pendekatan *fine-tuning* dan penyesuaian yang tepat, BERT dapat menjadi komponen yang sangat efektif dalam pengenalan suara, membuka peluang untuk sistem suara pintar yang lebih cerdas dan akurat. Namun, tantangan seperti kebutuhan sumber daya komputasi yang besar dan ketersediaan data berkualitas tinggi masih perlu diperhatikan untuk implementasi yang optimal.

7.2 Arsitektur BERT

Bentuk arsitektur BERT(Chouikhi et al., 2021) bisa dilihat pada gambar 19 sebagai berikut :



Gambar 22 : Arsitektur metode BERT

BERT (Bidirectional Encoder Representations from Transformers) adalah sebuah model bahasa yang dikembangkan oleh Google pada tahun 2018. BERT menggunakan teknik yang disebut *transformer* dan dirancang untuk memperbaiki kekurangan model-model sebelumnya dalam memahami konteks kata dalam sebuah kalimat. Secara umum, BERT lebih unggul dalam tugas-tugas pemahaman bahasa alami (NLP) seperti *question answering*, *sentence prediction*, dan analisis sentimen.

BERT dibangun di atas arsitektur *transformer*, yang pertama kali diperkenalkan oleh Vaswani dkk. pada 2017 dalam makalah berjudul *Attention is All You Need*. Transformer mengandalkan mekanisme yang disebut *self-attention* untuk memproses urutan data (seperti teks) secara paralel, berbeda dengan model-model sebelumnya seperti RNN (Recurrent Neural Networks) dan LSTM (Long Short-Term Memory) yang memproses data secara sekuensial.

Mekanisme *self-attention* memungkinkan setiap kata dalam kalimat untuk mempengaruhi representasi kata lainnya, tanpa memandang posisi mereka dalam kalimat. Ini memungkinkan pemodelan konteks yang lebih baik dan lebih cepat dibandingkan dengan model berbasis urutan seperti RNN.

Keunggulan utama BERT adalah kemampuannya untuk mengerti konteks dua arah. Sebelumnya, model seperti Word2Vec dan GloVe hanya menghasilkan representasi kata berdasarkan konteks satu arah (misalnya, hanya konteks kiri-kanan atau kanan-kiri). Namun, BERT mengatasi hal ini dengan memanfaatkan konteks dari kedua arah (kiri dan kanan) sekaligus.

BERT memanfaatkan *masked language model (MLM)* untuk membuat model belajar konteks dua arah. Dalam pelatihan, beberapa kata dalam kalimat akan disembunyikan (masked), dan tugas model adalah untuk memprediksi kata yang hilang tersebut, dengan memperhatikan konteks penuh dari kalimat, baik dari sebelah kiri maupun kanan.

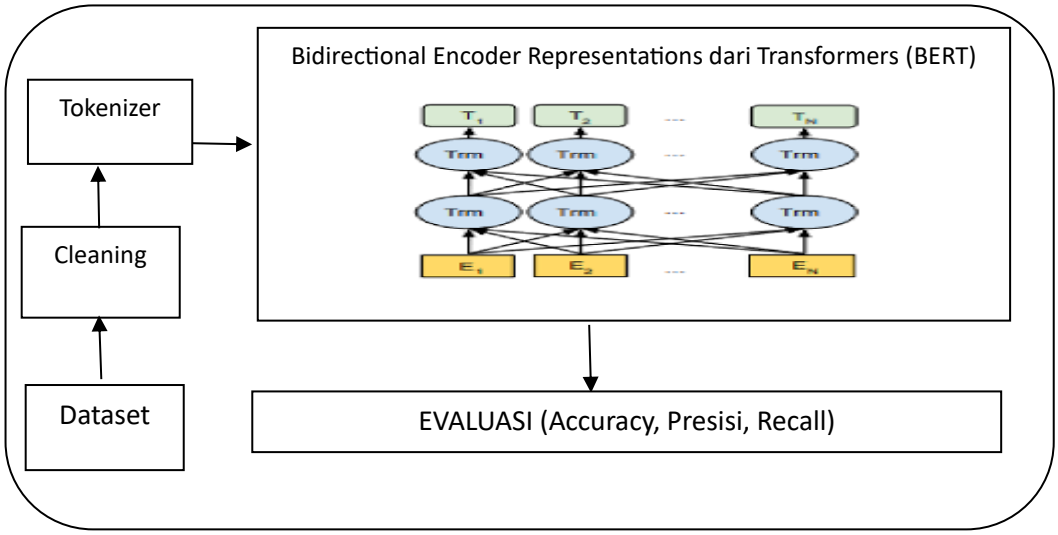
Arsitektur BERT memiliki dua komponen utama:

1) Encoder: BERT hanya menggunakan bagian encoder dari model transformer. Bagian ini bertugas untuk menghasilkan representasi kata yang kaya akan konteks dari seluruh kalimat.

