PEMANFAATAN METODE *MEL FREQUENCY CEPSTRAL COEFFCIENT* DAN *LEARNING VECTOR QUANTIZATION* DALAM PENGENALAN UCAPAN BAHASA INDONESIA

SKRIPSI

oleh :

Muhammad Reza Ivano Pahlevi

NIM E41211245

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER

2025

PEMANFAATAN METODE *MEL FREQUENCY CEPSTRAL COEFFCIENT* DAN *LEARNING VECTOR QUANTIZATION* DALAM PENGENALAN UCAPAN BAHASA INDONESIA

SKRIPSI

Sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains Terapan Komputer (S.Tr.Kom.) di Program Studi Teknik Informatika Jurusan Teknologi Informasi

oleh :

Muhammad Reza Ivano Pahlevi

NIM E41211245

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2025

KEMENTRIAN PENDIDIKAN TINGGI, SAINS, DAN TEKNOLOGI
POLITEKNIK NEGERI JEMBER
JURUSAN TEKNOLOGI INFORMASI

PEMANFAATAN METODE *MEL FREQUENCY CESPTRAL COEFFICIENT* DAN *LEARNING VECTOR QUANTIZATION* DALAM PENGENALAN UCAPAN BAHASA INDONESIA

Muhammad Reza Ivano Pahlevi (E41211245)

Telah Diuji pada Tanggal 25 Maret 2025
dan Dinyatakan Memenuhi Syarat

Ketua Penguji,

Elly Antika S.T., M.Kom.
NIP. 197810112005012002

Sekretaris Penguji	Anggota Penguji
Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs	Zilvanhisna Emka Fitri, ST. MT
NIP. 198907102019031010	NIP. 19920302 2018032001

Dosen Pembimbing

Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs
NIP. 198907102019031010

Mengesahkan
Ketua Jurusan Teknologi Informasi

Hendra Yufit Riskiawan, S.Kom, M.Cs
NIP. 198302032006041003

SURAT PERNYATAAN MAHASISWA

Saya yang bertanda tangan di bawah ini:

Nama : Muhammad Reza Ivano Pahlevi
NIM : E41211245

menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam Skripsi saya yang berjudul "Pemanfaatan Metode *Mel Frequency Cepstral Coefficent* dan *Learning Vector Quantization* Dalam Pengenalan Ucapan Bahasa Indonesia" merupakan gagasan dan hasil karya sendiri dengan arahan komisi pembimbing, dan belum pernah diajukan dalam bentuk apa pun pada perguruan tinggi mana pun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan dari penulis lain telah disebutkan dalam naskah dan dicantumkan dalam daftar pustaka di bagian akhir Skripsi ini.

Jember, 25 Maret 2025

Muhammad Reza Ivano Pahelvi

NIM E41211245

PERNYATAAN
PERSETUJUAN PUBLIKASI
KARYA ILMIAH UNTUK KEPENTINGAN
AKADEMIS

Yang bertanda tangan di bawah ini, Saya:

Nama : Muhammad Reza Ivano Pahlevi
NIM : E41211245
Program Studi : Teknik Informatika
Jurusan : Teknologi Informasi

Demi pengembangan Ilmu Pengetahuan, saya menyetujui untuk memberikan kepada UPT. Perpustakaan Politeknik Negeri Jember, Hak Bebas Royalti Non Eksklusif (*Non-Exclusive Royalty Free Right*) atas Karya Ilmiah berupa **Laporan Skripsi saya yang berjudul:**

PEMANFAATAN METODE MEL FREQUENCY CEPSTRAL COEFFICENT DAN LEARNING VECTOR QUANTIZATION DALAM PENGENALAN UCAPAN BAHASA INDONESIA

Dengan Hak Bebas Royalti Non-Eksklusifini UPT. Perpustakaan Politeknik Negeri Jember berhak menyimpan, mengalih media atau format, mengelola dalam bentuk Pangkalan Data (*Database*), mendistribusikan karya dan menampilkan atau mempublikasikannya di Internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis atau pencipta.
Saya bersedia untuk menanggung secara pribadi tanpa melibatkan pihak Politeknik Negeri Jember, Segala bentuk tuntutan hukum yang timbul atas Pelanggaran Hak Cipta dalam Karya ilmiah ini.
Demikian pernyataan ini saya buat dengan sebenernya.

Dibuat di : Jember
Pada tanggal : 25 Maret 2025
Yang menyatakan,

Nama : Muhammad Reza Ivano P.
NIM : E41211245

HALAMAN MOTTO

"*Kamu tidak harus menjadi hebat untuk memulai, tetapi kamu harus mulai untuk menjadi hebat."*

(Zig Ziglar)

PERSEMBAHAN

Puji syukur atas karunia Allah SMT sehingga Saya dapat menyelesaikan Skripsi dengan lancar. Karya Tulis Ilmiah ini Saya persembahkan kepada:

1. Diri saya Sendiri, yang telah berusaha sebaik mungkin.
2. Seluruh keluarga dan saudara, terima kasih doa dan dukungannya
3. Dosen pembimbing saya, Bapak Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs yang telah memberikan waktu, arahan dan bimbingan terbaik dan penuh kemudahan selama proses pengerjaan skripsi.
4. Teman seperjuangan yang telah membantu dan memberikan *support* selama ini.
5. Para staf pengajar Politeknk Negeri Jember khususnya Program Studi Teknik Informatika yang telah memberikan banyak ilmu dan pengetahuan.
6. Almamater tercinta Politeknik Negeri Jember.

Utilization of Mel Frequency Cepstral Coefficient Method and Learning Vector Quantization in Indonesian Speech Recognition

Supervised by Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs

Muhammad Reza Ivano Pahlevi
Study program of Informatic Engineering
Majoring of Information Technology

ABSTRACT

Speech recognition is one of the rapidly evolving technologies in the field of artificial intelligence, particularly in human-computer interaction systems. This study aims to develop an Indonesian speech recognition model using the Mel-Frequency Cepstral Coefficient (MFCC) method for feature extraction and Learning Vector Quantization (LVQ) as the classification method. The research begins with the collection of voice recordings from five participants, consisting of three males and two females, who pronounce five adjectives in Indonesian. The recorded speech data undergoes preprocessing steps, including amplitude normalization, padding, and noise reduction, before feature extraction using MFCC. The extracted speech features are then used to train the LVQ classification model. The results indicate that the combination of MFCC and LVQ can recognize words with a highest accuracy of 91.45%, using a learning rate of 0.01, 200 epochs, and a 70:30 dataset split. Evaluation using K-Fold Cross Validation (K=5) shows an average accuracy of 82.53%, indicating that the model has good generalization capabilities. Tests on new speech data also demonstrate that the model can correctly recognize most words, although some challenges arise with words that have similar acoustic characteristic.

Keywords*: Speech recognition, Mel-Frequency Cepstral Coefficient (MFCC), Learning Vector Quantization (LVQ), speech classification, speech signal processing.*

Pemanfaatan Metode *Mel Frequency Cepstral Coefficient* Dan *Learning Vector Quantization* Dalam Pengenalan Ucapan Bahasa Indonesia

Dibimbing oleh Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs

Muhammad Reza Ivano Pahlevi
Program Studi Teknik Informatika
Jurusan Teknologi Informasi

ABSTRAK

Pengenalan suara merupakan salah satu teknologi yang semakin berkembang dalam bidang kecerdasan buatan, terutama dalam sistem interaksi manusia dan komputer. Penelitian ini bertujuan untuk mengembangkan model pengenalan ucapan bahasa Indonesia menggunakan metode *Mel-Frequency Cepstral Coefficient* (MFCC) untuk ekstraksi fitur suara dan *Learning Vector Quantization* (LVQ) sebagai metode klasifikasi. Metode penelitian ini diawali dengan pengumpulan data rekaman suara dari lima partisipan, terdiri dari tiga laki-laki dan dua perempuan, yang mengucapkan lima kata sifat dalam bahasa Indonesia. Data suara yang diperoleh melalui proses *preprocessing* data, meliputi normalisasi amplitudo, *padding, dan noise reduction*, sebelum dilakukan ekstraksi fitur menggunakan MFCC. Selanjutnya, fitur suara yang telah diekstraksi digunakan untuk melatih model klasifikasi LVQ. Hasil penelitian menunjukkan bahwa kombinasi metode MFCC dan LVQ mampu mengenali kata dengan akurasi tertinggi sebesar 91.45% menggunakan kombinasi learning rate 0.01 dan epoch 200 dengan pembagian dataset 70:30. Evaluasi menggunakan *K-Fold Cross Validation* (K=5) menunjukkan akurasi rata-rata sebesar 82.53%, yang mengindikasikan bahwa model memiliki tingkat generalisasi yang baik. Pengujian terhadap data suara baru juga menunjukkan bahwa model dapat mengenali sebagian besar kata dengan benar, meskipun terdapat beberapa kendala pada kata dengan karakteristik suara yang mirip.

Kata Kunci: Pengenalan suara, *Mel-Frequency Cepstral Coefficient* (MFCC), *Learning Vector Quantization* (LVQ), klasifikasi suara, pengolahan sinyal suara.

RINGKASAN

Pemanfaatan Metode *Mel Frequency Cepstral Coefficient* Dan *Learning Vector Quantization* Dalam Pengenalan Ucapan Bahasa Indonesia, Muhammad Reza Ivano Pahlevi, NIM E41211245, Tahun 2025., hlm. 72. Teknologi Informasi, Politeknik Negeri Jember, Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs (Pembimbing).

Pengenalan suara merupakan teknologi yang semakin berkembang dan digunakan dalam berbagai aplikasi. Namun, dalam konteks bahasa Indonesia, tantangan masih ada dalam hal akurasi dan performa. Oleh karena itu, penelitian ini menggunakan *Mel-Frequency Cepstral Coefficient* (MFCC) sebagai metode ekstraksi fitur suara dan *Learning Vector Quantization* (LVQ) sebagai metode klasifikasi untuk meningkatkan akurasi pengenalan ucapan.

Penelitian ini dilakukan melalui beberapa tahap, dimulai dari pengumpulan data suara dari lima partisipan yang mengucapkan lima kata sifat dalam bahasa Indonesia. Selanjutnya, dilakukan *preprocessing* data, termasuk normalisasi *amplitudo, padding, dan noise reductio*n untuk meningkatkan kualitas rekaman. Setelah itu, ekstraksi fitur MFCC dilakukan untuk mengubah sinyal suara menjadi vektor fitur yang lebih representatif. Data yang telah diekstraksi kemudian digunakan untuk melatih model LVQ, yang diuji menggunakan *classification report* sebagai metrik evaluasi, mencakup *precision, recall, dan F1-score.*

Hasil penelitian menunjukkan bahwa model dengan learning rate 0.01 dan epoch 200 menghasilkan akurasi tertinggi sebesar 91.45% dengan pembagian data 70:30. Evaluasi menggunakan *K-Fold Cross Validation* (K=5) menunjukkan rata-rata akurasi 82.53%, yang menandakan model memiliki generalisasi yang cukup baik. Analisis *classification report* juga menunjukkan bahwa model memiliki *F1-score* tinggi untuk beberapa kata, namun masih mengalami kesulitan dalam mengenali kata-kata dengan fitur suara yang mirip.

Kesimpulan dari penelitian ini menunjukkan bahwa MFCC efektif dalam mengekstraksi fitur suara bahasa Indonesia, dan LVQ mampu melakukan klasifikasi dengan akurasi yang cukup tinggi. Namun, beberapa kata masih sulit

dikenali dengan baik, sehingga penelitian selanjutnya disarankan untuk menggunakan dataset yang lebih besar dan beragam, mengeksplorasi metode lain seperti *Deep Learning*, serta mengembangkan sistem berbasis mobile atau web agar model ini dapat diterapkan dalam aplikasi nyata.

PRAKATA

Puji syukur penulis panjatkan ke hadirat Allah SWT. Atas berkat rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Laporan Skripsi yang berjudul "Pemanfaatan Metode *Mel Frequency Cepstral Coefficient* Dan *Learning Vector Quantization* Dalam Pengenalan Bahasa Indoensia" dengan baik. Tulisan ini adalah laporan hasil penelitian yang dilaksanakan mulai bulan Januari sampai bulan Mei 2025, sebagai salah satu syarat memperoleh gelar Sarjana Sains Terapan (SST) di Program Studi Teknik Informatika Jurusan Teknologi Informasi.

Pada kesempatan ini, Penulis menyampaikan ucapan terima kasih yang sebensar-besarnya kepada:

1. Syaiful Anwar, S.Tp. M.P., selaku Direktur Politeknik Negeri Jember.
2. Hendra Yufit Riskiawan, S.Kom. M.Cs., selaku Ketua Jurusan Teknologi Informasi.
3. Bety Etikasari, S.Pd, M.Pd. selaku Ketua Program Studi Teknik Informatika.
4. Ery Setiyawan Jullev Atmadji, S.Kom, M.Cs. selaku Dosen Pembimbing yang telah membimbing dan membantu dalam penyelesaian tugas akhir ini.
5. Elly Antika S.T., M.Kom. dan Zilvanhisna Emka Fitri, ST. MT selaku Dosen Penguji.
6. Rekan-rekan dan semua pihak yang telah ikut membantu pelaksanaan penelitian dan penulisan Skripsi.

Penulis menyadari bahwa Laporan Akhir/Skripsi ini masih jauh dari kesempurnaan, oleh karena itu, kritik dan saran yang membangun sangat Penulis harapkan. Semoga laporan ini dapat bermanfaat bagi para pembaca pada umumnya dan bagi Penulis pada khususnya.

Jember, 25 Maret 2025

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR KODE PROGRAM

Halaman

DAFTAR LAMPIRAN

Halaman

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Pada era teknologi ini yang semakin maju, mesin atau aplikasi pengenalan suara telah menjadi bagian dari kehidupan sehari-hari, baik dalam aplikasi *mobile* maupun perangkat lainnya. Teknologi pengenalan suara memungkinkan pengguna untuk berinteraksi dengan perangkat mereka secara verbal tidak menggunakan interaksi fisik, yang memungkinkan memfasilitasi aksesibilitas yang lebih efisien dan pengalaman pengguna yang lebih baik mulai dari pencarian informasi hingga navigasi. Dari assisten *virtual* seperti *Google Assistant, Amazon Alexa, dan Microsoft Cortana* hingga sistem transkrip otomatis seperti *Amazon Transcribe,* pengenalan suara telah membuka berbagai kemungkinan.

Pengenalan suara telah menjadi semakin penting dalam berbagai aplikasi, seperti pencarian *online*, pengaturan jadwal, dan *control* perangkat dalam rumah. Oleh karena itu, penting untuk memahami kinerja realtif dari model pengenalan suara yang seperti *Google*, terutama dalam konteks penggunaan bahasa Indonesia.

Pengenalan suara atau juga dikenal dengan pengenal ucapan, adalah teknologi yang memungkinkan komputer atau perangkat elektronik unttuk mengenali dan memahami kata-kata yang di ucapkan oleh manusia. Teknologi pengenalan suara berkerja dengan menerjemahkan gelombang suara yang direkam dari ucapan manusia menjadi *text* yang dapat di proses oleh komputer. Teknologi ini semakin canggih seiring waktu. Memungkinkan pengenalan suara yang lebih akurat dan *responsive*. Dengan pengenalan suara pengguna dapat melakukan berbagai tugas tanpa harus mengetik, seperti mencari informasi atau mengirim pesan teks. Ini membawa kenyamanan dan efesiensi yang besar dalam interaksi manusia dan teknologi.

Automatic Speech Recognition (ASR) biasanya digunakan untuk mengubah ucapan menjadi teks. Selain itu ASR juga digunakan unutk otentikasi biometrik, yang mengauntentikasi pengguna dari suara. Dari proses identifikasi atau

pengenalan ASR bisa digunakan untuk melakukan tugas berdasarkan intruksi yang dikenali (Mustikarini et al., 2019).

Penelitian mengenai ASR telah dilakukan selama lebih dari 40 tahun, namun hingga saat ini masih ada penelitian untuk menemukan *Automatic Speech Recognition* yang bisa mengenali ucapan dalam subjek apa pun dengan berbagai bahasa. Salah satu cara untuk meningkatkan tingkat pengenalan adalah dengan menggunakan model dari suatu bahasa itu perlu dikenali (Mustikarini et al., 2019).

Saat ini, *Mel-frequency cepstral coeffcients* paling umum digunakan dalam pengenalan suara dan verifikasi pembicara karena MFCC dapat bekerja baik pada masukan dengan tingkat korelasi yang tinggi, yaitu dengan menghilangkan informasi yang tidak diperlukan. Selain itu, MFCC bisa mewakili suara manusia dan sinyal music dengan baik karena MFCC menggunakan frekuensi *mel* (Mustikarini et al., 2019).

Learning vector Quantization (LVQ) adalah teknik dalam pemebelajaran mesin yang digunakan unutk mengklasifikasikan data ke dalam kategori atau kelas yang sesauai. LVQ menggunakan serangkaian *vector* prototipe yang mewakili setiap kelas yang mungkin. Selama proses pelatihan, vector-vektor ini disesuaikan agar sesuai dengan pola data pelatihan yang diberikan.

Cara terbaik untuk menguji kualitas beragam dari sistem *Automatic Speech Recognition* (ASR) adalah dengan menggunakan *classification report* sebagai metrik evaluasi utama. Dalam penelitian ini, *classification report* digunakan untuk mengevaluasi kinerja metode MFCC dan LVQ dalam mengenali ucapan. *Classification report* memberikan berbagai metrik penting, seperti *precision, recall, F1-score, dan support,* yang dapat menggambarkan seberapa baik sistem dalam mengklasifikasikan kata-kata yang diucapkan. *Precision* mengukur seberapa banyak prediksi yang benar dibandingkan dengan seluruh prediksi dalam suatu kelas, sedangkan *recall* menunjukkan seberapa banyak kata dalam kelas tertentu yang berhasil dikenali dengan benar oleh sistem. Sementara itu, *F1-score* adalah rata-rata harmonik dari *precision* dan *recall*, yang memberikan gambaran keseimbangan antara keduanya.

Penggunaan *classification report* akan memberikan gambaran yang lebih komprehensif tentang kinerja model dalam mengenali ucapan bahasa Indonesia. Dengan metrik seperti *precision, recall, dan F1-score,* evaluasi dapat dilakukan secara lebih detail untuk setiap kelas kata yang diuji. Hal ini memungkinkan analisis tidak hanya terhadap kesalahan dalam pengenalan kata, tetapi juga terhadap keseimbangan antara prediksi benar dan salah dalam setiap kategori. Dengan demikian, penelitian ini akan bermanfaat dalam mengembangkan sistem pengenalan ucapan yang lebih akurat dan dapat diandalkan dalam konteks bahasa Indonesia.

1.2 Rumusan Masalah

Berdasarkan latar belakang sebelumnya, didapatkan rumusan masalah sebagai berikut:

a. Bagaimana memanfaatkan metode *Mel-Frequency Cepstral Coeffcient* (MFCC) dalam implementasi model pengenalan suara untuk bahasa Indonesia?
b. Bagaimana penerapan metode *Learning Vector Quantization* (LVQ) dapat meningkatkan akurasi pengenalan ucapan dalam konteks bahasa Indonesia?
c. Sejauh mana penggunaan kombinasi MFCC dan LVQ dapat meningkatkan performa pengenalan suara untuk bahasa Indonesia berdasarkan metrik *precision, recall, dan F1-score* dalam *classification report*?

1.3 Tujuan

Tujuan penelitian ini adalah untuk menyelidiki pemanfaatan metode *Mel-Frequency Cepstral Coeffcient* (MFCC) dan teknik *Learning Vector Quantization* (LVQ) dalam implementasi model pengenalan suara untuk bahasa Indonesia, dalam mengenali ucapan dalam bahasa Indonesia. Selain itu, penelitian ini juga bertujuan untuk mengukur sejauh mana penggunaan kombinasi MFCC dan LVQ dapat memberikan akurasi terbaik dengan menggunakan evaluasi *classification report* sebagai metode evaluasi, sehingga memberikan kontribusi yang signifikan dalam pengembangan teknologi pengenalan suara yang lebih akurat dan andal untuk

meningkatkan efisiensi dan aksesibilitas dalam berbagai aplikasi teknologi sehari-hari.

1.4 Manfaat

Penelitian ini memiliki potensi yang mendukung dalam pengembangan aplikasi yang berbasis pengenalan suara yang lebih efektif dan efisien dalam berbagai konteks, meningkatkan kualitas layanan dan meningkatkan pengalaman pengguna secara keseluruhan

1.5 Batasan Masalah

Adapun batasan masalah yang dilakukan dalam penelitian ini yaitu:

a. Penelitian ini akan difokuskan pada penggunaan metode *Mel-Frequency Cepstral Coeffcient* (MFCC) dan teknik *Learning Vector Quantization* (LVQ) dalam implementasi model pengenalan suara untuk bahasa Indonesia, dengan tidak melibatkan teknik atau metode lain dalam analisis.
b. Penelitian ini akan difokuskan pada pengembangan dan evaluasi sistem pengenalan suara berbasis MFCC dan LVQ tanpa membandingkannya dengan platform pengenalan suara komersial lainnya. Evaluasi kinerja akan dilakukan tanpa mempertimbangkan aspek lain seperti kecepatan pengenalan atau fitur tambahan.
c. Penelitian ini akan membatasi pengujian dan evaluasi terhadap data ucapan dalam bahasa Indonesia yang telah terverifikasi dan terstruktur, tanpa memperluas cakupan pada bahasa lain atau data ucapan yang belum diverifikasi.

BAB 2. KAJIAN PUSTAKA

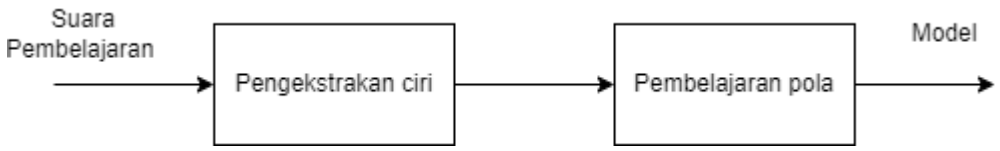
2.1 Pengenalan Suara

Suara adalah fenomena fisik yang dihasilkan oleh getaran benda atau getaran suatu benda yang berupa sinyal analog dengan amplitudo yang berubah secara kontinyu terhadap waktu. Suara atau bunyi biasanya merambat melalui udara. Suara atau bunyi tidak bisa merambat melalui ruang hampa Suara dihasilkan oleh getaran suatu benda. Selama bergetar, perbedaan tekanan terjadi di udara sekitarnya. Pola osilasi yang terjadi dinamakan sebagai "Gelombang". Manusia mendengar bunyi saat gelombang bunyi, yaitu getaran di udara atau medium lain, sampai ke gendang telinga manusia. Batas frekuensi bunyi yang dapat didengar oleh telinga manusia kira-kira dari 20 Hz sampai 20 kHz. Suara di atas 20 kHz disebut ultrasonik dan di bawah 20 Hz disebut infrasonik (Kristina et al., 2020).

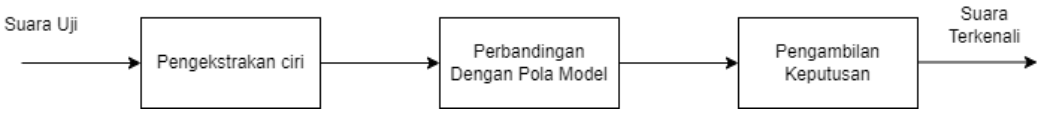
Para ahli mengatakan frekuensi suara pria berada pada rentang 65 hingga 260 Hertz. Sementara frekuensi suara wanita tercatat pada rentang 100 sampai 525 Hertz. Ini berarti pria dan wanita dengan frekuensi suara 100 sampai 260 Hertz seharusnya sulit dibedakan kalau hanya didengar dari suaranya saja (Tri Handoko, 2019).

Pengenalan suara merupakan salah satu upaya untuk dapat mengenali atau mengindentifikasi suara sehingga dapat dimanfaatkan untuk berbagai aplikasi. Teknologi ini bekerja dengan menerjemahkan gelombang suara yang di rekam dari ucapan manusia menjadi text yang dapat diproses oleh komputer (Seminar et al., 2012).

Secara umum tahap pengenalan suara dibagi menjadi dua bagian yakni tahap pembejalaran pola dan tahap pengenalan suara melalui pengenalan pola, blok diagram pembelajaran pola dan pengenalan pola ditunjukan pada Gambar 2.1 dan Gambar 2.2(Seminar et al., 2012).



Gambar 2. 1 Blok Diagram Pembelajaran Pola



Gambar 2. 2 Blok Diagaram Pengenalan Suara

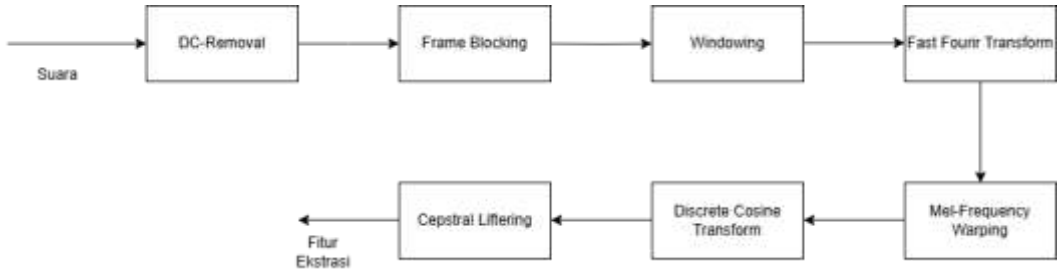
Berikut ini merupakan penjelasan dari masing-masing blok:

a. Pengekstrakan Ciri
   Ini adalah proses mendapatkan sederetan besaran pada bagian sinyal masukan untuk membuat pola pembelajaran atau pola uji. Untuk sinyal suara, ciri-ciri besaran biasanya berasal dari teknik analisis spektrum seperti MFCC (*Mel-Frequency Cepstral Coefficient*).
b. Pembelajaran Pola
   Satu atau lebih pola uji yang berhubungan dengan bunyi suara dari kelas yang sama, hasilnya yang dikenal sebagai "pola referensi", dapat menjadi sebuah model yang memiliki karakteristik ciri-ciri pola referensi dalam bentuk statistik.
c. Perbandingan dengan pola model
   Untuk menghitung kesamaan besaran antara pola data uji dan setiap kelas pola referensi, pola uji dikenali dibandingkan dengan setiap kelas pola referensi.
d. Pengambilan keputusan
   Bagian ini merupakan proses menentukan kelas pole referensi mana yang paling cocok untuk pola uji berdasarkan klasifikasi pola.

2.2 *Mel-Frequency Cepstral Coeffcient* (MFCC)

Dalam penelitian ini digunakan *Mel-Frequency Cepstral Coefficient* (MFCC) Merupakan salah satu teknik ekstrasi ciri (*Feature Extraction*) pada bidang *Speech Recognition* yang dilakukan dengan cara mengubah sinyal suara menjadi beberapa parameter yang dapat di analisis(Sidik Permana et al., 2018).

Pemrosesan MFCC di bagi menjadi 6 bagian, dan dapat dilihat pada Gambar 2.3.



Gambar 2. 3 Diagram Blok Proses MFCC

Pada Gambar 2.3 masing-masing proses pada diagaram MFCC akan di uraikan sebagai berikut:

a. *DC Removal*

Ditujukan untuk menghitung rata-rata dari data sampel suara dan mengurangi nilai setiap sampel suara dengan nilai rata-rata tersebut, tujuannya adalah membuang data-data yang tidak dibutuhkan (Sidik Permana et al., 2018).

b. *Frame Bloking*

Tahap *Frame Blocking* dalam ekstraksi MFCC adalah langkah awal yang penting dalam analisis suara. Dalam proses ini sinyal audio dibagi menjadi bingkai-bingkai (frame) waktu yang lebih kecil. Proses ini penting kerena memungkinkan kita untuk menganalisis sinyal secara terpisah pada waktu tertentu, yang sesusai dengan sifat-sifat temporal dari suara manusia. Setiap bingkai (frame) ini kemudian akan melalui proses-proses berikutnya yaitu *Windowing*.

c. *Windowing*

Setelah bingkai-bingkai (*frame*) terbentuk, langkah selanjutnya adalah menerapkan fungsi *Windowing* pada masing-masing bingkai. Ini bertujuan untuk mengurangi efek artefak yang muncul ketika sinyal dipotong, sehingga memastikan peralihan antar bingkai berjalan dengan mulus, sehingga setiap bingkai yang dihasilkan memiliki memiliki sifat spektral yang lebih konsisten dan terdefinisi dengan baik.

d. *Fast Fourier Transfrom* (FFT)

Setelah dilakukan *Windowing*, tahapan selanjutnya adalah *Fast Fourier Transform* (FFT), algoritma ini digunakan untuk mengubah sinyal suara dari domain waktu menjadi domain frekuensi. Dengan menerapkan FFT pada setiap bingkai sinyal yang telah diberi jendela (*Windowing*), selanjutnya dapat dilakukan analisis komponen frekuensi dari setiap bingkai tersebut.

e. *Mel Frequency Wrapping*

Pada tahap ini skala frekuensi dari hasil FFT diubah dari skala *Hertz* menjadi skala *mel*. Skala *mel* adalah skala yang berdasarkan pada presepsi

pendengaran manusia terhadap frekuensi suara. Manusia cenderung lebih peka terhadap perubahan frekuensi pada rentang frekuensi rendah dari pada frekuensi tinggi. Oleh karena itu, menggunakan skala *Mel* memungkinkan kita untuk lebih akurat mempresentasikan presepsi pendengaran manusia terhadap manusia

f. *Discrete Cosine Transform* (DCT)

Pada langkah ini mendapatkan nilai *cepstrum* MFCC yang masih ada didalam domain frekuensi maka *mel* frekuensi tersebut harus dikonversikan kembali menjadi domain waktu, karena untuk menentukan ciri akan mengacu pada urutan watu. Oleh sebab itu makan metode DCT akan mengubah nilai *cepstrum* MFCC yang masih berada dalam domain waktu menjadi domain frekuensi (Fitria et al., 2021).

g. *Cepstral Liftering*

Adalah proses pemfilteran koefisien MFCC untuk meningkatkan performa fitur yang dihasilkan, tanpa proses ini MFCC bisa didominasi di frekuensi rendah dan bisa menyebabkan kehilangan informasi di frekuensi tinggi, begitupun sebaliknya.

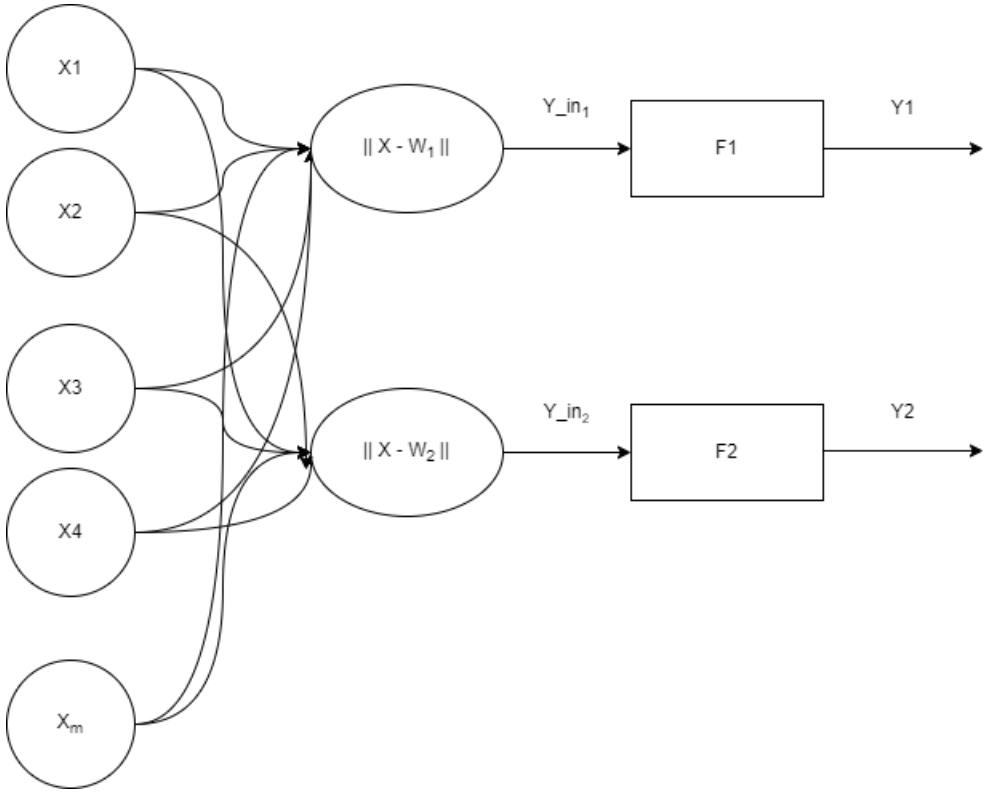
2.3 Jaringan Syaraf Tiruan (JST)

Jaringan Syaraf Tiruan (JST) merupakan model statistic yang struktur dan fungsinya mirip dengan otak manusia. *Neuron* tiruan pada Jaringan Syaraf Tiruan (JST) memiliki fungsi yang mirip dengan *neuron* pada sistem syaraf manusia (Wirawan Setialaksana et al., 2020).

Jaringan Syaraf Tiruan terdiri dari lapisan-lapisan *neuron* yang saling terhubung, dengan lapisan *input* yang menerima data, lapisan *ouput* yang menghasilkan *output* yang diinginkan, serta lapisan-lapisan tersembunyi yang berfungsi untuk melakukan pemrosesan *internal*. Proses belajar dalam jaringan syaraf tiruan melibatkan penyesuaian bobot-bobot, hal ini berdasarkan pada data latihan yang diberikan, sehingga jaringan dapat mempelajari pola dan hubungan dalam data dan menghasilkan prediksi atau *output* yang diharapkan.

2.4 *Learning Vector Quantization* (LVQ)

Dalam penelitian ini digunakan *Learning Vector Quantization* (LVQ) adalah suatu metode pelatihan untuk melakukan pembelajaran pada lapisan kompetitif yang terawasi (*supervised learning*) yang arsitektur jaringannya berlayer tunggal (*single layer*). Kelas-kelas yang didapatkan sebagai hasil dari lapisan kompetitif ini hanya tergantung pada jarak antara vektor-vektor input. Jika dua vektor input mendekati sama, maka lapisan kompetitif akan meletakkan kedua vektor input tersebut ke dalam kelas yang sama (Aren et al., 2022). Berikut adalah gambaran arsitektur dari LVQ,



Gambar 2. 4 Arsitektur LVQ

Berikut adalah langkah-langkah algoritma *Learning Vector Quantization* (Tawakal et al., 2020).

a. Inisialisai bobot awal
   1) Inisialisasikan bobot vector (W) secara acak. Setiap bobot vektor mewakili suatu kelas tertentu.

2) Parameter lain yang harus diatur adalah *learning rate (a)*, Konstanta pengurangan *learning rate(c)*, dan iterasi maksimum (*MaxEpoch)*.

b. Iterasi Pelatihan

1) Berikan input (*X*) ke jaringan LVQ
2) Hitung jarak antara input *X* dan bobot vektor w_i dinyatakan sebagai berikut:

$$||X - W_i|| = \sqrt{\sum_{j=1}^{m} (X_j - W_{ij})^2} \quad \text{……..….2.1}$$

3) Temukan bobot vektor terdekat dengan input (minimal jarak)
4) Perbarui bobot vektor terdekat berdasarkan aturan LVQ:
   - Jika input termasuk dalam kelas yang benar, perbarui bobot vektor dengan:

$$W_i(t+1) = W_i(t) + a \cdot (X - W_i(t)) \quad \text{……..….2.2}$$

   - Jika input termasuk dalam kelas yang salah, perbarui bobot vektor dengan:

$$W_i(t+1) = W_i(t) - a \cdot (X - W_i(t)) \quad \text{……..….2.3}$$

   Disini, t menunjukan iterasi pelatihan saat ini

c. Konvergensi dan Hasil Akhir

1) Ulangi langkah 2 hingga konvergensi (misalnya, perubahan bobot vektor yang kecil atau mencapai iterasi maksimum)
2) Setelah pelatihan selesai, bobot vektor akan mewakili kelas-kelas yang telah dipelajari oleh jaringan LVQ.