2) Decoder: BERT tidak menggunakan bagian decoder seperti pada model transformer tradisional. Oleh karena itu, BERT lebih cocok untuk tugas-tugas pemahaman bahasa daripada tugas pembangkitan teks.

7.3 Metode Penelitian

Tahapan penelitian merujuk pada langkah-langkah sistematis yang dilakukan oleh peneliti untuk merancang, melaksanakan, dan menganalisis suatu penelitian. Tahapan ini membantu memastikan bahwa penelitian dilakukan dengan metode yang tepat, data yang akurat, dan hasil yang dapat dipercaya. Pada penelitian ini setiap tahapan penelitian yang dilakukan bisa dilihat pada gambar 23 sebagai berikut :



Gambar 23 : Tahapan Penelitian Diusulkan

Dataset dan Preprocessing

Pengolahan dataset merupakan langkah penting dalam proses machine learning yang melibatkan persiapan dan penyesuaian data sebelum digunakan untuk melatih atau menguji model. Tujuan dari pengolahan dataset adalah untuk memastikan bahwa data yang digunakan sesuai dan relevan dengan tujuan model machine learning yang akan dikembangkan. Pada penelitian ini dataset terdiri dari 3000 data komentar pada pembelian produk elearning yang diambil dari :

https://github.com/revanmd/indonesian-dataset-SA-ML

Repository berisi dataset mengenai review produk yang ada pada e-commerce berbahasa indonesia. Review diambil dari tiap kategori produk yang ada di e-commerce. Tiap kategori memiliki rata-rata 200 lebih produk yang berkaitan dengan kategori tersebut.

Visualisasi dataset dapat dilihat pada gambar sebagai berikut:

Gambar 24 : Visualisasi Dataset review product Ecommerce

Pada penelitian ini dipergunakan 2000 baris data review produk dari kombinasi berbagai review produk pembelian di platform Ecommerce dengan jenis barang ang berbeda beda. Pada penelitian ini dipergunakan teknik Tokenizer yang merupakan proses pemrosesan awal dalam analisis teks yang bertujuan untuk mengubah data mentah (raw text) menjadi format yang lebih terstruktur agar dapat digunakan dalam model machine learning atau deep learning. Disini dipergunakan teknik tokenizing **Word Tokenizer** yaitu Memisahkan teks berdasarkan kata (contoh: *"Saya suka AI"* → ["Saya", "suka", "AI"]). Berikut merupakan script pada Jupiter Notebook Python :

Arsitektur BERT

BERT sepenuhnya berbasis pada Encoder dari Transformer (tidak menggunakan Decoder seperti pada Transformer asli (Vaswani et al., 2017).

- ✓ Input Representation: Setiap token (kata/sub-kata) diubah menjadi representasi embedding yang terdiri dari tiga komponen:
  1. Token Embeddings – representasi dari token yang dihasilkan oleh *WordPiece tokenizer*.
  2. Segment Embeddings – menandai kalimat A atau B (berguna untuk *next sentence prediction*).
  3. Position Embeddings – menambahkan informasi urutan posisi karena self-attention tidak memiliki sifat urutan secara alami.
- ✓ Stacked Encoder Layers: BERT memiliki beberapa lapisan Encoder Transformer (BERT-Base: 12 layer, BERT-Large: 24 layer). Setiap layer terdiri dari:
  1. Multi-Head Self-Attention (MHSA)
     Menghitung perhatian antar semua token sehingga tiap kata bisa "melihat" kata lain di dalam urutan input.
  2. Feed-Forward Neural Network (FFN)
     Dua lapisan fully-connected dengan aktivasi non-linear (biasanya GELU).
  3. Residual Connections + Layer Normalization
     Membantu stabilitas pelatihan dan mencegah hilangnya informasi.

Mekanisme Kunci :

1. Self-Attention

Rumus utama:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Di mana Query (Q), Key (K), dan Value (V) diperoleh dari embedding input. Dengan multi-head attention, BERT bisa menangkap berbagai jenis hubungan semantik sekaligus.

2. Bidirectional Context

Tidak seperti model sebelumnya (misalnya GPT yang hanya *left-to-right*), BERT menggunakan Masked Language Model (MLM), sehingga bisa belajar konteks dari kiri dan kanan.

Evaluasi

1. Confusion Matrix sebagai Dasar

Saat memahami metrik evaluasi, kita perlu melihat Confusion Matrix (khususnya pada klasifikasi biner):

	Prediksi Positif	Prediksi Negatif
Aktual Positif	True Positive (TP)	False Negative (FN)
Aktual Negatif	False Positive (FP)	True Negative (TN)

TP (True Positive) → Model benar memprediksi positif.

TN (True Negative) → Model benar memprediksi negatif.

FP (False Positive) → Model salah memprediksi positif (padahal negatif).

FN (False Negative) → Model salah memprediksi negatif (padahal positif).