2.5 Classification Report

Classification Report adalah metrik evaluasi yang digunakan untuk mengukur kinerja model klasifikasi dengan menampilkan *precision, recall, F1-score,* dan *support* untuk setiap kelas. *Precision* menunjukan seberapa akurat prediksi positif dibuat, sedangkan *recall* mengukur seberapa banyak sampel positif yang berhasil di kenali oleh model, *F1-Socre* adalah rata – rata harmonik dari *precision* dan *recall*, yang digunakan untuk menyeimbangkan keduanya, sementara *support* menunjukan jumlah sampel sebenarnya dalam setiap kelas.

Metrik ini banyak digunakan dalam evaluasi model *machine learning.* Dengan *classification report,* pengguna dapat memahami peforma model secara leboh mendetail, termasuk mengetahui kelas mana yang memiliki prediksi kurang akurat sehingga dapat dilakukan perbaikan atau penyesuaian pada model.

2.6 State Of The Art

Table 2. 1 State Of the Art

No	Penulis	Masalah	Metode	Hasil	Poin
1	Liza Fitria, Kahirul Muttaqin, dan Muhammad Syahputra Nasution, tahun 2021	Kurangnya penjelasan sebelumnya	MFCC (*Mel Frequency Cepstral Coeffcien*)	Hasil penelitian pada sistem *speech recognition* menunjukan	Peneliti hanya menggunakan metode MFCC untuk mengenali suara ucapan
2	Tawakal, Firman Azkiya, dan Ahmedika, tahun 2020	Kesulitan membedak-an gejala antara DBD dan *Typhoid* yang mengakibat-kan salah diagnosis	LVQ (*Learning Vector Quantization*)	Hasil dari penelitian tersebut berhasil memperoleh tingkat akurasi sebesar 91,14% dengan nilai *Mean Square Error* sebesar 0.028571	Menambahkan variabel lain yang lebih muda diperoleh seperti gejala umum yang biasa diketahui langsung oleh pasien, diharapkan penelitian selanjutnya menggunakan metode yang

No	Penulis	Masalah	Metode	Hasil	Poin
					berbeda seperti *Backpropagat-ion*
3	Irham Sidik Permana, Youllia Indrawaty Nurhasana, Andriana Zulkarnain, tahun 2018	Tingkat akurasi yang beragam, peneliti menggunaka n DTW(*Dynamic Time Warping)* untuk berbagai jenis suara baik pria maupun wanita	MFCC (*mel Frequen cy Cepstral Coeffcie nt*), DTW (*Dynamic Time Warping)*	Hasil penelitian yaitu tingkat akurasi pada suara wanita sebesar 90% dan 80%, tingkat akurasi pada suara Pria adalah 80%,70% dan 60%	Peneliti menggunakan metode DTW untuk mengenali suara pria dan wanita
4	Melisa, Akim Manaor Hara Pardede, Marto Sihombing, tahun 2022	Membedakan gula aren asli dengan gula aren campuran dengan menggunaka n metode LVQ (*Learning Quantizati-on*)	LVQ (*Learnig Vector Quantiz-ation*)	Hasil dari penelitian tersebut menggunaka n metode LVQ dapat diterapkan untuk mengindentif ikasi gula aren asli dan campuran	Peneliti menggunakan metode LVQ dan ekstrasi ciri warna RGB, HSV, Citra *Grayscale*

No	Penulis	Masalah	Metode	Hasil	Poin
5	Nuruddin Wiranda, Agfianto Eko Putra, tahun 2019	Permasalahan komunikasi para tuna wisma yang diselesaikan dengan membuat sistem pengenalan pola suara yang menggunakan metode MFCC (*Mel – Frequency Cepstral Coeffcient*) dan *Backpropagation*	MFCC (*mel Frequecy Cepstral Coeffci-nt*), *Backpropagation*	Pada penelitian tesebut mengguna-kan 750 sampel suara yang terdiri dari 5 penutur, masing masing ada 30 kali pengulangan pengucapan kata, akurasi alat mencapai 50% pada pengujian data penutur 1,2,4 dan 5, serta 60% ketika mengguna-kan	Akurasi yang dihasilkan alat tersebut bisa di bilang rendah, perlu dilakukan penambahan fitur

BAB 3. METODE PENELITIAN

3.1 Waktu dan Tempat Pelaksanaan

Penelitian ini dilakukan di Gedung Jurusan Teknologi Informasi Politeknik Negeri Jember dengan kisaran waktu kurang lebih selama 8 bulan.

3.2 Alat dan Bahan

3.2.1 Alat

Ada dua jenis alat bantu dalam penyusunan ini, yaitu perangkat keras dan perangkat lunak, sebagai berikut:

a. Perangkat Keras
   1) Laptop Lenovo Thinkpad T540P *Processor* Intel Core I5-4300M CPU 2.60 GHZ (4 CPUs) 2.6 GHZ, Ram 8 GB, SSD 1 TB
   2) *Smartphone* Xiaomi Redmi Note 9 Pro
   3) M*ouse*

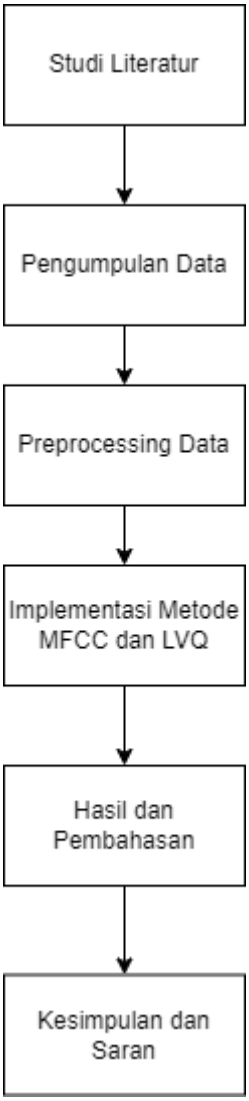
b. Perangkat Lunak
   1) Text Editor Visual Studio Code
   2) Word Office 2019
   3) Excel Office 2019
   4) Google Collab
   5) Adobe Audition 2022
   6) Python

3.2.2 Bahan

Bahan yang digunakan dalam penelitian ini berupa data suara yang terdiri dari pengucapan beberapa kata sifat dalam bahasa Indonesia. Data ini diperoleh dari sejumlah partisipan yang terdiri dari 3 laki-laki dan 2 perempuan. Rekaman suara dilakukan dalam kondisi terkendali untuk memastikan kualitas data yang baik sebelum diproses lebih lanjut menggunakan metode ekstraksi fitur MFCC.

3.3 Tahapan Penelitian

Tahapan dalam penelitian "Pemanfaatan Metode *Mel-Frequency Cepstral Coeffcient* dan *Learning Vector Quantization* Dalam Pengenalan Ucapan Bahasa Indonesia ditunjukan pada Gambar 3.1 dengan alur yang sudah dirancang.



Gambar 3. 1 Tahapan Penelitian

3.3.1 Studi Literatur

Pada tahapan ini dilakukan pengumpulan data terkait penelitian sebagai acuan yang bisa didapatkan dari berbagai sumber tertulis seperti buku, jurnal, artikel dan artikel ilmiah. Bahan acuan pada penelitian ini yaitu pengenalan ucapan manusia, sampel suara manusia, teknik pengekstrakan ciri suara *Mel Frequency Cepstral Coefficient* dan algoritma *Learning Vector Quantization* yang akan digunakan.

3.3.2 Pengumpulan Data Sampel Suara Manusia

Tahap pengumpulan data adalah tahap paling penting dalam sebuah penelitan. Pada tahap ini, peneliti melakukan pencarian data terkait, sebagai penunjang dalam proses penelitian, data yang digunakan pada penelitian ini adalah sampel suara manusia yang mengucapkan kata sifat. Data yang digunakan dalam penelitian ini merupakan data primer yang di peroleh langsung melalui rekaman suara manusia. Proses pengumpulan data dilakukan dengan merekam suara individu yang mengucapkan serangkaian kata sifat dalam bahasa Indonesia. Rekaman dilakukan dalam lingkungan yang terkendali dan tenang untuk memastikan kualitas data yang tinggi dan minim gangguan.

Pengambilan data akan menyesuaikan dengan penelitian sebelumnya, yang akan dilakukan terhadap 3 orang laki – laki dan 2 orang perempuan yang akan mengucapkan beberapa kata sifat berulang kali yang kemudian akan di simpan di format .wav, berikut adalah serangkaian kata sifat yang akan dipakai ditunjukan pada Table 3.1

Table 3. 1 Sample Kata Sifat

No	Sampel Kata
1	Baik
2	Bodoh
3	Licik
4	Rajin
5	Sombong

3.3.3 *Preprocessing Data*

Preprocessing data adalah tahap penting yang dilakukan untuk mempersiapkan data mentah agar siap digunakan dalam tahap ekstrasi ciri dan analisis lebih lanjut. Pada tahap ini, data suara yang telah dikumpulkan akan melalui serangkaian langkah untuk meningkatkan kualitas dan konsistensi data.

a. Normalisasi Amplitudo

Proses normalisasi amplitudo bertujuan untuk mengatur level volume dari setiap rekaman suara ke dalam rentang yang sama. Hal ini penting untuk memastikan bahwa variasi amplitudo tidak mempengaruhi hasil ekstrasi ciri.

b. Pemotongan dan *padding*

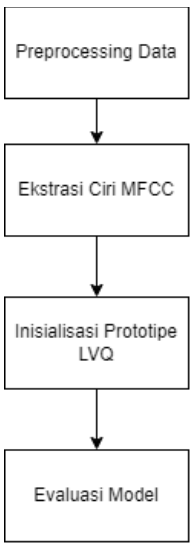
Langkah ini bertujuan untuk memastikan bahwa semua sampel suara memiliki durasi yang sama. Jika durasi rekaman lebih panjang dari yang diinginkan maka bagian awal atau akhir akan di potong. Sebaliknya jika durasi rekaman lebih pendek dari yang diinginkan maka bagian awal atau akhir akan diisi oleh *Zero Padding.*

c. *Noise Reduction*

Pengurangan *noise* dilakukan untuk menghilangkan suara latar belakang yang tidak di inginkan dari rekaman. Untuk metode pengurangan suara latar belakang bisa menggunakan filter digital seperti *low-pass* filter atau *high-pass* filter.

3.3.4 Implementasi Metode MFCC dan LVQ

Tahap pengembangan model untuk pengenalan ucapan bahasa Indonesia menggunakan *Mel-Frequency Cepstral Corefficient* dan *Learning Vector Quantization* di jelaskan pada Gambar 3.2.



Gambar 3. 2 Implementasi Metode

a. Preprocessing Data

Preprocessing data adalah langkah penting yang dilakukan untuk menyiapkan data mentah untuk digunakan dalam tahap ektrasi ciri fitur. Selama fase ini data audio yang dikumpulkan melewati serangkaian langkah untuk meningkatkan kualitas dan kosistensi data. Proses yang dilakukan yaitu normalisasi amplitudo, pemotongan atau *padding* dan *noise reduction.*

Normalisi Amplitudio bertujuan untuk mengatur volume dari setiap data audio suara ke dalam rentang yang sama. Hal ini penting untuk memastikan bahwa variasi amplitudo tidak mempengaruhi hasil ekstrasci ciri. Untuk alat yang dipakai adalah pemrograman *Pyhton* dengan menggunakan pustaka *Librosa.*

Pemotongan dan padding bertujuan untuk memastikan semua data audio memiliki panjang yang sama. Jika waktu perekaman lebih lama dari yang diperlukan, awal atau akhir akan dipotong. Namum, jika waktu perekaman lebih pendek dari yang diinginkan, maka akan diisi dengan angka nol di depan atau di belakang. Untuk alat yang dipakai adalah pemrograman Python dengan menggunakan pustaka *Librosa.*

Noise reduction digunakan untuk menghilangkan kebisingan latar belakang yang tidak diinginkan dari rekaman, untuk mengurangi kebisingan latar belakang, dapat menggunakan filter digital seperti *Low-pass* dan *High-pass* filter. Untuk alat yang dipakai adalah pemrograman *Python* dengan menggunakan pustaka *Librosa* dan *scipy.*

b. Ekstrasi Fitur Menggunakan MFCC

Pada tahap ini akan dilakukan ekstrasi fitur suara menggunakan *Mel-Frequency Cepstral Coeffcient* menggunakan data yang sudah di proses sebelumnya, berikut adalah tahapan ekstrasi fitur.

Dc Removal, Ditujukan untuk menghitung rata-rata dari data sampel suara dan mengurangi nilai setiap sampel suara dengan nilai rata-rata tersebut, tujuannya adalah membuang data-data yang tidak dibutuhkan (Sidik Permana et al., 2018).

Frame Blocking pada ekstrasi MFCC ini sampel suara akan dipotong menjadi *frame-frame* berdurasi lebih pendek. Pada sampel suara akan ditemui sinyal suara yang tidak stabil, sehingga akan susah mencari karakteristik dari suatu suara. Dengan memotong sampel menjadi *frame-frame* kecil, maka sinyal suara yang ada akan lebih stabil, sehingga akan lebih mudah mendapatkan karakteristik dari suara tersebut (Fitria et al., 2021).

Windowing, pada proses ini dilakukan *windowing* pada setiap *frame.* Hal ini tersebut dilakukan untuk meminimalisasi terjadinya *diskontinyu* sinyal di awal dan akhir dari tiap *frame* (Fitria et al., 2021).

Fast Fourier Transform, langkah selanjutnya adalah menerapkan *Fast Fourier Transform* sebuah logika yang mengubah sinyal audio dari domain waktu ke domain frekuensi. Dengan menerapkan FFT ke setiap kerangka (*frame*).

Mel Frequency Warpping, pada proses ini dilakukan filtering dari spectrum setiap *frame* yang di peroleh dari tahap sebelumnya. Sebelum melakukan filtering maka di tentukan batas atas dan bawah dari filter, sehingga nilai yang berada di luar dari batas tidak dilakukan filter. Kemudian keuda batas tersebut diubah ke dalam skala *mel* (Fitria et al., 2021).

Discrete Cosine Transform, Pada langkah ini mendapatkan nilai *cepstrum* MFCC yang masih ada didalam domain frekuensi maka *mel* frekuensi tersebut harus dikonversikan kembali menjadi domain waktu, karena untuk menentukan ciri akan mengacu pada urutan watu. Oleh sebab itu makan metode DCT akan mengubah nilai *cepstrum* MFCC yang masih berada dalam domain waktu menjadi domain frekuensi (Fitria et al., 2021).

c. Inisialisasi Prototipe LVQ

Pada tahap inisialisasi LVQ ini akan dijelaskan pada gambar berikut ini.

Gambar 3. 3 Tahap LVQ

Matrix Koefisien MFCC adalah hasil dari ekstrasi ciri fitur suara dari rekaman suara yang sudah di *preprocessing* sebelumnya menggunakan metode MFCC.

Langkah selanjutnya adalah melakukan labeling data, dimana setiap sampel data diberikan label kelas yang sesuai. Label ini digunakan untuk mengkategorikan data sehingga algoritma LVQ dapat mempelajari dan mengklasifikasi data berdasarkan kelas kelas tersebut.

Inisialisasi prototipe, adalah tahap dimana menentukan titik data dalam ruang fitur yang digunakan untuk memodelkan dan mengklasifikasikan kelas

kelas data yang berbeda, jadi setiap kelas akan mewakili setiap kelasnya untuk menjadi titik data dalam ruang fitur.

Langkah selanjutnya adalah melakukan prototipe awal untuk setiap kelas, setelah menentukan inisialiasi bobot awal maka tahap selanjutnya adalah melakukan perhitungan dengan rumus jarak *Euclidean distance* dengan persamaan berikut,

$$(P, Q) = \sqrt{(x2 - x1)^2 + (y2 - y1)^2} \qquad \text{……..….3.1}$$

Pelatihan LVQ adalah tahap dimana sistem belajar untuk mengklasifikasikan data berdasarkan fitur yang telah diekstrasi dan dilabeli. Pada tahap ini, prototipe yang telah diinisialiasi sebelumnya akan diperbarui secara iteratif untuk mencerminkan distribusi data yang sebenarnya. Tujuannya adalah mengoptimalkan posisi prototipe sehingga mereka dapat mempresentasikan kelas kelas data dengan lebih akurat.

Pada tahap terakhir bertujuan untuk mengevaluasi performa model yang telah dilatih. Proses validasi memastikan bawah model dapat menggeneralisasi dengan baik ke data yang belum pernah dilihat sebelumnya dan tidak hanya bekerja baik pada data latih.

3.3.5 Hasil dan Pembahasan

Pada tahap ini dimana output dari proses implementasi dan pengujian model *Learning Vector Quantization* dan *Mel Frequency Cepstral Coeffiecient*, akan mencakup presentasi hasil kinerja model dan analisis mendalam untuk memahami kekuatan dan kelamahan model. Dengan demikian tahap ini memberikan wawasan kritis yang diperlukan untuk menentukan efektivitas model dalam aplikasi nyata dan langkah yang mungkin diperlukan.

3.3.6 Kesimpulan dan Saran

Pada tahap akhir dari penelitian ini, tujuan utamanya adalah untuk menyajikan ringkasan hasil penelitian, menarik kesimpulan berdasarkan analisis

yang telah dilakukan, dan memberikan panduan dan wawasan yang bisa digunakan untuk penelitan berikutnya.

3.4 Jadwal Penelitian

Penelitian ini akan dilakukan selam kurang lebih 8 bulan dengan jadwal sebagai berikut.

Table 3. 2 Jadwal Penelitian

No	Keterangan	Bulan Ke-							
		1	2	3	4	5	6	7	8
1	Studi Literatur	■							
2	Pengumpulan Data		■	■					
3	*Preprocessing* Data			■	■				
4	Implementasi Metode				■	■	■	■	
5	Hasil dan Pembahasan							■	
6	Kesimpulan dan Saran								■

BAB 4. HASIL DAN PEMBAHASAN

4.1 Studi Litelatur

Pada tahap ini dilakukan kajian pada beberapa litelatur yang dapat mendukung pendekatan terkait dengan pemanfaatan metode *MFCC* (*Mel Frequency Cepstral Coeffcient*) dan LVQ (*Learning Vector Quantization*) yang diperoleh beberapa jurnal, artikel, skripsi, tugas akhir, dan tugas karya ilmiah yang lainnya. Selain itu, saran dari dosen pembimbing juga dapat menjadi referensi bagi penulis, setelah semuanya sudah tekumpul kemudian dilakukan pengumpulan data. Tujuan dilakukannya pengumpulan data terlebih dahulu adalah mempersiapkan data-data yang nantinya akan di proses menggunakan metode *MFCC* (*Mel Frequency Cepstral Coeffcient*) dan *LVQ* (*Learning Vector Quantization*).

4.2 Pengumpulan Data

Tahap yang paling penting dalam penelitian ini merupakan tahap pengumpulan data. Pengumpulan data diperlukan dalam penelitian ini adalah audio file yang berformat WAV. data yang digunakan merupakan data primer yang dikumpulkan secara langsung dari lima partisipan. partisipan terdiri dari tiga laki-laki dan dua perempuan. Pengambilan data dilakukan di dalam ruang tertutup untuk meminimalisir gangguan suara dari lingkungan sekitar.

4.2.1 Prosedur Pengambilan Data

Proses pengumpulan data dilakukan dengan langkah-langkah sebagai berikut:

a. Tahap pertama yaitu persiapan, sebelum pengambilan data, dilakukan pengecekan terhadap alat perekam suara untuk memastikan kualitas rekaman yang baik, partisipan juga juga diberikan intruski mengenai kata-kata yang harus diucapkan.
b. Tahap kedua yaitu proses perekaman, setiap partisipan diminta untuk mengucapkan lima kata sifat dalam bahasa Indonesia, yaitu “baik”, “bodoh”,

"licik", "rajin", dan sombong". Setiap kata diucapkan sebanyak 10 kali untuk memperoleh variasi dalam pengucapan.

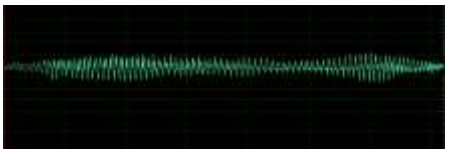
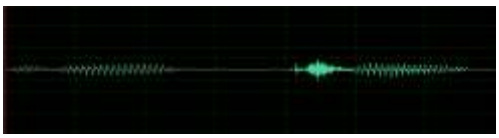
c. Proses penyimpanan semua data suara rekaman disimpan didalam format WAV dengan sampel rate 48 KHz untuk memastikan kualitas audio tetap tinggi.

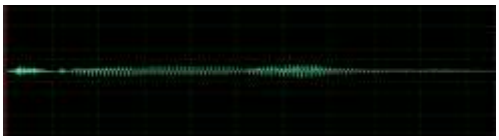
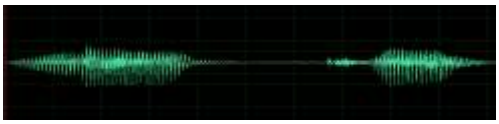
d. Tahap terakhir adalah pengelompokan data rekaman yang telah di kumpulkan disimpan dalam folder yang diberi nama sesuai dengan identitas partisipan untuk mempermudah proses analisis lebih lanjut.

4.2.2 Hasil Pengambilan Data

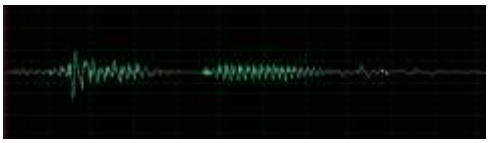
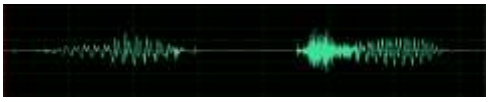
Hasil pengambilan data rekaman suara terdapat pada Tabel 4.1.

Table 4. 1 Table Hasil Pengambilan Dataset Data Rekaman

No	Rekaman Audio	Partisipan	Kata	Jumlah
1		Alvia	Baik	10
2		Alvia	Bodoh	10
3		Alvia	Licik	10
4		Alvia	Rajin	10

No	Rekaman Audio	Partisipan	Kata	Jumlah
5		Alvia	Sombong	10
6		Nania	Baik	10
7		Nania	Bodoh	10
8		Nania	Licik	10
9		Nania	Rajin	10
10		Nania	Sombong	10
11		IndraBagus	Baik	10
12		IndraBagus	Bodoh	10

No	Rekaman Audio	Partisipan	Kata	Jumlah
13		IndraBagus	Licik	10
14		IndraBagus	Rajin	10
15		IndraBagus	Sombong	10
16		MIndra	Baik	10
17		MIndra	Bodoh	10
18		MIndra	Licik	10
19		MIndra	Rajin	10

No	Rekaman Audio	Partisipan	Kata	Jumlah
20		MInda	Sombong	10
21		Ivan	Baik	10
		Ivan	Bodoh	10
22		Ivan	Licik	10
23		Ivan	Rajin	10
24		Ivan	Sombong	10

Data pada tabel 4.1 diatas didapatkan melalui *mincrophone* dari *smartphone Xiaomi Redmi Note 9 Pro.* Peneliti melakukan pengamatan sekaligus memeberi intruksi kepada partisipan agar kualitas suara tetap terjaga selama proses perekaman dan juga melakukan pengecekan setelah proses prekaman untuk menentukan rekaman audio sudah layak atau belum. Pengambilan rekaman suara juga dilakukan di tempat tertutup supaya hasil dari rekaman suara tidak terganggu oleh suara oleh lingkungan sekitar dan juga untuk mendapatkan hasil rekaman audio yang bagus.

4.3 *Preprocessing* Data

Pada tahapan ini, data suara yang telah dikumpulkan melalui proses perekaman perlu dipersiapkan agar siap digunakan dalam implementasi menggunakan metode *MFCC* (*Mel Frequency Cepstral Ceoffcient*) dan *LVQ* (*Learning Vector Quantizationi).* Proses *preprocessing* bertujuan untuk meningkatkan kualitas data serta memastikan bahwa setiap data memiliki format yang seragam.

Tahapan *preprocessing* yang dilakukan dalam penilitan ini mencakup beberapa langkah berikut:

4.3.1 Normaliasi Amplitudo

Normalisasi amplitude dilakukan untuk menyakaman tingkat intensitas suara pada setiap rekaman. Hal ini bertujuan untuk menghindari perbedaan volume suara yang dapat mempengaruhi hasil ekstrasi ciri fitur. Proses ini dilakukan dengan mengubah amplitude sinyal suara ke dalam rentang tertentu, sehingga setiap rekaman memiliki tingkat suara yang seimbang. Proses ini dilakukan secara otomatis dengan bantuan kode pemrograman *Python.*

Kode Program 4. 1 Normaliasi Amplitude

```
1)    def normalize_audio(y):
2)    """Normalisasi amplitudo antara -1 dan 1."""
3)    return y / np.max(np.abs(y))
```

Pada Kode Program 4.1, fungsi *normalize_audio* menerima sinyal yaitu *y* dan membaginya dengan nilai absolute tertinggi yang ada di sinyal tersebut, memastikan bawha nilai amplitudo tetap dalam rentang yang seragam untuk seluruh dataset. Berikut hasil dari normaliasi amplitudo.

Gambar 4. 1 Hasil Normaliasi Amplitudo

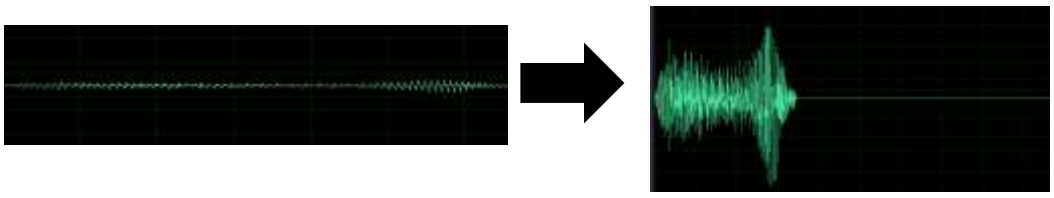
4.3.2 *Padding*

Setiap rekaman memiliki durasi yang bervariasi tergantung pada kecepatan pelafalan setiap individu, setiap semua data memiliki durasi dibawah 1 detik. Untuk memastikan bahwa setiap sampel memiliki panjang yang sama, dilakukan padding, yaitu menyesuaikan panjang rekaman menjadi 1 detik. Jika rekaman lebih pendek dari 1 detik, makan akan ditambahkan nilai nol atau *zero padding* di bagian akhir sinyal. Proses dilakukan secara otomatis menggunakan kode *python.*

Kode Program 4. 2 *Padding* atau Pemotongan

```
1)   def pad_audio(y, sr=SAMPLE_RATE,
  target_duration=TARGET_DURATION):
2)   """Menyesuaikan durasi audio dengan padding atau
  pemotongan."""
3)   target_length = int(sr * target_duration)  #
  Hitung jumlah sampel target
4)   if len(y) > target_length:
5)   return y[:target_length]  # Potong jika lebih
  panjang
6)   else:
7)   return np.pad(y, (0, target_length - len(y)),
  mode='constant')  # Tambahkan padding
```

Pada kode diatas *taget_length* mempunyai nilai 4800 dikarenakan rekaman atau dataset yang digunakan berada di sampling 48KHz dan durasi yang di inginkan adalah 1 detik, jika panjang sinyal kurang dari 4800 maka akan ditambahkan *zero padding* atau suara kosong di akhir sinyal. Berikut hasil dari proses padding



Gambar 4. 2 Hasil Proses *Padding*

4.3.3 *Noise Reduction*

Proses *noise reduction* diterapkan untuk menghilangkan atau mengurangi gangguan suara yang tidak di inginkan, seperti suara latal belakang atau dengungan

dari lingkungan sekitar. Proses ini berfungsi sebagai mendeteksi dan mengurangi frekuensi yang tidak diinginkan dari rekaman suara. Proses ini dilakukan secara otormatis menggunakan kode program *python.*

Kode Program 4. 3 Noise reduction

```
1)    def reduce_noise(y, sr=SAMPLE_RATE):
2)    """Mengurangi noise dengan noisereduce."""
3)    return nr.reduce_noise(y=y, sr=sr)
```

Pada kode diatas *prop_decreace* mempunyai nilai 0.8 yang berarti 80% dari noise yang terdeteksi akan dikurangi, sementara 20% tetap dipertahankan agar tidak menghilangkan terlalu banyak informasi dalam sinyal.

4.4 Implementasi Metode MFCC

Setelah melalui tahapan preprocessing data, data sudah siap untuk diekstraksi fiturnya menggunakan metode *MFCC (Mel Frequency Cepstral Coeffcient). MFCC* digunakan untuk mengekstrak fitur penting dari sinyal suara yang akan digunakan dalam pelatihan model *LVQ (Leraning Vector Quantization).*

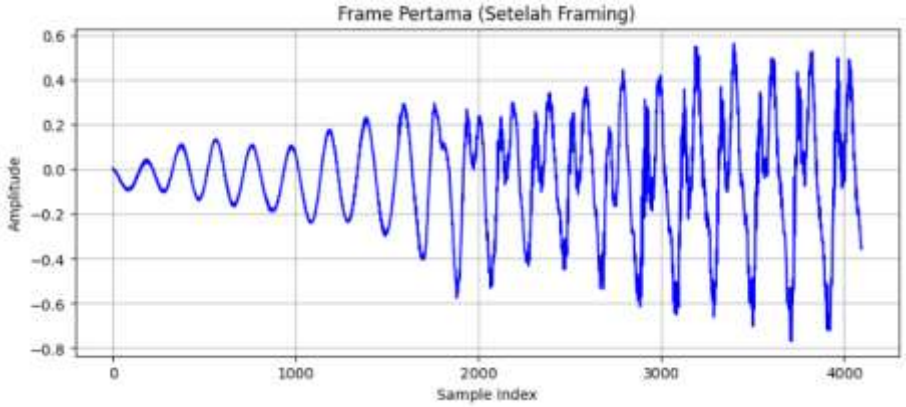
Mel Frequency Cepstral Coeffcient (MFCC) adalah salah satu teknik ekstrasi ciri (*Feature Extraction*) pada bidang *Speech Recognition* yang dilakukan dengan cara mengubah sinyal suara menjadi beberapa parameter yang dapat di analisis. Metode ini paling umum digunakan dalam pengenalan suara, *MFCC* meniru cara kerja telinga manusia dalam mengenali frekuensi suara. Berikut adalah proses dari *MFCC.*

4.4.1 *Dc Removal*

Dc Removal diperlukan agar menangani gangguan dalam sinyal suara, gangguan terjadi ketika sinyal suara tidak simetris terhadap garis nol, ada bagian dari sinyal suara yang tergeser ke atas atau kebawah. Solusi yang bisa di berikan adalah melewati proses *Dc Removal,* dalam penelitian ini akan dibantu dengan libray *pyton librosa,* yang nanti akan otomatis di perbaiki ketika memuat data sinyal suara.

4.4.2 *Frame Blocking*

Frame Blocking adalah proses membagi sinyal audio yang panjnag menjadi potongan – potongan kecil (*frame*) untuk di proses lebih lanjut. Berikut contoh hasil dari pemotongan tersebut.

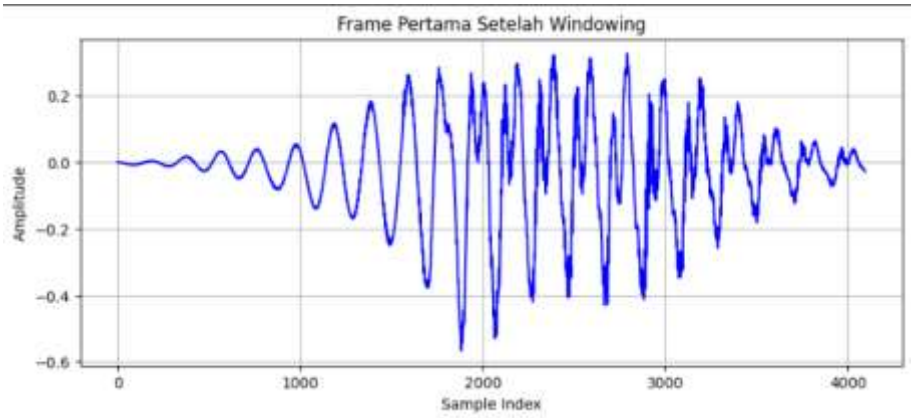


Gambar 4. 3 Hasil *Framming Frame* Pertama

Berdasarkan hasil yang ditampilkan pada Gambar 4.3, hasil dari *framing frame* pertama dari sinyal audio, panjang dari setiap frame di tentukan oleh nilai *FFT* (*Fast Fourier Transform)* dan *hop length, hop length* adalah seberapa sering mengambil frame baru dalam sinyal audio, jika nilai *hop length* lebih kecil maka *frame* akan lebih banyak dan lebi detail. Dalam kode ekstrasi fitur suara saya nilai *FFT* = 4096 dan *hop Length* = 512.

4.4.3 *Windowing*

Proses *windowing* dalam *MFCC* terjadi setalah melakukan proses *frame blocking* setalah sinyal audio dibagi menjadi *frame,* setiap frame perlu melalui proses ini sebelum memasuki proses *FFT.* Berikut adalah contoh hasil dari proses *Windowing.*

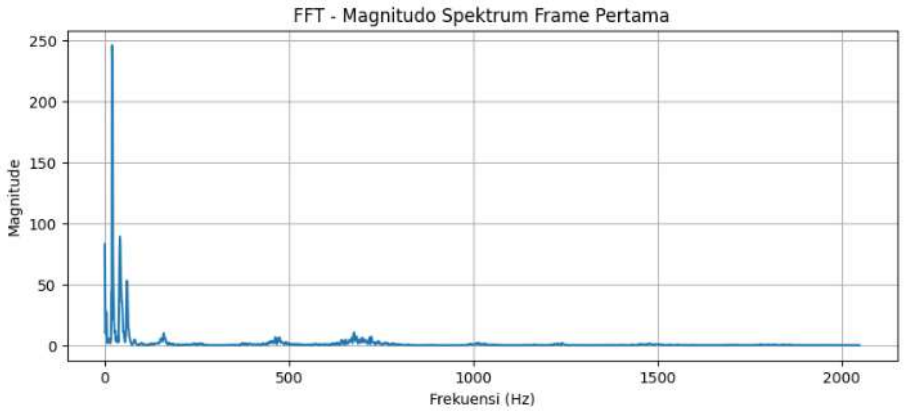


Gambar 4. 4 Hasil Proses Windowing

Berdasarkan Gambar 4.4, contoh hasil *windowing* pada *frame* pertama dari sinyal audio yang sudah di proses sebelumnya, proses ini mencegah ada kemungkinan dikontinuitas, yaitu perbedaan mendadak antara satu frame dan frame berikutnya, dikarenakan di awal dan di akhir tepi *frame* terjadi lonjakan tiba – tiba. Solusi yang bisa diberikan adalah melewati proses *windowing*, pada proses *windowing* ini menggunakan jenis *Hann Window* yang sudah dilakukan otomatis oleh *library* dari *python* yaitu *Librosa*.

4.4.4 *Fast Fourier Transform*

Proses berikutnya adalah *FFT (Fast Fourier Transfom),* berfungsi sebagai mengubah sinyal suara dari domain waktu (amplitudo dan waktu) menjadi domain frekuensi (kekuatan dan frekuensi). *FFT* membantu untuk melihat frekuensi dominan dalam sinyal suara, yang penting dalam analisi ucapan. Berikut adalah contoh hasil dari proses *FFT*.