2. Akurasi (Accuracy)

Mengukur seberapa sering model benar secara keseluruhan.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Kelebihan: mudah dipahami, cocok bila data seimbang.

Kekurangan: menyesatkan pada data imbalanced (misalnya 95% data negatif → model selalu tebak negatif → akurasi tinggi tapi tidak berguna).

3. Presisi (Precision)

Mengukur seberapa tepat prediksi positif model.

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

Tinggi jika FP kecil → model jarang memberikan prediksi positif yang salah.

Contoh kasus: deteksi spam → lebih baik presisi tinggi agar email normal tidak salah dikira spam.

4. Recall (Sensitivity / True Positive Rate)

Mengukur seberapa banyak kasus positif yang berhasil ditemukan oleh model.

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

Tinggi jika FN kecil → model jarang melewatkan kasus positif.

Contoh kasus: deteksi kanker → lebih baik recall tinggi agar tidak ada pasien positif yang terlewat.

5. F1-Score

Merupakan harmonic mean antara Presisi dan Recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Berguna saat data imbalanced.

Nilai tinggi hanya tercapai jika presisi dan recall seimbang.

Cocok untuk kasus di mana FN dan FP sama-sama berbahaya.

7.4 Hasil Penelitian

Penelitian yang dilakukan merupakan jenis experimental dengan langkah awalnya adalah dengan mendapatkan dataset dari Github dengan link pada alamat web sebagai berikut :

https://github.com/revanmd/indonesian-dataset-SA-ML

Proses selanjutnya adalah membuka dataset dengan script python sebagai berikut :

```
# Load dataset
file_path = "/content/drive/My Drive/Colab Notebooks/Udin/BERTLSTM/ECommerce-Sentiment.csv"  # Sesuaikan dengan path lokal Anda
df = pd.read_csv(file_path, delimiter=';')
```

Selanjutnya dilakukan proses **preprocessing** untuk mempersiapkan data mentah agar lebih siap digunakan dalam analisis atau model pembelajaran mesin. Scriptnya bisa dilihat pada tampilan sebagai berikut :

```
df['sentiment'] = df['rating'].apply(categorize_sentiment)

df.dropna(subset=['review'], inplace=True)  # Hapus review kosong
df['review'] = df['review'].astype(str)  # Pastikan semua review berbentuk string

# Load tokenizer
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")

def tokenize_texts(texts, tokenizer, max_length=128):
    return tokenizer(
        list(texts),
        padding="max_length",
        truncation=True,
        max_length=max_length,
        return_tensors="pt"
    )
```

Sebelum proses training model dilakukan, perlu dilakukan split (pembagian) dataset dimana 80% data akan menjadi data training dan sisanya 20% menjadi data testing. Script program selengkapnya bisa dilihat pada tampilan dibawah ini :

```
# Split data
from sklearn.model_selection import train_test_split
train_texts, test_texts, train_labels, test_labels = train_test_split(
    df['review'].tolist(), df['sentiment'].tolist(), test_size=0.2, random_state=42, stratify=df['sentiment']
)

train_encodings = tokenize_texts(train_texts, tokenizer)
test_encodings = tokenize_texts(test_texts, tokenizer)

train_labels_tensor = torch.tensor(train_labels, dtype=torch.long)
test_labels_tensor = torch.tensor(test_labels, dtype=torch.long)

train_dataset = TensorDataset(train_encodings["input_ids"], train_encodings["attention_mask"], train_labels_tensor)
test_dataset = TensorDataset(test_encodings["input_ids"], test_encodings["attention_mask"], test_labels_tensor)

train_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=16, shuffle=False)
```

Langkah selanjutnya adalah mempersiapkan model dimana program aplikasinya menggunakan python bisa dilihat selengkapnya sebagai berikut ini :

```python
# Define Model
class BertLSTMClassifier(nn.Module):
    def __init__(self, bert_model_name="bert-base-uncased", hidden_dim=128, num_classes=3):
        super(BertLSTMClassifier, self).__init__()
        self.bert = BertModel.from_pretrained(bert_model_name)
        self.lstm = nn.LSTM(768, hidden_dim, batch_first=True, bidirectional=True)
        self.fc = nn.Linear(hidden_dim * 2, num_classes)
        self.softmax = nn.Softmax(dim=1)

    def forward(self, input_ids, attention_mask):
        with torch.no_grad():
            bert_output = self.bert(input_ids, attention_mask=attention_mask)
        lstm_output, _ = self.lstm(bert_output.last_hidden_state)
        output = self.fc(lstm_output[:, -1, :])
        return self.softmax(output)
```