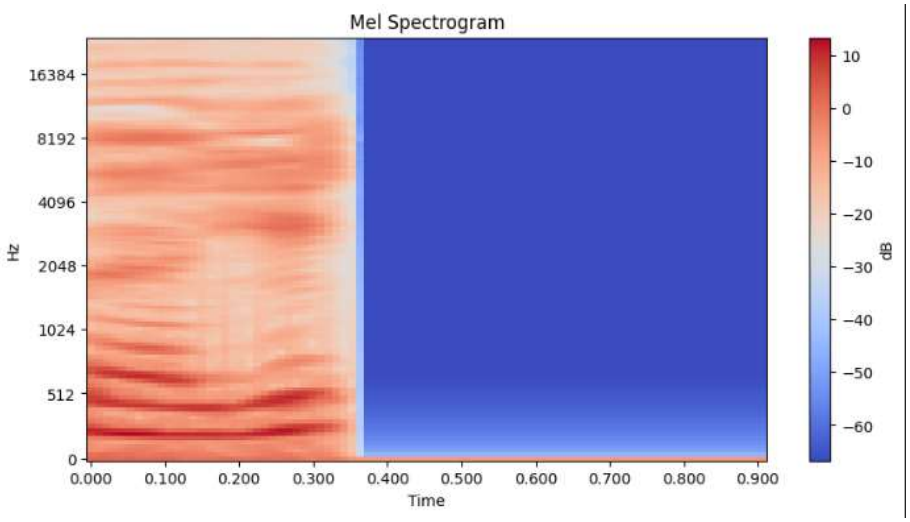


Gambar 4.5 Hasil Proses FFT

Berdasarkan Gambar 4.5, contoh hasil dari proses *FFT* dari *frame* pertama dari sinyal audio yang sudah di proses sebelumnya. *FTT* memungkinkan untuk bisa melihat dimana dominan frekuensi berada, dari berdasarkan dari hasil *FFT* dari *frame* pertama dominan frekuensi berada di frekuensi rendah.

4.4.5 *Mel Frequency Warping*

Mel Frequency Warping adalah proses mengubah skala frekuensi linear (*hz*) menjadi skala *mel.* Skala *mel* adalah skala frekuensi yang dirancang untuk meniru cara telinga manusia mempersepsikan perbedaan frekuensi suara. Berikut adalah contoh hasil dari proses *Mel Frequency Warping.*

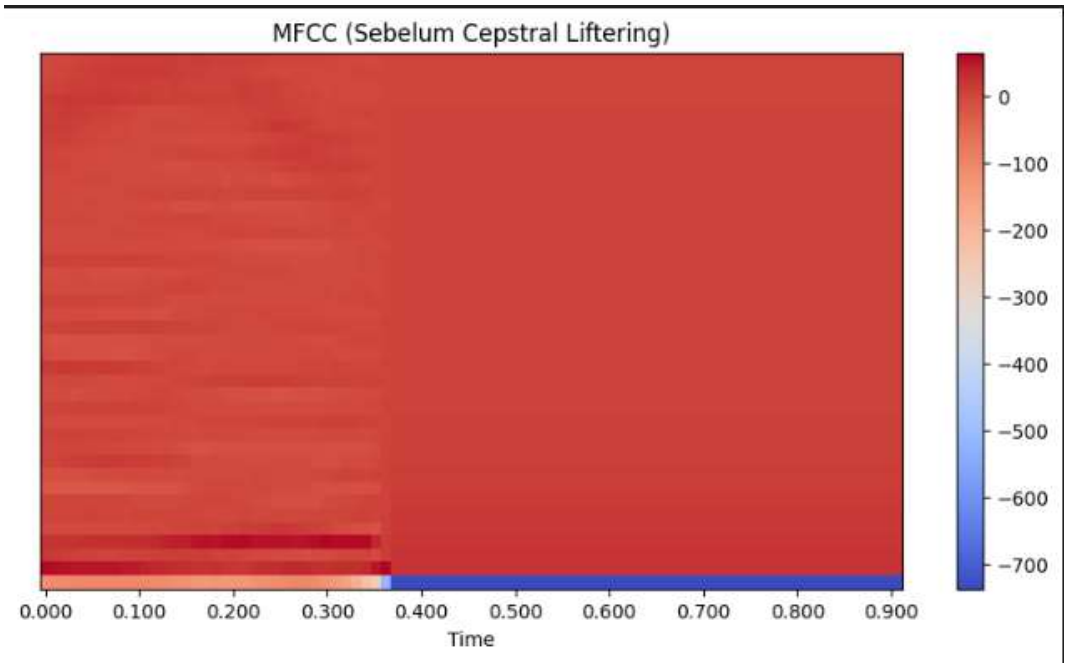


Gambar 4. 6 Hasil Proses *Mel Frequency Warping*

Dalam Gambar 4. 6 adalah contoh hasil dari proses *mel frequency warping* yang mengubah sinyal suara dari skala frekuensi ke skala *mel.* Dari gambar tersebut sumbu X (*horizontal*) menunjukan waktu dalam detik dan sumbu Y (*verticalI),* warna merah menunjukan energi (sinyal kuat) dan warna bitu menunjukan energi yang rendah. Dalam gambar tersebut suara hanya terlihat hanya sektiar 0.35 detik, hal tersebut terjadi karena dalam data suara pengucapan hanya terjadi dalam kurang dari 1 detik kemudian ketika memasuki tahap preprocessing semua data suara menyamakan durasi menjadi 1 detik semua. Seperti terlihat pada gambar energi besar berpusat pada frekuensi rendah yang sesuai dengan karakteristik suara manusia yang memiliki frekuensi dominan di rentang yang rendah.

4.4.6 *Discrete Cosine Transform*

Dicrate Cosine Transform (DCT) digunakan untuk mengkompresi informasi frekuensi dan menghasilkan fitur yang lebih ringkas dan representative yaitu *MFCC.* Hasil akhirnya adalah vektor *MFCC* yang mempresentasikan pola suara individu dalam berbicara, Vektor inilah yang nantinya digunakan seabgai fitur untuk klasifikasi suara dalam model *LVQ (Learning Vector Quantization).* Berikut adalah hasil dari proses DCT.

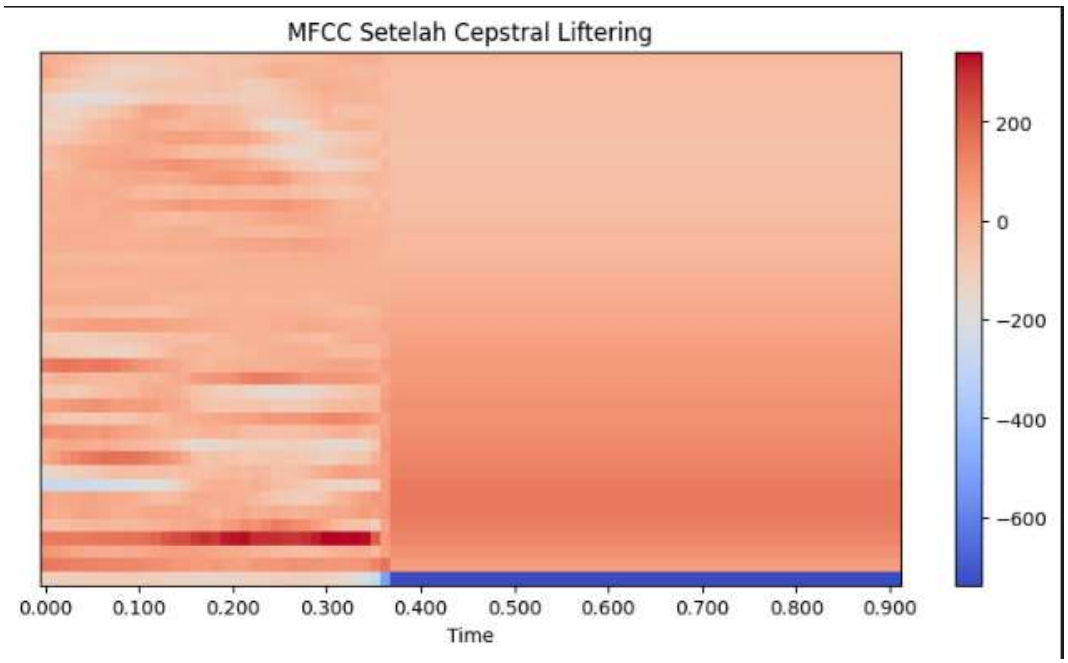


Gambar 4. 7 Hasil Proses DCT

Dari Gambar 4. 7 Adalah visual hasil dari proses DCT, dapat terlihat bahwa setelah 0.35 detik memiliki warna yang seragam atau sama tidak ada perubahan, ini terjadi karena proses *padding* yang dilakukan sebelumnya pada proses preprocessing data. Jika kita melihat di rentang waktu 0 – 0,35 detik kita bisa melihat terdapat area yang gelap terutama dibawah bagian spektrum, ini menunjukan bahwa komponen frekuensi rendah masih dominan yang sering terjadi dalam suara manusia karena energi utama terdapat di frekuensi rendah hingga menengah. Terlihat juga keseluruhan spektrum berwarna merah tapi masih terlihat dominasi nilai pada koefisien rendah dan kurangnya penekanan pada koefisien tinggi, oleh karena itu harus dilakukan proses *cepstral liftering* yang nantinya akan terlihat perbedaan yang jelas antar koefisien rendah dan tinggi.

4.4.7 *Cepstral Liftering*

Proses terakhir adalah *ceptral liftering* proses ini dilakukan untuk meningkatkan kualitas fitur *MFCC* dengan memperjelas informasi penting dan mengurangi pengaruh *noise* di koefisien tinggi. Berikut adalah hasil dari proses *cepstral liftering.*



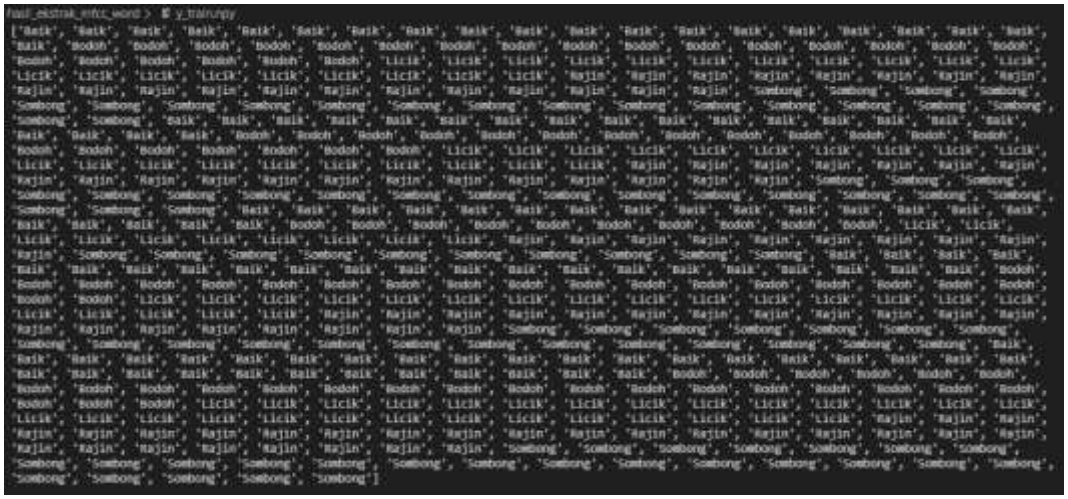
Gambar 4. 8 Hasil Proses Cepstral Liftering

Dari Gambar 4. 8 terlihat jika dibandingkan dengan *MFCC* sebelum proses *liftering*, tampak bahwa hasil setelah *liftering* lebih jelas dan lebih halus, bagian bawah spektrum terlihat lebih diperjelas yang menunjukan bawha informasi utama dalam suara lebih di pertajam dan *noise* di bagian atas spektrum tampak berkurang, sesuai dengan fungsi *liftering* yang melemahkan komponen tidak penting.

Setelah semua proses *MFCC* selesai, nilai *MFCC* yang diperoleh diambil dengan menghitung rata – rata tiap koefisien *MFCC* sepanjang waktu. Hasil rata – rata ini kemudian disimpan dalam format *numpy array* (.npy) untuk mempermudah penggunaan dalam proses pelatihan model *LVQ (Learning Vector Quantization).* Penyimpanan dalam format (.npy) dipilih karena format ini lebih efisien dibandingkan format teks atau csv, baik segi kecepatan pemrosesan maupun ukuran file. Berikut adalah hasil akhir dari ekstraksi fitur suara menggunakan MFCC.

Gambar 4. 9 Hasil Akhir Ekstrasi Fitur MFCC

Dari gambar 4. 9 adalah hasil akhir dari ekstrasi fitur suara menggunakan *MFCC* dapat terlihat dalam jumlah data yang berada dalam array berjumlah 40, dikarenakan dalam penelitian ini menggunakan nilai koefisien 40. Proses ini akan terus diulang ke semua dataset rekaman yang nantinya juga akan disimpan kedalam 1 file (.npy) beserta 1 file lagi yang berformat (.npy) yang berisi label dari rekaman suara.



Gambar 4. 10 Hasil Labeling Data Rekaman

Semua proses ini dilakukan dengan bantuan bahasa pemrograman *pyton* dan juga *library* yang disediakan yaitu *librosa* dan *numpy.* Proses selanjutnya adalah proses implementasi hasil ekstraksi fitur suara kedalam model *LVQ (Learning Vectir Quantization).*

Kode Program 4.4 Ekstraksi Fitur Suara MFCC

```
# Parameter MFCC
SAMPLE_RATE = 48000
N_MFCC = 40
N_FFT = 4096
HOP_LENGTH = 512
LIFTER = 22
# Fungsi untuk ekstraksi MFCC dari file audio
def extract_mfcc(file_path):
y, sr = librosa.load(file_path, sr=SAMPLE_RATE)
mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=N_MFCC,
    n_fft=N_FFT, hop_length=HOP_LENGTH, lifter=LIFTER)
return np.mean(mfcc, axis=1)  # Ambil rata-rata tiap koefisien
```

Pada Kode Program 4.4 adalah full kode program untuk mengekstrak fitur suara menggunakan *MFCC (Mel Frequency Cepstral Coefssien).* Dalam kode tersebut menggunakan bahasa pemrograman *python,* untuk menggunakan *MFCC python* menyediakan *library* yang bisa digunakan yang bernama *librosa.*

4.5 Implementasi Model *Learning Vector Quantization*

Proses klasifikasi digunakan untuk mengelompokkan data suara berdasarkan kata yang diucapkan dalam bahasa Indonesia. Tujuan dari proses ini adalah untuk mengetahui apakah model dapat mengenali kata yang diucapkan dengan akurasi yang baik. Proses klasifikasi terdiri dari dua tahap, yaitu tahap pelatihan dan tahap pengujian. Pada tahap pelatihan, model *Learning Vector Quantization* (*LVQ*) dibentuk dan dilatih untuk mengklasifikasikan fitur *MFCC* dari data latih ke dalam kategori kata yang telah ditentukan. Selanjutnya, pada tahap pengujian, model LVQ yang telah dilatih akan diuji menggunakan data yang berbeda dari proses pelatihan. Hasil dari tahap pelatihan dan pengujian kemudian akan dievaluasi untuk menilai sejauh mana model dapat mengenali kata yang diucapkan dengan benar.

Proses klasifikasi akan dibagi menjadi 5 kelas, melakukan proses klasifikasi kata yang di ucapkan. Terdapat 5 kelas yaitu:

a. Kelas 1 menyatakan kelas kata (Baik).
b. Kelas 2 menyatakan kelas kata (Bodoh).
c. Kelas 3 menyatakan kelas kata (Licik).
d. Kelas 4 menyatakan kelas kata (Rajin).
e. Kelas 5 menyatakan kelas kata (Sombong).

Pada penelitian ini, metode yang digunakan untuk proses pengenalan kata dalam ucapan bahasa Indonesia adalah *Learning Vector Quantization* (*LVQ*). *LVQ* merupakan sebuah arsitektur jaringan yang terdiri dari sekumpulan neuron yang saling terhubung dan membentuk pola tertentu dalam satu lapisan. Jaringan ini bekerja dengan cara mengelompokkan data berdasarkan fitur yang telah diekstraksi, dalam hal ini adalah koefisien *MFCC* dari rekaman suara. Struktur jaringan dalam *Learning Vector Quantization* (*LVQ*) dirancang untuk dapat membedakan pola fitur suara dari setiap kata yang diucapkan, sehingga dapat digunakan dalam proses klasifikasi kata yang lebih akurat.

4.5.1 *Input Layer*

Input layer merupakan bagian dari jaringan *Learning Vector Quantization* (*LVQ*) yang berfungsi untuk menerima nilai dari parameter yang akan digunakan

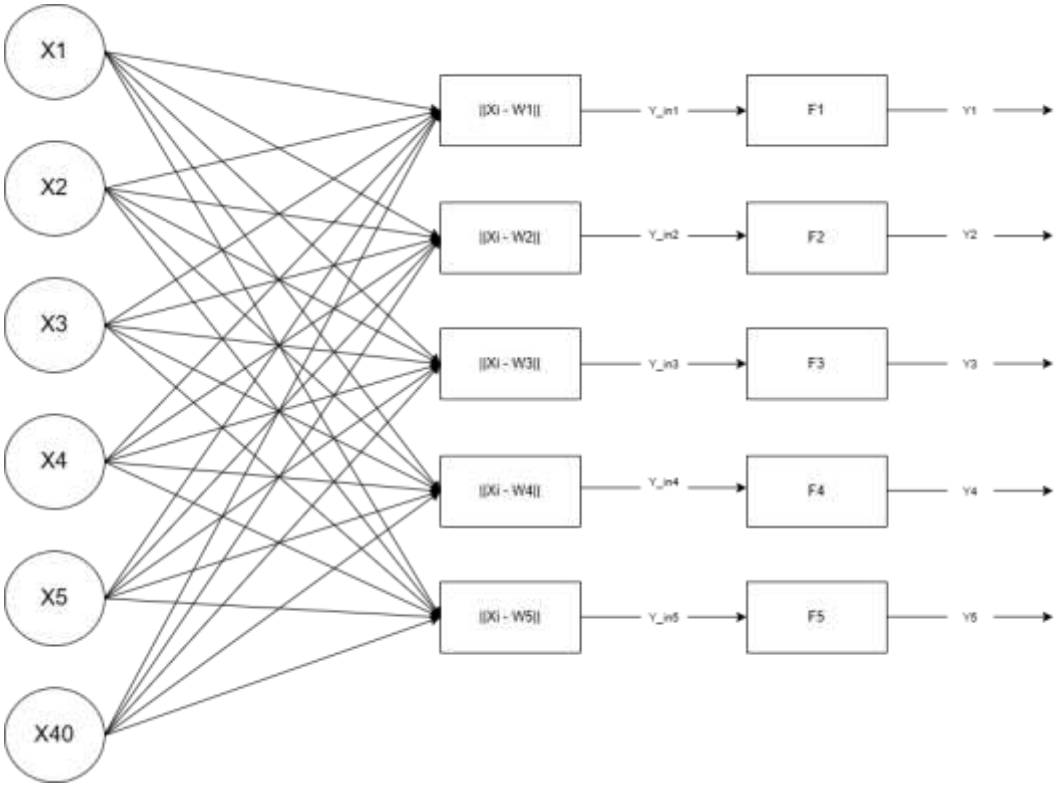
dalam proses klasifikasi. Lapisan ini dapat menerima sejumlah fitur sebagai masukan untuk membentuk representasi dari data suara yang akan dikenali. Dalam penelitian ini, jumlah masukan yang digunakan adalah koefisien *Mel Frequency Cepstral Coefficients* (*MFCC*), yang diperoleh dari proses ekstraksi fitur suara. Fitur-fitur ini kemudian diproses oleh jaringan LVQ untuk melakukan klasifikasi.