Model yang dihasilkan perlu ditraining menggunakan script sebagai berikut :

```python
# Train Model
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = BertLSTMClassifier().to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=2e-5)

num_epochs = 5
train_losses = []

for epoch in range(num_epochs):
    model.train()
    total_loss = 0
    for input_ids, attention_mask, labels in train_loader:
        input_ids, attention_mask, labels = input_ids.to(device), attention_mask.to(device), labels.to(device)
        optimizer.zero_grad()
        outputs = model(input_ids, attention_mask)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        total_loss += loss.item()
    train_losses.append(total_loss / len(train_loader))
    print(f"Epoch {epoch+1}, Loss: {train_losses[-1]:.4f}")
```

Langkah selanjutnya adalah evaluasi dari kinerja model yang telah dihasilkan dimana scriptnya bisa dijelaskan sebagai berikut :

```python
# Evaluate Model
model.eval()
predictions, true_labels = [], []
with torch.no_grad():
    for input_ids, attention_mask, labels in test_loader:
        input_ids, attention_mask = input_ids.to(device), attention_mask.to(device)
        outputs = model(input_ids, attention_mask)
        preds = torch.argmax(outputs, dim=1).cpu().numpy()
        predictions.extend(preds)
        true_labels.extend(labels.numpy())

accuracy = accuracy_score(true_labels, predictions)
precision, recall, f1, _ = precision_recall_fscore_support(true_labels, predictions, average='weighted')
print(f"Accuracy: {accuracy:.4f}, Precision: {precision:.4f}, Recall: {recall:.4f}, F1-score: {f1:.4f}")
```

Hasil Eksperimen

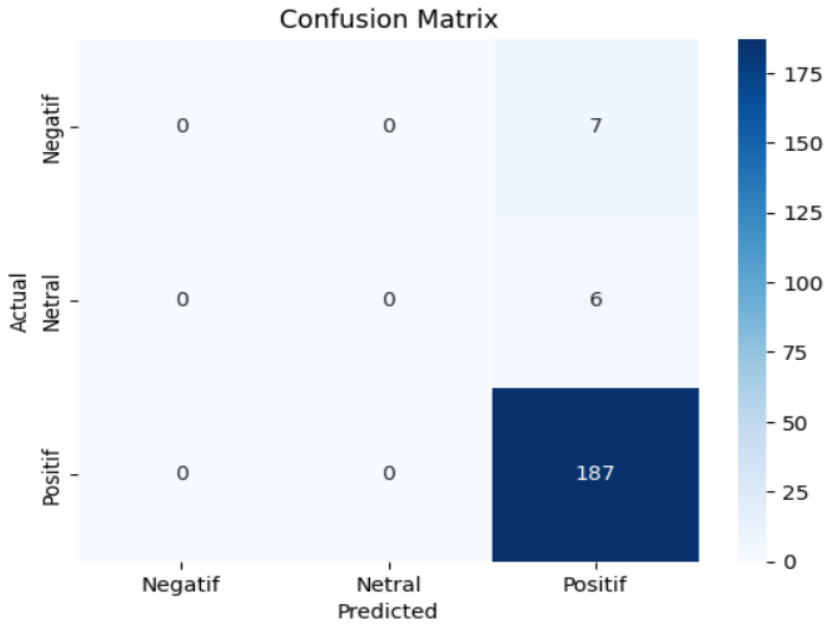
A. Simulasi 1000 baris data

```
Epoch 1, Loss: 1.0437
Epoch 2, Loss: 0.9388
Epoch 3, Loss: 0.8337
Epoch 4, Loss: 0.7469
Epoch 5, Loss: 0.6931
```

Pada eksperimen dengan 5 epoch terlihat dari epoch 1 sampai dengan 5 tingkat Loss semakin mengecil dari 1.0437 sampai dengan 0.931 sedangkan pada pengukuran Akurasi,Presisi dan Recall didapakan hasil sebagai berikut :

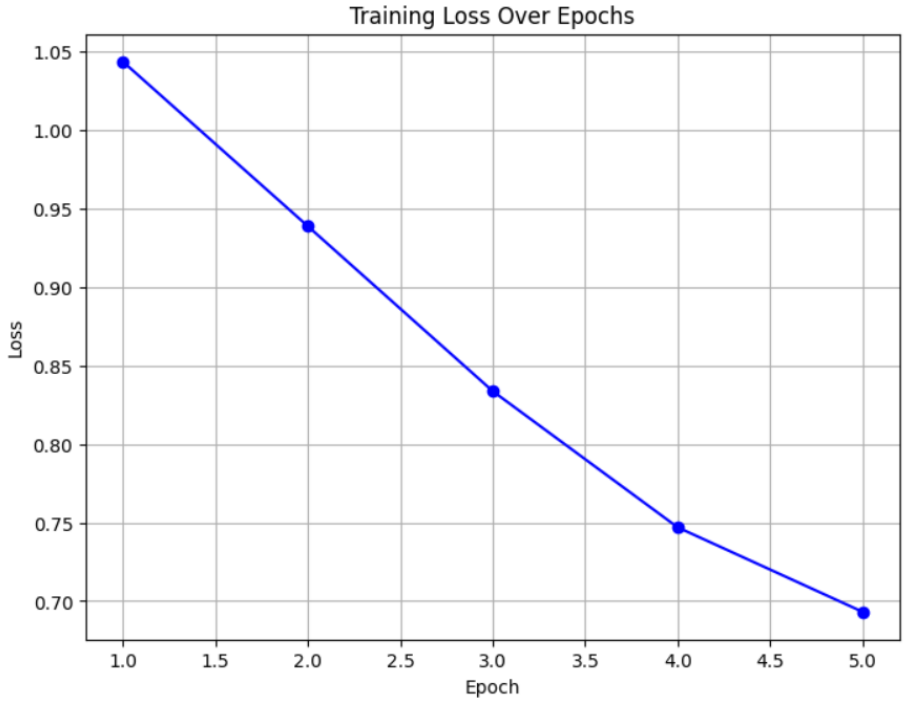
Matriks Evaluasi	Hasil
Accuracy	93,50 %
Precission	87,42 %
Recall	93,50 %
F1 Score	90,36 %

Jika kinerjanya di visualisasikan menggunakan confusion matriks maka hasilnya nampak pada gambar 25 sebagai berikut ini :



Gambar 25 : Confusion matriks 1000 data

Selanjutnya hasil kinerja dari model yang telah dihasilkan ditampilkan pada gambar grafik seperti pada gambar 26 sebagai berikut ini :



Gambar 26 : Grafik Kinerja Model 1000 data

B. Simulasi 1500 baris data

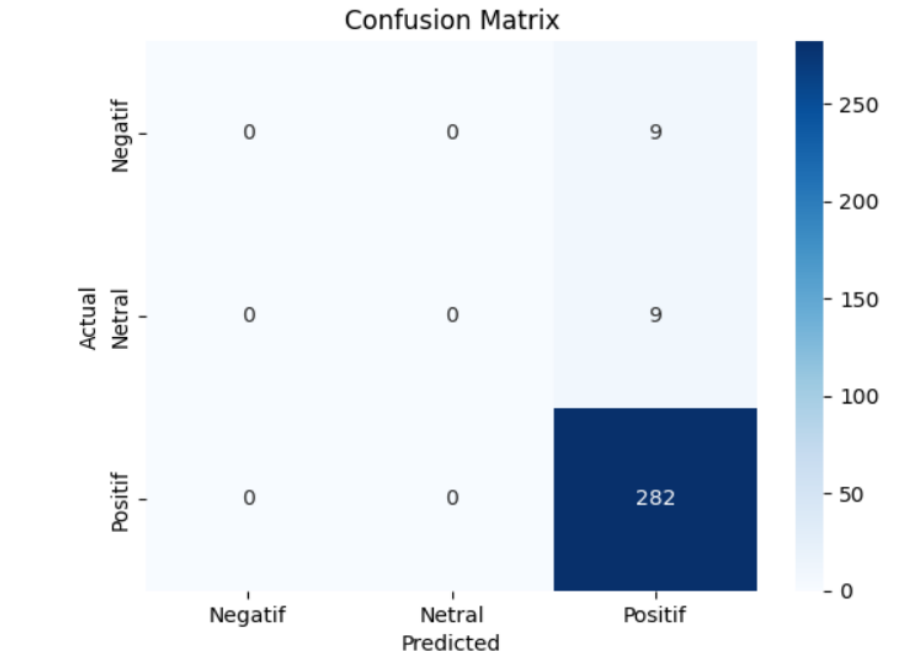
Pada eksperimen dengan 5 epoch terlihat dari epoch 1 sampai dengan 5 tingkat Loss semakin mengecil dri 1.0341 sampai dengan 0.6359 sedangkan pada pengukuran Akurasi,Presisi dan Recall didapakan hasil sebagai berikut :

```
Epoch 1, Loss: 1.0341
Epoch 2, Loss: 0.8653
Epoch 3, Loss: 0.7257
Epoch 4, Loss: 0.6611
Epoch 5, Loss: 0.6359
```

Hasil pengukuran Kinerja Model :