4.5.2 *Hidden Layer*

Hidden layer dalam jaringan *Learning Vector Quantization* (*LVQ*) adalah bagian yang berisi sejumlah neuron yang berperan dalam proses komputasi antara *input layer* dan *output layer*. Pada lapisan ini, vektor fitur yang telah diterima dari *input layer* akan dibandingkan dengan vektor prototipe yang telah ditentukan. *Neuron* pada *hidden layer* akan menghitung jarak antara fitur input dengan setiap vektor prototipe untuk menentukan kesesuaian dengan kategori yang ada. Proses pembaruan bobot juga terjadi di lapisan ini. Dalam penelitian ini, digunakan satu lapisan tersembunyi yang berfungsi untuk mengklasifikasikan data suara berdasarkan pola fitur MFCC yang telah diekstrak sebelumnya.

4.5.3 *Output Layer*

Output layer dalam *Learning Vector Quantization* (*LVQ*) adalah tahap terakhir dalam proses klasifikasi. Lapisan ini menerima hasil dari *hidden layer* dan menentukan kelas dari data input berdasarkan vektor prototipe terdekat. Berikut adalah arsitektur *LVQ* yang di gunakan di penelitian kali ini.



Gambar 4. 11 Arsitektur Jaringan LVQ

Keterangan:

X: Lapisan Masukan (*Input Layer)*

W: Vektor Bobot

F: Keluaran

4.5.4 Pelatihan Data Rekaman Suara

Pada tahap ini melakukan proses pelatihan pada data rekaman suara menggunakan *LVQ (Learning Vector Quantization).* Dataset yang digunakan terdiri dari 500 file rekaman suara, yang dikumpulkan dari lima partisipan, yaitu tiga laki – laki dan dua perempuan. Setiap partisipan mengucapkan sekitar 10 kali untuk setiap kata yang ada dalam dataset. Dalam proses pelatihan, model LVQ menggunakan beberapa parameter utama, yaitu jumlah prototipe per kelas, *learning rate* awal, serta jumlah *epoch* untuk menentukan hasil yang paling optimal untuk model. Kode program yang digunakan pada tahap pelatihan ini akan disajikan pada kode program berikut.

Kode Program 4.5 Kode LVQ

```
1)  class LVQ:
2)  def __init__(self, n_classes, n_prototypes=2,
    learning_rate=0.01, epochs=003200):
3)  self.n_classes = n_classes
4)  self.n_prototypes = n_prototypes
5)  self.learning_rate = learning_rate
6)  self.epochs = epochs
7)  self.prototypes = None
8)  self.prototype_labels = None
9)  def fit(self, X, y):
10) np.random.seed(42)
11) # Inisialisasi prototype
12) self.prototypes = []
13) self.prototype_labels = []
14) for label in np.unique(y):
15) idx = np.where(y == label)[0]
16) chosen = np.random.choice(idx, self.n_prototypes,
    replace=False)
17) self.prototypes.extend(X[chosen])
18) self.prototype_labels.extend([label] * self.n_prototypes)
19) self.prototypes = np.array(self.prototypes)
20) self.prototype_labels = np.array(self.prototype_labels)
21) # Training LVQ
22) for epoch in range(self.epochs):
23) for i in range(len(X)):
24) sample = X[i]
25) label = y[i]
26) # Cari prototype terdekat
27) distances = np.linalg.norm(self.prototypes - sample, axis=1)
28) winner_idx = np.argmin(distances)
29) # Update prototype
30) if self.prototype_labels[winner_idx] == label:
31) self.prototypes[winner_idx] += self.learning_rate * (sample -
    self.prototypes[winner_idx])
32) else:
33) self.prototypes[winner_idx] -= self.learning_rate * (sample -
    self.prototypes[winner_idx])
34) # Learning rate decay
35) self.learning_rate *= 0.95
36) def predict(self, X):
37) y_pred = []
38) for sample in X:
39) distances = np.linalg.norm(self.prototypes - sample, axis=1)
40) winner_idx = np.argmin(distances)
41) y_pred.append(self.prototype_labels[winner_idx])
42) return np.array(y_pred)
```

Kode Program 4.5 di atas adalah kode model *LVQ* yang dipakai di penelitian ini, kode tersebut memiliki tiga tahapan utama yaitu inisialisasi prototipe yang diambil 2 data disetiap kelasnya, *traning* yang menggunakan *epoch* 200, dan fungsi prediksi untuk mencari protoitpe terdekat untuk menentukan hasil klasifikasi.

Setelah dilakukan proses *traning* terhadap data latih, peneliti akan melakukan pembagian terhadap nilai data latih, data uji, *learning rate*, dan *epoch* yang berbeda. Perbandingan akurasi pelatihan pada 500 data menggunakan *learning vector quantization* dengan nilai *learning rate* dan *epoch* berbeda disajikan pada table berikut.

Table 4. 2 Perbandingan Parameter Training LVQ 1

Learning Rate	*Epoch*	Pembagian Dataset	Akurasi
0.01	50	9:1	90.20%
0.01	100	9:1	88.24%
0.01	200	9:1	88.24%
0.05	50	9:1	86.27%
0.05	100	9:1	86.27%
0.05	200	9:1	86.27%
0.1	50	9:1	88.24%
0.1	100	9:1	88.24%
0.1	200	9:1	88.24%

Pada Table 4.1 menunjukan bahwa proses traning menggunakan pembagian dataset sebesar 9:1 (90% data latih dan 10% datauji), dalam table tersebut akurasi tertinggi adalah 90.20% dengan parameter *learning rate* 0.01, *epoch* 50, dan akurasi terendah yang bisa didapat adalah 88.24% dengan parameter *learning rate* 0.01,.01, dan *epoch* 50,100,200. Untuk table berikutnya akan menggunakan pembagian dataset yang berebeda, tujuan untuk mengetahui parameter mana yang paling optimal yang dibutuhkan oleh model.

Table 4. 3 Perbandingan Parameter Traning LVQ 2

Learning Rate	*Epoch*	Pembagian Dataset	Akuasi
0.01	50	8:2	86.14%
0.01	100	8:2	85.15%
0.01	200	8:2	85.15%
0.05	50	8:2	83.17%
0.05	100	8:2	83.17%
0.05	200	8:2	83.17%
0.1	50	8:2	83.17%
0.1	100	8:2	83.17%
0.1	200	8:2	83.17%

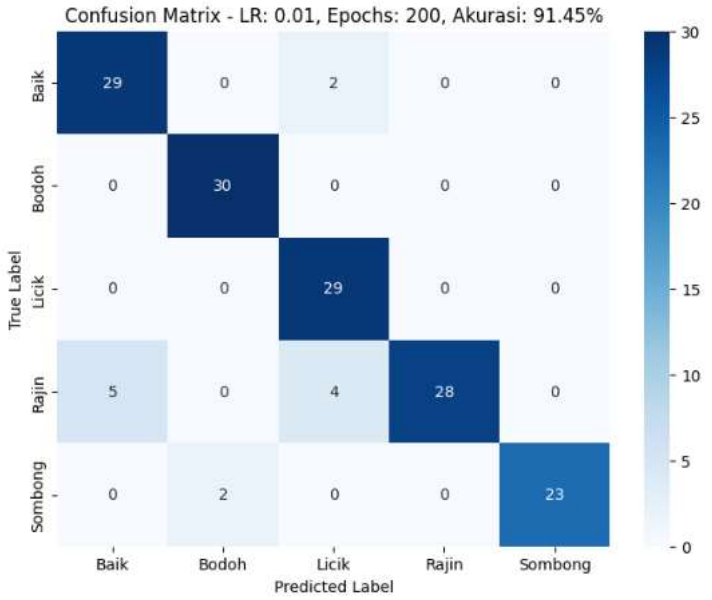
Pada Table 4.2 diatas menggunakan pembagian dataset 8:2 (80% data latih dan 20% data uji), untuk parameter yang digunakan sama seperti di table sebelumnya. Akurasi tertinggi yang bisa didapat adalah 85.15% dengan parameter *learning rate* 0.01, dan *epoch* 100 atau 200, sedankan akurasi terendah yang bisa didapat adalah 83.17% ada 6 hasil pengujian yang mendapatkan akurasi yang sama, dengan parameter *learning rate* 0.01, 0.05, 0.1 dan *epoch* 50,100,200.

Table 4. 4 Perbandingan Parameter Traning LVQ 3

Learning Rate	*Epoch*	Pembagian Dataset	Akurasi
0.01	50	7:3	90.13%
0.01	100	7:3	90.79%

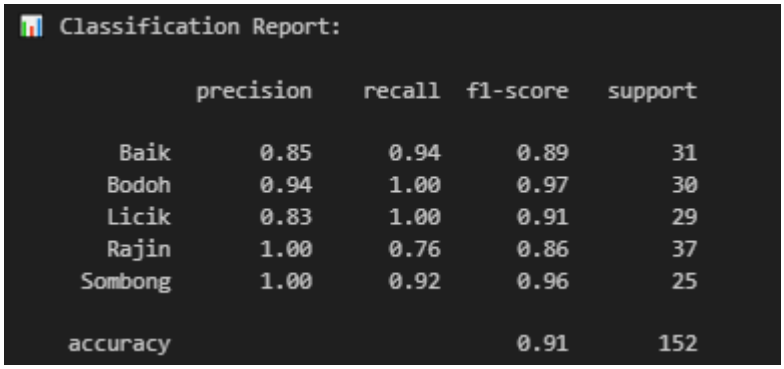
Learning Rate	*Epoch*	Pembagian Dataset	Akurasi
0.01	200	7:3	91.45%
0.05	50	7:3	91.45%
0.05	100	7:3	91.45%
0.05	200	7:3	91.45%
0.1	50	7:3	91.45%
0.1	100	7:3	91.45%
0.1	200	7:3	91.45%

Setelah mencoba berabagai kombinasi pembagian dataset, *learning rate* dan *epoch*, hasil yang paling bagus berada di Table 4.3 dengan akurasi 91.45%. Tahap selanjutnya yaitu melakukan proses pelatihan *LVQ* menggunakan parameter yang paling optimal yaitu menggunakan pembagian dataset sebesar 7:3 (70% menjadi data latih dan 30% menjadi data uji) *learning rate* 0.01 dengan *epoch* 200, untuk hasil lebih detail akan di sajikan pada gambar *confusion matrix* berikut.



Gambar 4. 12 *Confusion Matrix Training LVQ*

Pada gambar 4.12 diatas menunjukan bahwa metode *Learning Vector Quantization* dapat mengindentifikasi kata yang diucapkan berdasarkan fitur *MFCC* antar kelas yang berbeda. Hasil dari pelatihan data ini menghasilkan akurasi tertinggi sebesar 91.45% pada *learning rate* 0.01 dengan jumlah *epoch* 200, sementara itu, akurasi terendah diperoleh pada kombinasi *learning rate 0.01* dengan jumlah *epoch 50* dengan akurasi sebesar 90.13%. hal ini menunjukan bahwa parameter pelatihan sangat berpengaruh terhadap performa model dalam mengenalai ucapan. Berikut adalah hasil *classification report* yang menunjukan lebih detail tentang performa model.



Classification Report:

	precision	recall	f1-score	support
Baik	0.85	0.94	0.89	31
Bodoh	0.94	1.00	0.97	30
Licik	0.83	1.00	0.91	29
Rajin	1.00	0.76	0.86	37
Sombong	1.00	0.92	0.96	25
accuracy			0.91	152

Gambar 4. 13 *Classification Report* Hasil *Training LVQ*

Pada gambar 4.13 diatas hasil *classification report* model *learning vector quantization* yang digunakan untuk pengenalan ucapan menunjukan performa yang baik dengan akurasi sebesar 91%. Nilai *precision, recall,* dan *f1-score* menunjukan bahwa model dapat mengklasifikasikan sebagian besar kata dengan benar. Kelas “Bodoh” dan “Sombong” memiliki peforma terbaik dengan nilai *f1-socre* masing masin berada di 0.97 dan 0.96, yang menunjukan model sangat baik dalam mengenali kata-kata tersebut, namun, pada kelas “Rajin”, nilai *recall* hanya 0.76, yang berarti beberapa kata “Rajin” diklasifikasikan sebagai kata lain. Hal ini bisa disebabkan oleh kemiripan fitur suara *MFCC* dengan kelas lain.

Untuk memastikan peforma model setelah proses pelatihan dan evaluasi menggunakan *confussion matrix* dan *classification report,* peneliti akan menggunakan metode yang lain yaitu *K-fold Cross Valdiation.* Teknik ini digunakan untuk mengukur akurasi model dengan membagi dataset menjadi

beberapa bagian (*folds*), dimana di setiap *fold* secara bergantian digunakan sebagai data uji, semetara *fold* lainnya digunakan untuk pelatihan. Tujuan dari metode evaluasi ini adalah untuk mengetahui apakah model mengalami *overfitting* dan memberikan gambaran yang lebih akurat terkait peforma model secara keseluruhan. Peneliti menggunakan k=5, yang artinya dataset dibagi menjadi 5 bagian. Pemilihan nilai k=5 didasarkan pada penelitan sebelumnya yang serupa dalam bidang pengenalan suara menggunakan fitur MFCC (Dyarbirru & Hidayat, 2020).Untuk melakukan evaluasi ini peneliti dibantu dengan *library python* yang digunakan yaitu dari *sklearn.*

Table 4. 5 Hasil Evaluasi *Cross Validation*

Folds	Akurasi
1	84.16%
2	78.22%
3	88.12%
4	84.16%
5	78.00%
Rata - rata	82.53%

Berdasarkan Table 4. 4 evaluasi ini, model masih memiliki performa yang cukup baik dengan rata – rata akurasi di atas 80%, dengan akurasi paling tinggi pada *folds* ke 3 yaitu 88.12% dan yang terendah pada *folds* ke 78%.

4.5.5 Pengujian Model

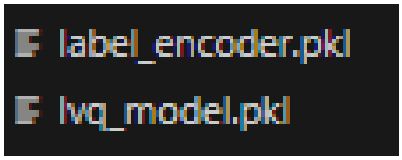
Setelah melalui proses pelatihan, model *Learning Vector Quantization* (*LVQ*) yang telah dibuat akan disimpan menggunakan *library Pickle* untuk memungkinkan penggunaan kembali tanpa perlu melakukan pelatihan ulang.

Kode Program 4. 6 Simpan Model LVQ

```
# Simpan model LVQ dengan pickle
with open("E:\! KULIAHHH\Ivano Kuliah\!SEMESTER 8\!SKRIPSI\Data Suara\lvq_model.pkl", "wb") as model_file:
pickle.dump(lvq_model, model_file)
# Simpan label encoder dengan pickle
with open("E:\! KULIAHHH\Ivano Kuliah\!SEMESTER 8\!SKRIPSI\Data Suara\label_encoder.pkl", "wb") as encoder_file:
```

```
6) pickle.dump(label_encoder, encoder_file)
```

Penyimpanan model dilakukan dalam dua file terpisah, yaitu lvq_model.pkl yang berisi bobot dan konfigurasi model LVQ, serta lvq_encoder.pkl yang menyimpan informasi tentang label encoder yang digunakan untuk mengubah label kelas menjadi format numerik selama proses pelatihan. Penyimpanan model dalam format ini memungkinkan kemudahan dalam implementasi dan pengujian, karena model dapat langsung dipanggil dan digunakan untuk melakukan prediksi pada data baru tanpa perlu melalui tahapan pelatihan kembali.



Gambar 4. 14 Hasil Penyimpanan Model

Pada tahap pengujian ini, model akan diuji menggunakan data baru yang bukan berasal dari dataset pelatihan. Pengujian ini bertujuan untuk mengevaluasi kemampuan generalisasi model dalam memprediksi data yang belum pernah dilihat sebelumnya, Hasil pengujian ini akan menjadi indikator penting untuk menilai kinerja model dan menentukan kesiapan model untuk diimplementasikan dalam aplikasi nyata. Berikut kode program yang dipakai.