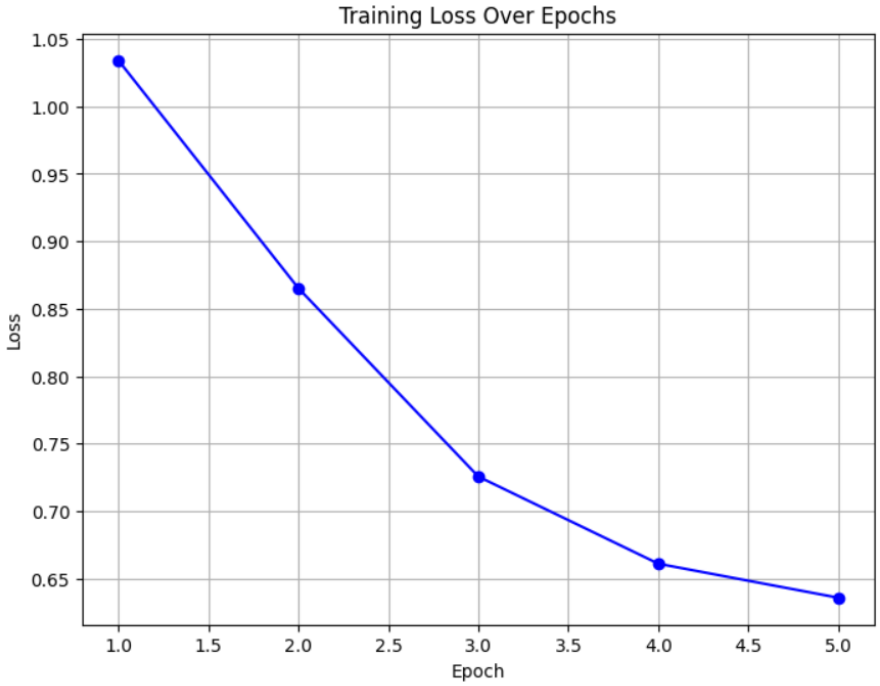
Matriks Evaluasi	Hasil
Accuracy	94 %
Precission	88,36 %
Recall	94 %
F1 Score	91,09 %

Jika kinerja ditampilkan pada confusion matriks didapatkan hasil sebagai berikut ini :



Gambar 27 : Confusion matriks 1500 data

Selanjunya kinerja model divsualisasikan menggunakan gambar grafik seperti dibawah ini:



Gambar 28 : Gambar kinerja model 1500 data

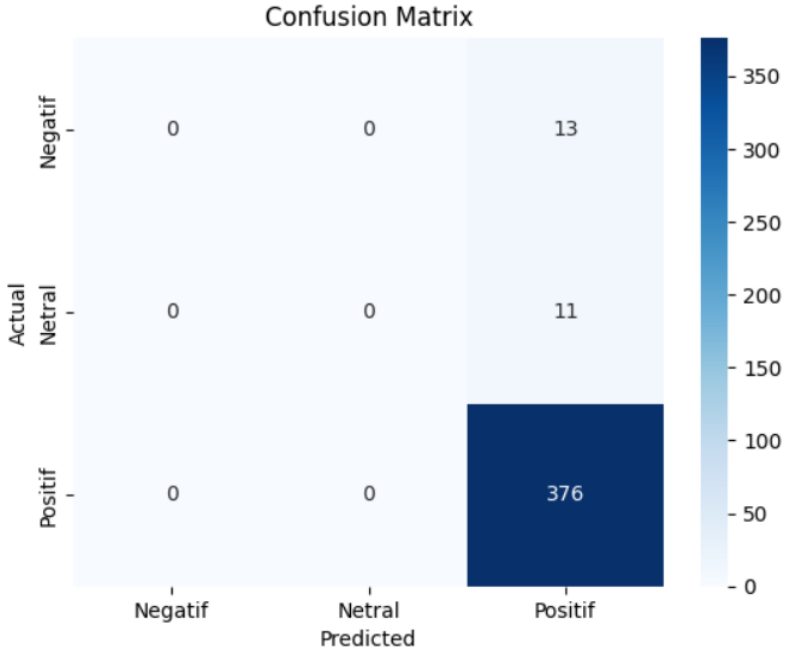
C. Simulasi 2000 baris data

```
Epoch 1, Loss: 0.9514
Epoch 2, Loss: 0.7451
Epoch 3, Loss: 0.6565
Epoch 4, Loss: 0.6327
Epoch 5, Loss: 0.6228
```

Pada eksperimen dengan 5 epoch terlihat dari epoch 1 sampai dengan 5 tingkat Loss semakin mengecil dari 0,9514 sampai dengan 0.6228 sedangkan pada pengukuran Akurasi,Presisi dan Recall didapakan hasil sebagai berikut :

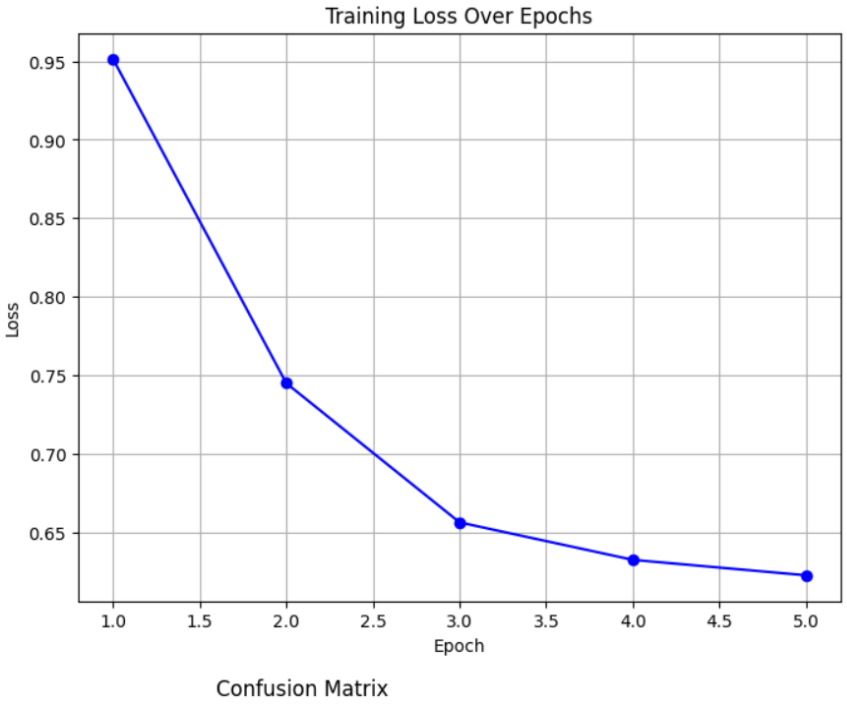
Matriks Evaluasi	Hasil
Accuracy	94 %
Precission	88,36 %
Recall	91 %
F1 Score	91,09 %

Hasil pengukuran tersebut divisualisasikan menggunakan confusion mattriks dapat dilihat pada tampilan berikut ini :



Gambar 29 : Confusion matriks 2000 data

Tahapan selanjutnya menampilkan hasil kinerja model pada grafik dengan tampilan seperti pada gambar 30 sebagai berikut :



Gambar 30 : Gambar kinerja model 2000 data

Penelitian mengenai analisis sentimen ulasan produk pada online store berbahasa Indonesia dengan metode BERT menunjukkan hasil yang sangat baik dalam mengklasifikasikan sentimen positif, negatif, dan netral dari ulasan pelanggan. Model yang dikembangkan dalam penelitian ini mampu mencapai akurasi sebesar 94%, menandakan bahwa pendekatan deep learning dengan arsitektur BERT sangat efektif dalam memahami konteks dan makna kata dalam bahasa Indonesia. Keunggulan utama model ini terletak pada kemampuannya dalam menangkap hubungan kata dalam satu kalimat secara lebih akurat dibandingkan dengan metode konvensional seperti Naïve Bayes atau LSTM. Hasil eksperimen menunjukkan bahwa model BERT mampu mengatasi tantangan dalam analisis sentimen, seperti keberagaman struktur kalimat, slang, serta penggunaan kata-kata ambigu yang sering muncul dalam ulasan online. Selain itu, implementasi fine-tuning pada BERT menggunakan dataset khusus ulasan e-commerce berbahasa Indonesia turut meningkatkan performa model dalam mengenali pola sentimen secara lebih presisi. Dengan pencapaian akurasi yang tinggi, model ini memiliki potensi besar untuk diterapkan dalam sistem rekomendasi, pemantauan reputasi merek, serta peningkatan pengalaman pelanggan dalam e-commerce berbasis bahasa Indonesia, sehingga memberikan manfaat signifikan bagi pelaku bisnis online dalam memahami opini

BAB 8

PENUTUP

Buku *"Riset Deep Learning Berbasis Suara & Teks"* menyoroti pentingnya integrasi pemrosesan suara dan teks dalam membangun sistem kecerdasan buatan modern. Dengan dukungan arsitektur deep learning seperti RNN, LSTM, CNN, Transformer, dan BERT, teknologi ini mampu memahami intonasi, pola suara, serta konteks linguistik untuk menghasilkan interaksi manusia-mesin yang lebih alami. Fokus utama buku ini adalah memberikan pemahaman teoritis sekaligus praktik dalam mengembangkan sistem yang cerdas, adaptif, dan relevan dengan kebutuhan nyata.

Metodologi penelitian yang dipaparkan meliputi penggunaan **Singular Value Decomposition (SVD)** untuk deteksi suara, **Data Augmentation** (pitch shifting, time stretching, white noise) untuk memperkaya dataset, hingga penerapan **Deep Neural Network (DNN)**, **Generative Adversarial Network (GAN)**, dan **BERT** untuk tugas klasifikasi suara, pengenalan pembicara multi-etnis, serta analisis sentimen. Hasil eksperimen menunjukkan bahwa model deep learning secara konsisten menghasilkan akurasi lebih tinggi dibandingkan metode tradisional seperti Naive Bayes, Logistic Regression, maupun SVM.

Aplikasi yang dibahas mencakup bidang keamanan, kesehatan, industri, hingga inklusi sosial. Misalnya, deteksi ujaran palsu dan identifikasi pembicara untuk keamanan, pemantauan pasien melalui suara untuk kesehatan, otomatisasi layanan pelanggan dan asisten virtual untuk dunia industri, hingga *text-to-speech* bagi penyandang disabilitas. Hal ini menegaskan bahwa riset suara dan teks tidak hanya bernilai akademis, tetapi juga memberikan manfaat langsung bagi kehidupan masyarakat modern.