Kode Program 4. 7 Untuk Prediksi Data Baru

```
1)   import numpy as np
2)   import pickle
3)   import librosa
4)   import noisereduce as nr
5)   import soundfile as sf
6)   import os
7)   #Definisikan ulang kelas LVQ agar bisa dipanggil ulang dari
     pickle
8)   class LVQ:
9)   def __init__(self, n_classes, n_prototypes=2,
     learning_rate=0.01, epochs=200):
10)  self.n_classes = n_classes
11)  self.n_prototypes = n_prototypes
12)  self.learning_rate = learning_rate
13)  self.epochs = epochs
14)  self.prototypes = None
15)  self.prototype_labels = None
```

```
16)  def fit(self, X, y):
17)  np.random.seed(42)
18)  self.prototypes = []
19)  self.prototype_labels = []
20)  for label in np.unique(y):
21)  idx = np.where(y == label)[0]
22)  chosen = np.random.choice(idx, self.n_prototypes,
     replace=False)
23)  self.prototypes.extend(X[chosen])
24)  self.prototype_labels.extend([label] * self.n_prototypes)
25)  self.prototypes = np.array(self.prototypes)
26)  self.prototype_labels = np.array(self.prototype_labels)
27)  for epoch in range(self.epochs):
28)  for i in range(len(X)):
29)  sample = X[i]
30)  label = y[i]
31)  distances = np.linalg.norm(self.prototypes - sample, axis=1)
32)  winner_idx = np.argmin(distances)
33)  if self.prototype_labels[winner_idx] == label:
34)  self.prototypes[winner_idx] += self.learning_rate * (sample -
     self.prototypes[winner_idx])
35)  else:
36)  self.prototypes[winner_idx] -= self.learning_rate * (sample -
     self.prototypes[winner_idx])
37)  self.learning_rate *= 0.95
38)  def predict(self, X):
39)  y_pred = []
40)  confidence_scores = []
41)  for sample in X:
42)  distances = np.linalg.norm(self.prototypes - sample, axis=1)
43)  winner_idx = np.argmin(distances)
44)  y_pred.append(self.prototype_labels[winner_idx])
45)  confidence_scores.append(1 / (1 + distances[winner_idx]))  #
     Confidence score berdasarkan jarak
46)  return np.array(y_pred), np.array(confidence_scores)
47)  #Load model LVQ dan Label Encoder
48)  MODEL_PATH = "lvq_model.pkl"
49)  ENCODER_PATH = "label_encoder.pkl"
50)  with open(MODEL_PATH, "rb") as model_file:
51)  lvq_model = pickle.load(model_file)
52)  with open(ENCODER_PATH, "rb") as encoder_file:
53)  label_encoder = pickle.load(encoder_file)
54)  # Parameter MFCC
55)  SAMPLE_RATE = 48000
56)  TARGET_DURATION = 1.0
57)  N_MFCC = 40
58)  N_FFT = 4096
59)  HOP_LENGTH = 512
60)  LIFTER = 22
```

```
# Fungsi preprocessing audio
def normalize_audio(y):
return y / np.max(np.abs(y))
def pad_audio(y, sr=SAMPLE_RATE,
target_duration=TARGET_DURATION):
target_length = int(sr * target_duration)
if len(y) > target_length:
return y[:target_length]
else:
return np.pad(y, (0, target_length - len(y)),
mode='constant')
def reduce_noise(y, sr=SAMPLE_RATE):
return nr.reduce_noise(y=y, sr=sr)
def trim_silence(audio, sr, threshold=0.02):
energy = np.abs(audio)
indices = np.where(energy > threshold)[0]
if len(indices) == 0:
return audio  # Jika tidak ada suara, kembalikan audio asli
start_index = indices[0]
trimmed_audio = audio[start_index:]
return trimmed_audio
# Fungsi untuk preprocessing & ekstraksi MFCC
def preprocess_and_extract_mfcc(file_path):
try:
# \ud83c\udfa7 Baca file dengan soundfile langsung (hindari
audioread error)
y, sr = sf.read(file_path)
# Cek kalau audio kosong
if len(y) == 0:
raise ValueError("File audio kosong atau rusak.")
#  Preprocessing
y = normalize_audio(y)
y = pad_audio(y, sr)
y = reduce_noise(y, sr)
# ✂️ Potong bagian diam di awal suara
y = trim_silence(y, sr, threshold=0.02)
# Ekstraksi fitur MFCC
mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=N_MFCC,
n_fft=N_FFT, hop_length=HOP_LENGTH, lifter=LIFTER)
return np.mean(mfcc, axis=1)
except Exception as e:
raise ValueError(f"Gagal memproses audio: {e}")
# Fungsi untuk prediksi suara
def predict_audio(file_path):
try:
mfcc_features = preprocess_and_extract_mfcc(file_path)
new_mfcc_features = np.expand_dims(mfcc_features, axis=0)
predicted_label, confidence_score =
lvq_model.predict(new_mfcc_features)
```

```
105) predicted_label = predicted_label[0]
106) confidence_score = confidence_score[0]
107) confidence_score_percent = round(confidence_score * 100, 2)
108) if confidence_score_percent < 0.5:
109) predicted_label = "suara tidak dikenali"
110) else:
111) predicted_label =
     label_encoder.inverse_transform([predicted_label])[0]
112) return {
113) "prediction": predicted_label,
114) "confidence_score": confidence_score_percent
115) }
116) except Exception as e:
117) return {"error": f"Gagal memproses audio: {e}"}
118) # Contoh pemanggilan fungsi prediksi
119) audio_file = "aditbaik2.wav"
120) hasil_prediksi = predict_audio(audio_file)
121) print(hasil_prediksi)
```

Kode Program 4.7 adalah implementasi dari model *LVQ* (*Learning Vector Quantization*) yang digunakan untuk melakukan prediksi suara berdasarkan fitur audio yang diekstraksi menggunakan *MFCC (Mel Frequency Cepstral Coefficient).* Berikut adalah penjelasan bagian-bagian penting dari kode tersebut:

a. Kelas *LVQ*, kelas *LVQ* disini di definisikan ulang karena *library pickle* membuthkan model *LVQ* dalam 1 script yang sama, agar pemanggilam model yang sudah disimpan bisa digunakan.

b. Pemanggilan Model yang sudah disimpan, Model *LVQ* yang telah dilatih disimpan dalam format file menggunakan library Pickle. Terdapat dua file yang disimpan, yaitu lvq_model.pkl yang berisi model *LVQ* beserta bobot dan konfigurasinya, serta lvq_encoder.pkl yang menyimpan informasi tentang label encoder. Label encoder digunakan untuk mengubah label kelas dari format string ke format numerik selama pelatihan dan sebaliknya selama prediksi. Dengan menyimpan model dalam format ini, model dapat dipanggil dan digunakan kembali tanpa perlu melalui proses pelatihan ulang.

c. *Preprocessing* audio, sebelum dialkukan ekstrasi fitur, audio akan mengalami beberapa tahap *preprocessing,* normaliasi audio, *padding, noise redecution.*

d. Ekstrasi Fitur *MFCC,* parameter yang digunakan dalam ektrasi *MFCC* meliputi *sample_rate,* jumlah koefisien *MFCC* (N_MFCC), ukuran window

FFT (N_FFT), HOP_LENGTH dan LIFTER. Untuk hasil ekstrasi kemudian diambil rata – rata untuk menghasilkan vektor fitur yang akan digunakan sebagai input model *LVQ.*

e. Prediksi audio, digunakan untuk melakukan prediksi pada file audio baru. Hasil prediksi berupa label kelas dan confidence score yang dihitung dalam bentuk persentase. Jika confidence score kurang dari 50%, hasil prediksi akan dianggap sebagai "suara tidak dikenali".

Pengujian model dilakukan dengan menggunakan data suara baru dari dari laki – laki dan perempuan, masing – masing akan mengucapkan 5 kata yang telat ditentukan, yaitu “baik”, “bodoh”, “licik”, “rajin”, “sombong”. Setiap kata akan diucapkan sebanyak 2 kali yang berarti satu orang ada 10 sampel jika di total kan dengan 2 orang total menjadi 20 sampel kata. Proses pengujian dimulai dengan melakukan tahapan *preprocessing* pada setiap file audio, selanjutnya ekstrak fitur suara menggunakan *MFCC,* fitur yang dihasilkan akan digunakan input untuk model *LVQ* yang tekah dilatih. Hasil prediksi dari model akan dibandingkan dengan label sebenarnya untuk mengevaluasi akurasi prediksi. *Table* pengujian akan disajikan pada *table* berikut:

Table 4. 6 Hasil Testing Data Baru Suara Laki-Laki

No	Kata	Hasil Prediksi	*Confidence Score*	Keterangan
1	Baik	Baik	85%	Benar
2	Baik	Baik	80%	Benar
3	Bodoh	Suara Tidak Dikenali	49%	Salah
4	Bodoh	Bodoh	53%	Benar
5	Licik	Licik	53%	Benar
6	Licik	Suara Tidak Dikenali	47%	Salah
7	Rajin	Rajin	62%	Benar

No	Kata	Hasil Prediksi	*Confidence Score*	Keterangan
8	Rajin	Rajin	74%	benar
9	Sombong	Sombong	74%	Benar
10	Sombong	Sombong	96%	Benar

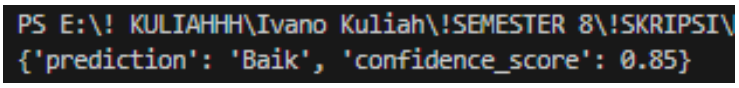
Pada Pengujian ini peneliti melakukan pendekatan berbeda ketika perekaman untuk data uji seperti variasi dalam pengucapan, intonasi dan kondisi lingkungan saat perekaman yang mempengaruhi kualitas audio. Dalam kode model peneliti menambahkan sebuah kondisi yang dimana jika hasil tingkat *confidence socre* berada dibawah 50% maka *output* yang dihasilkan adalah "Suara Tidak Dikenali". Hasil testing menunjukkan bahwa model LVQ mampu menghasilkan prediksi yang benar meskipun dengan tingkat *confidence score* yang bervariasi, mulai dari 96% hingga 47%. Hal ini mengindikasikan bahwa model memiliki kemampuan untuk mengenali pola suara meskipun dengan tingkat kepercayaan yang berbeda-beda. Namun, *confidence score* yang rendah menunjukkan bahwa model kurang yakin dengan prediksinya, meskipun hasil prediksi tersebut ternyata benar. Hal ini dapat terjadi karena beberapa faktor, seperti variasi dalam pengucapan, intonasi, atau kondisi lingkungan saat perekaman.

Table 4. 7 Hasil Testing Data Baru Suara Perempuan

No	Kata	Hasil Prediksi	Confidence Score	Keterangan
1	Baik	Baik	87%	Benar
2	Baik	Baik	82%	Benar
3	Bodoh	Bodoh	72%	Benar
4	Bodoh	Suara Tidak Dikenali	49%	Salah
5	Licik	Licik	61%	Benar
6	Licik	Licik	62%	Benar
7	Rajin	Rajin	58%	Benar

No	Kata	Hasil Prediksi	Confidence Score	Keterangan
8	Rajin	Suara Tidak Dikenali	49%	benar
9	Sombong	Sombong	52%	Benar
10	Sombong	Suara Tidak Dikenali	47%	Benar

Pada Table 4.6 diatas hasil dari pengujian kurang lebih sama dengan table pengujian utnuk suara laki – laki yang dimana saat perekaman untuk data uji melakukan variasi dalam pengucapan, intonasi dan kondisi lingkungan. Meskipun confidence rendah, model masih bisa memprediksi kata dengan benar karena mengenali pola suara yang mirip berdasarkan pelatihan sebelumnya. Misalnya, kata "Rajin" yang diucapkan dalam kondisi bising mungkin hanya mendapatkan *confidence score* 0.49, tetapi tetap dipilih sebagai hasil akhir karena masih memiliki probabilitas tertinggi dibanding alternatif lainnya. Dengan demikian, meskipun kualitas audio mempengaruhi tingkat keyakinan model, prediksi kata tetap dapat akurat berdasarkan pola suara yang dikenali. Berikut adalah contoh hasil output model yang dihasilkan kedalam terminal.



Gambar 4. 15 Output Model Di Terminal

Untuk pengujian selanjutnya peneliti membuat sebuah API dengan bantuan *library flask* dari *python.* Selain untuk pengujian, pembuatan API juga akan membantu dalam implementasi ke lingkungan yang lebih luas. Untuk alat yang digunakan pengujiaan ini adalah Postman, sebuah tool yang memungkinkan pengiriman request HTTP secara interaktif, sehingga memudahkan validasi dan analisis hasil prediksi. Berikut adalah kode API *Flask* yang digunakan,

Kode Program 4. 8 API Model Flask

```
1)   from flask import Flask, request, jsonify
2)   import numpy as np
3)   import pickle
```

```
import librosa
import noisereduce as nr
import soundfile as sf
import os
app = Flask(__name__)
#  Definisikan ulang kelas LVQ agar bisa dipanggil ulang dari pickle
class LVQ:
def __init__(self, n_classes, n_prototypes=2, learning_rate=0.01, epochs=200):
self.n_classes = n_classes
self.n_prototypes = n_prototypes
self.learning_rate = learning_rate
self.epochs = epochs
self.prototypes = None
self.prototype_labels = None
def fit(self, X, y):
np.random.seed(42)
self.prototypes = []
self.prototype_labels = []
for label in np.unique(y):
idx = np.where(y == label)[0]
chosen = np.random.choice(idx, self.n_prototypes, replace=False)
self.prototypes.extend(X[chosen])
self.prototype_labels.extend([label] * self.n_prototypes)
self.prototypes = np.array(self.prototypes)
self.prototype_labels = np.array(self.prototype_labels)
for epoch in range(self.epochs):
for i in range(len(X)):
sample = X[i]
label = y[i]
distances = np.linalg.norm(self.prototypes - sample, axis=1)
winner_idx = np.argmin(distances)
if self.prototype_labels[winner_idx] == label:
self.prototypes[winner_idx] += self.learning_rate * (sample - self.prototypes[winner_idx])
else:
self.prototypes[winner_idx] -= self.learning_rate * (sample - self.prototypes[winner_idx])
self.learning_rate *= 0.95
def predict(self, X):
y_pred = []
confidence_scores = []
for sample in X:
distances = np.linalg.norm(self.prototypes - sample, axis=1)
winner_idx = np.argmin(distances)
y_pred.append(self.prototype_labels[winner_idx])
```

```
47)  confidence_scores.append(1 / (1 + distances[winner_idx]))   #
     Confidence score berdasarkan jarak
48)  return np.array(y_pred), np.array(confidence_scores)
49)  #  Load model LVQ dan Label Encoder
50)  MODEL_PATH = "lvq_model.pkl"
51)  ENCODER_PATH = "label_encoder.pkl"
52)  with open(MODEL_PATH, "rb") as model_file:
53)  lvq_model = pickle.load(model_file)
54)  with open(ENCODER_PATH, "rb") as encoder_file:
55)  label_encoder = pickle.load(encoder_file)
56)  #  Parameter MFCC
57)  SAMPLE_RATE = 48000
58)  TARGET_DURATION = 1.0
59)  N_MFCC = 40
60)  N_FFT = 4096
61)  HOP_LENGTH = 512
62)  LIFTER = 22
63)  #  Fungsi preprocessing audio
64)  def normalize_audio(y):
65)  return y / np.max(np.abs(y))
66)  def                pad_audio(y,                sr=SAMPLE_RATE,
     target_duration=TARGET_DURATION):
67)  target_length = int(sr * target_duration)
68)  if len(y) > target_length:
69)  return y[:target_length]
70)  else:
71)  return np.pad(y, (0, target_length - len(y)), mode='constant')
72)  def reduce_noise(y, sr=SAMPLE_RATE):
73)  return nr.reduce_noise(y=y, sr=sr)
74)  def trim_silence(audio, sr, threshold=0.02):
75)  energy = np.abs(audio)
76)  indices = np.where(energy > threshold)[0]
77)  if len(indices) == 0:
78)  return audio  # Jika tidak ada suara, kembalikan audio asli
79)  start_index = indices[0]
80)  trimmed_audio = audio[start_index:]
81)  return trimmed_audio
82)  #  Fungsi untuk preprocessing & ekstraksi MFCC
83)  def preprocess_and_extract_mfcc(file_path):
84)  try:
85)  #  Baca file dengan soundfile
86)  y, sr = sf.read(file_path)
87)  # Cek kalau audio kosong
88)  if len(y) == 0:
89)  raise ValueError("File audio kosong atau rusak.")
90)  #  Preprocessing
91)  y = normalize_audio(y)
92)  y = pad_audio(y, sr)
93)  y = reduce_noise(y, sr)
```

```
94)  #  Potong bagian diam di awal suara
95)  y = trim_silence(y, sr, threshold=0.02)
96)  #  Ekstraksi fitur MFCC
97)  mfcc   =   librosa.feature.mfcc(y=y,   sr=sr,   n_mfcc=N_MFCC,
     n_fft=N_FFT, hop_length=HOP_LENGTH, lifter=LIFTER)
98)  return np.mean(mfcc, axis=1)
99)  except Exception as e:
100) raise ValueError(f"Gagal memproses audio: {e}")
101) #  Route API untuk prediksi suara
102) @app.route("/predict", methods=["POST"])
103) def predict():
104) if "file" not in request.files:
105) return jsonify({"error": "Tidak ada file yang dikirim"}), 400
106) file = request.files["file"]
107) if file.filename == "":
108) return jsonify({"error": "Nama file kosong"}), 400
109) # Simpan file sementara
110) file_path = "temp_audio.wav"
111) file.save(file_path)
112) # Cek ukuran file
113) if os.path.getsize(file_path) == 0:
114) return jsonify({"error": "File audio kosong atau rusak"}), 400
115) try:
116) #  Preprocessing & Ekstraksi MFCC
117) mfcc_features = preprocess_and_extract_mfcc(file_path)
118) new_mfcc_features = np.expand_dims(mfcc_features, axis=0)
119) #  Prediksi dengan LVQ
120) predicted_label,              confidence_score              =
     lvq_model.predict(new_mfcc_features)
121) predicted_label = predicted_label[0]
122) confidence_score = confidence_score[0]
123) # Ubah confidence score ke persentase dan bulatkan ke 2 angka
     di belakang koma
124) confidence_score_percent = round(confidence_score * 100, 2)
125) # Jika confidence score kurang dari 50%, kembalikan "suara
     tidak dikenali"
126) if confidence_score_percent < 0.5:
127) predicted_label = "suara tidak dikenali"
128) else:
129) predicted_label                                            =
     label_encoder.inverse_transform([predicted_label])[0]
130) # Hapus file sementara
131) os.remove(file_path)
132) #  Hasil Prediksi
133) return jsonify({
134) "prediction": predicted_label,
135) "confidence_score":  confidence_score_percent    #  Confidence
     score dalam persentase
136) })
```

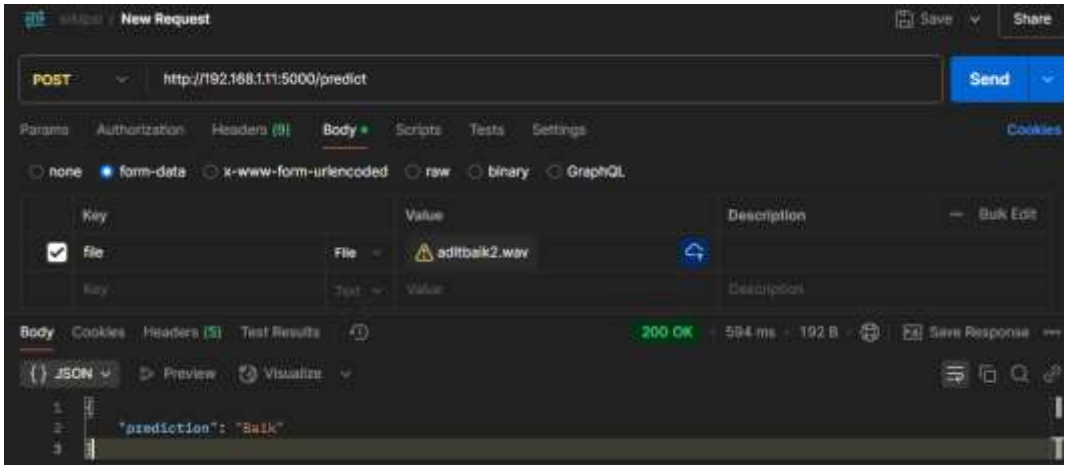
```
except Exception as e:
os.remove(file_path)
print(f"❌ ERROR di server Flask: {e}")  # Tampilkan error di terminal Flask
return jsonify({"error": f"Gagal memproses audio: {e}"}), 500
# Menjalankan API
if __name__ == "__main__":
app.run(host="0.0.0.0", port=5000, debug=True)
```

Dalam kode Program 4. 8 diatas adalah kode API yang dibuat dengan bantuan *library python* yaitu *flask,* isi dari kode program tersebut tidak jauh berbeda dengan kode model sebelumnya hanya menambahan fungsi API. Alur dari kode program tersebut masih sama yang pertama mendfiniskan ualng model LVQ, pemanggilan model menggunakna *pickle, preprocessing* audio, ekstrasi fitur suara *MFCC,* dan yang terakhir prediksi audio. Berikut adalah hasil pengujian model API menggunakan *postman*.