Namun, buku ini juga menekankan sejumlah tantangan penting, seperti kebutuhan dataset besar dan berkualitas, biaya komputasi yang tinggi, keterbatasan interpretabilitas model, serta potensi bias algoritmik. Oleh karena itu, pengembangan deep learning berbasis suara dan teks harus disertai kesadaran etika serta pendekatan lintas disiplin. Secara keseluruhan, buku ini memberikan kontribusi berharga sebagai referensi ilmiah dan praktis, yang mampu menjembatani dunia akademik dan industri dalam menciptakan teknologi AI yang lebih humanistik, inklusif, dan bermanfaat luas.

REFERENSI

An, N. N., Thanh, N. Q., & Liu, Y. (2019). Deep CNNs With Self-Attention for Speaker Identification. *IEEE Access*, *7*(c), 85327–85337. https://doi.org/10.1109/ACCESS.2019.2917470

Chakroun, R., & Frikha, M. (2020). Robust Text-independent Speaker recognition with Short Utterances using Gaussian Mixture Models. *2020 International Wireless Communications and Mobile Computing, IWCMC 2020*, 2204–2209. https://doi.org/10.1109/IWCMC48107.2020.9148102

Chouikhi, H., Chniter, H., & Jarray, F. (2021). Arabic Sentiment Analysis Using BERT Model. *Communications in Computer and Information Science*, *1463*(November), 621–632. https://doi.org/10.1007/978-3-030-88113-9_50

Hanifa, R. M., Isa, K., & Mohamad, S. (2020). Speaker ethnic identification for continuous speech in Malay language using pitch and MFCC. *Indonesian Journal of Electrical Engineering and Computer Science*, *19*(1), 207–214. https://doi.org/10.11591/ijeecs.v19.i1.pp207-214

Little, C., Elliot, M., Allmendinger, R., & Samani, S. S. (2021). *Generative Adversarial Networks for Synthetic Data Generation: A Comparative Study*. *2018*, 1–16. http://arxiv.org/abs/2112.01925

Long, Y., Li, Y., Zhang, Q., Wei, S., Ye, H., & Yang, J. (2020). Acoustic data augmentation for Mandarin-English code-switching speech recognition. *Applied Acoustics*, *161*, 107175. https://doi.org/10.1016/j.apacoust.2019.107175

Maurya, A., Kumar, D., & Agarwal, R. K. (2018). Speaker Recognition for Hindi Speech Signal using MFCC-GMM Approach. *Procedia Computer Science*, *125*, 880–887. https://doi.org/10.1016/j.procs.2017.12.112

Mclaren, M., Lei, Y., Ferrer, L., & Aires, U. D. B. (2015). *ADVANCES IN DEEP NEURAL NETWORK APPROACHES TO SPEAKER RECOGNITION*. 4814–4818.

Mouaz, B., Abderrahim, B. H., & Abdelmajid, E. (2019). Speech recognition of Moroccan dialect using hidden Markov models. *IAES International Journal of Artificial Intelligence (IJ-AI)*, *151*(2018), 985–991. https://doi.org/10.1016/j.procs.2019.04.138

Novotný, O., Plchot, O., Glembek, O., Černocký, J. “Honza,” & Burget, L. (2019). Analysis

of DNN Speech Signal Enhancement for Robust Speaker Recognition. *Computer Speech and Language*, *58*, 403–421. https://doi.org/10.1016/j.csl.2019.06.004

Rajyaguru, V., Vithalani, C., & Thanki, R. (2020). ORIGINAL RESEARCH A literature review : various learning techniques and its applications for eye disease identification using retinal images. *International Journal of Information Technology*. https://doi.org/10.1007/s41870-020-00442-8

Salamon, J., & Bello, J. P. (2017). Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Processing Letters*, *24*(3), 279–283. https://doi.org/10.1109/LSP.2017.2657381

Saleem, N., & Irfan Khattak, M. (2020). Deep neural networks based binary classification for single channel speaker independent multi-talker speech separation. *Applied Acoustics*, *167*, 107385. https://doi.org/10.1016/j.apacoust.2020.107385

Seifert, C., Aamir, A., Balagopalan, A., Jain, D., Sharma, A., Grottel, S., & Gumhold, S. (2017). *Visualizations of Deep Neural Networks in Computer Vision : A Survey*.

Seki, H., Yamamoto, K., & Nakagawa, S. (2016). Deep neural network based acoustic model using speaker-class information for short time utterance. *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2015*, *December*, 1222–1225. https://doi.org/10.1109/APSIPA.2015.7415467

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., & Khudanpur, S. (2018). X-Vectors: Robust DNN Embeddings for Speaker Recognition. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2018-April*, 5329–5333. https://doi.org/10.1109/ICASSP.2018.8461375