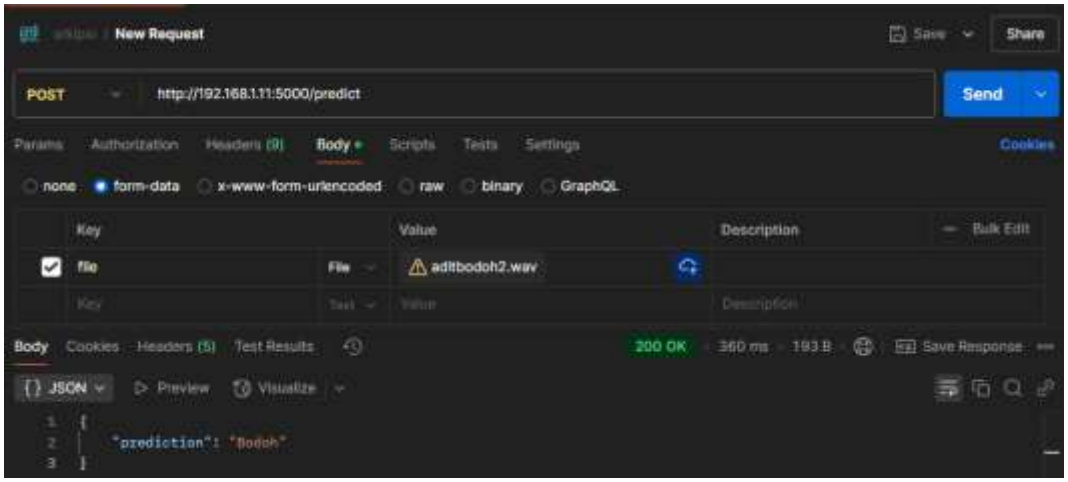
Table 4. 8 Hasil Pengujian API

No	Kata	Hasil Prediksi	Keterangan
1	Baik	Baik	Benar
2	Bodoh	Bodoh	Benar
3	Licik	Licik	Benar
4	Rajin	Rajin	Benar
5	Sombong	Sombong	Sombong

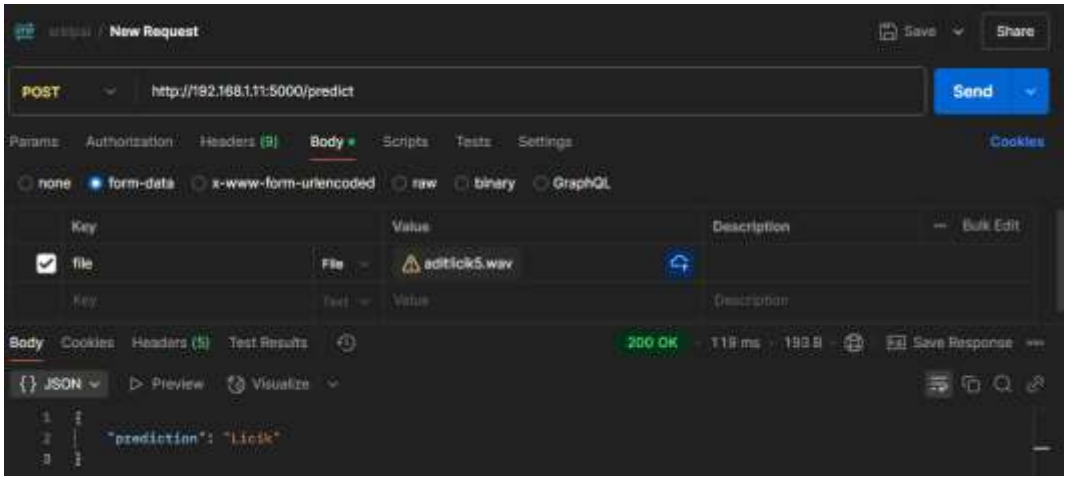
Dari table 4.7 hasil pengujian tersebut menggunakan data baru yang digunakan pengujian sebelumnya, dari hasil tersebut bawha model bisa memprediski kata dengan benar meskipun sudah menjadi sebuah API. Berikt ada hasil output yang ada di postman.



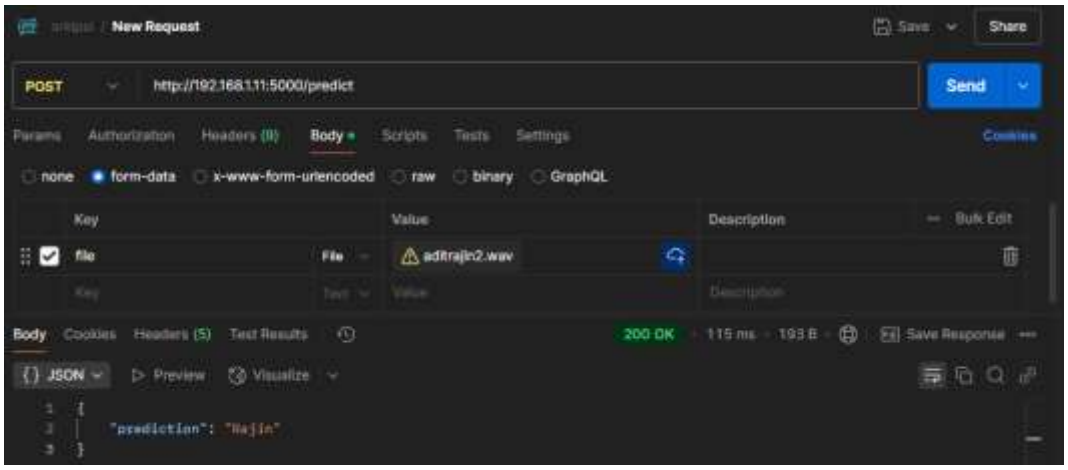
Gambar 4. 16 Output Hasil Postman Kata Baik



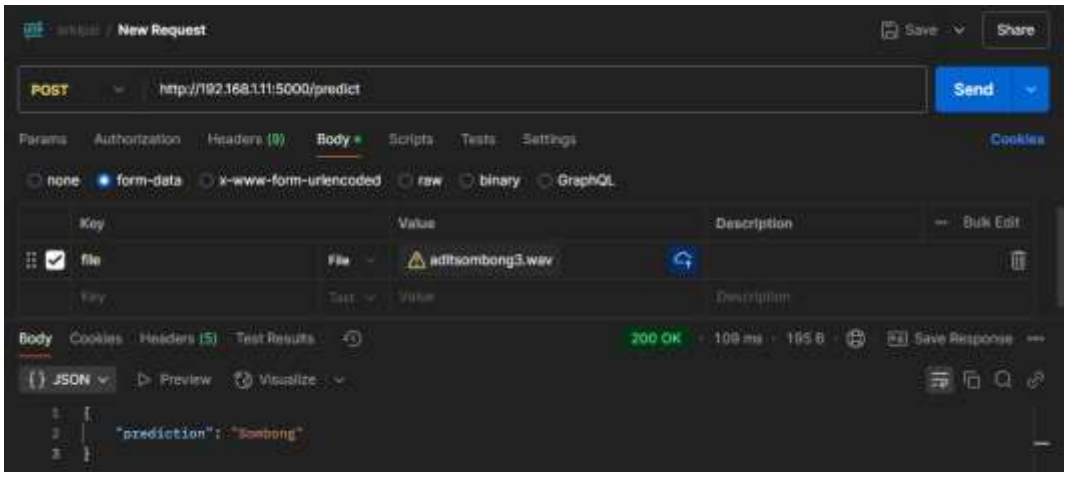
Gambar 4. 17 Output Hasil Postman kata Bodoh



Gambar 4. 18 Output Hasil Postman Kata Licik



Gambar 4. 19 Output Hasil Postman Kata Rajin



Gambar 4. 20 Output Hasil Postman Kata Sombong

4.6 Pembahasan

4.6.1 Dataset

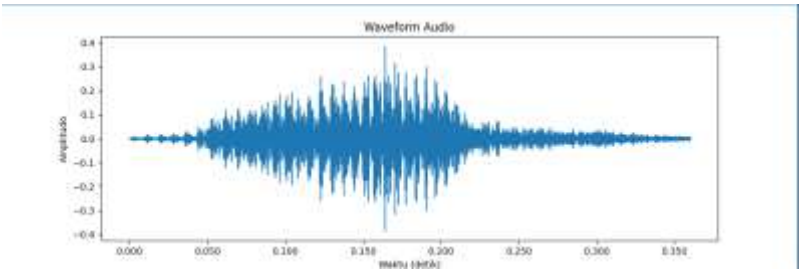
Berdarkan penelitian ini, data yang digunakan berjumlah 500 data rekaman terdiri dari 100 data rekaman kata “baik”, 100 data rekaman kata “bodoh”, 100 data rekaman kata “licik”, 100 data rekaman kata “rajin”, 100 data rekaman kata “sombong”.

4.6.2 *Preprocessing*

Sebelum memasuki tahap *preprocessing* data rekaman yang berhasil dikumpulkan akan melalui proses *noise reduction* yang berfungsi sebagai mengurangi *noise* pada data rekaman, *padding* dilakukan karena data remakan rata – rata mempunyai durasi 1 detik, dengan melakukan *padding* akan menyamakan semua data rekaman di durasi 1 detik¸ dan normalisasi amplitudo dilakukan karena data rekaman atau dataset mempunyai kekuatan suara yang berbeda beda, dengan menggunakan normaliasi kekuatan suara dalam dataset akan disamaratakan. Yang dimana output dari hasil *preprocessing* akan meningkatkan kualitas audio yang akan di ekstrak fiturnya nanti.

4.6.3 Ekstark Fitur Suara MFCC

Tahap selanjutnya adalah mengekstrak fitur suara dari data rekaman menggunakan *MFCC (Mel Frequency Cepstral Coefficient.* Perhitungan berikut akan menggunakan audio sample yang di ambil dari dataset.




```
Sample rate: 48000
Jumlah data: 17280
Contoh data pertama: [0.00253296 0.00238037 0.00265503 0.00256348 0.00262451 0.00259399
 0.00247192 0.00241089 0.00265503 0.00271606]
```

Gambar 4. 21 Sample Data Audio Dari Dataset

Diketahui dari gambar 4.12 diatas sample rate yang digunakan adalah 48000 dengan jumlah data 17280. Berikut adalah parameter *MFCC* yang digunakan.

```
# Parameter MFCC
SAMPLE_RATE = 48000
N_MFCC = 40  # Jumlah koefisien MFCC yang diambil
N_FFT = 4096  # Panjang FFT
HOP_LENGTH = 512  # Hop size
```

Gambar 4. 22 Parameter MFCC

Dalam penelitina ini *MFCC* dibagi menjadi 7 tahapan yaitu *Dc Removal, Frame Blokcing, Windowing, Fast Fourier Transform, Mel Frequency Warping, Discrete consine Transform,* dan *Cepstral Liftering*.

a. DC Removal

Dc Removal diperlukan agar menangani gangguan dalam sinyal suara, gangguan terjadi ketika sinyal suara tidak simetris terhadap garis nol, ada bagian dari sinyal suara yang tergeser ke atas atau kebawah. Untuk menghitung proses *dc removal* maka dilakukan dengan menggunakan persaaman berikut:

$$y[n] = x[n] - \frac{1}{N}\sum_{n=0}^{N-1} x[n]$$

Proses perhitungannya sebagai berikut:

Potongan file audio,

$x = [0.00253296, 0.00238037,\ 0.00265503,\ 0.00256348,\ 0.00262451, \ldots]$

Hitung Rata – Rata,

$$Mean = \frac{1}{N}\sum_{n=0}^{N-1} x[n]$$

$$Mean = \frac{0.00253296 + 0.00238037 + 0.00265503 + \ldots}{17280} = \frac{2.320465}{17280}$$

$$= 0.000134286$$

Hitung sinyal hasil *DC Removal*:

$$y[n] = x[n] - Mean$$

$$y[0] = 0.00253296 - 0.000134286 = 0.002398673$$

$y[1] = 0.00238037 - 0.000134286 = 0.002246085$
$y[2] = 0.00265503 - 0.000134286 = 0.002520743$
$y[17280] = 0.002838135 - 0.000134286 = 0.002703849$

b. *Frame Blocking*

Frame Blocking adalah proses membagi sinyal audio yang panjnag menjadi potongan – potongan kecil (frame) untuk di proses lebih lanjut. Berikut adalah perhitungan untuk pembagian *frame*.

$$Jumlah\ frame = \frac{Total\ Sample - Frame\ Size}{Hop\ Length} + 1$$

$$Jumlah\ frame = \frac{17280 - 4096}{512} + 1$$

$$= \frac{13184}{512} + 1$$

$$= [25.75] + 1 = 26 \text{ frame}$$

Berikut adalah hasil pembagian *frame*:

Table 4. 9 Hasil Pembagian Frame

Frame	Sampel awal	Sampel Akhir	Data
1	0	4095	y[0] sampai y[4095]
2	512	4607	y[512] sampai y[4607]
3	1024	5120	y[1024] sampai y[5120]
4	1536	5632	y[1536] sampai y[5632]
5	2048	6144	y[2048] sampai y[6144]
…	…	…	…
26	12800	16895	y[12800] sampai y[16895]

Pada Table 4. 8 *frame* yang dihasilan berjumlah 26, pada *frame* sisa 16895 – 1780 tidak cukup untuk dibuat 1 *frame* penuh jadi tambahkankan *padding* 0 agar tepat dengan *frame size* yaitu 4096.

c. *Windowing*

Proses *windowing* dalam *MFCC* dilakukan setalah melakukan proses *frame blocking* setalah sinyal audio dibagi menjadi *frame*, setiap frame perlu melalui proses ini sebelum memasuki proses *FFT*. Berikut adalah perhitungan untuk *windowing* 1 *frame*:

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right)$$

Setiap sampel frame di kali dengan hasil windowing:

$$y_{window}(n) = y(n) \text{ x } w(n)$$

Menghitung dari sampel 0 sampai 4096 (1 frame):
Sampel ke-0:

$$w(0) = 0.54 - 0.46\,x\,cos\left(\frac{2\pi\ x\ 0}{4095}\right) = 0.54 - 0.46\,x\,1 = 0.08$$

$y(0) = 0.002398673$:

$$y_{window}(0) = 0.002398673 \text{ x } 0.08 = 0.000191894$$

Sampel ke -1:

$$w(1) = 0.54 - 0.46\,x\,cos\left(\frac{2\pi\ x\ 1}{4095}\right) = 0.54 - 0.46\,x\,1$$
$$= 0.080000541$$

$y(1) = 0.002246085$:

$$y_{window}(1) = 0.002246085 \text{ x } 0.080000541 = 0.000179688$$

Sampel ke- 4095:

$$w(4095) = 0.54 - 0.46\,x\,cos\left(\frac{2\pi\ x\ 4095}{4095}\right) = 0.54 - 0.46\,x\,1$$
$$= 0.80000541$$
$$y(4095) = 0.067095938$$
$$y_{window}(4095) = 0.067095938\,x\,0.80000541 = 0.005367711$$

d. *Fast Fourier Transform*

Mengubah sinyal suara dari domain waktu (amplitudo dan waktu) menjadi domain frekuensi (kekuatan dan frekuensi). Berikut adalah rumus perhitungan untuk *FFT:*

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot \left(\cos\left(\frac{2\pi kn}{N}\right) - j\sin\left(\frac{2\pi kn}{N}\right)\right)$$

Ambil contoh 8 sampel awal:

$$X(0) = 0.000191894\ x\ \cos\left(\frac{2\pi\ x\ 0\ x\ 0}{8}\right) - j\ x\ sin\ \cos\left(\frac{2\pi\ x\ 0\ x0}{8}\right)$$
$$+ 0.000179688\ x\ \cos\left(\frac{2\pi\ x\ 0\ x\ 1}{8}\right) - j\ x\ sin\ \cos\left(\frac{2\pi\ x\ 0\ x\ 1}{8}\right) + \ldots$$
$$\ldots + 0.000182189\ x\ \cos\left(\frac{2\pi\ x\ 0\ x\ 7}{8}\right) - j\ x\ sin\ \cos\left(\frac{2\pi\ x\ 0\ x\ 7}{8}\right)$$
$$= 0.001533915 + 0j = 0.001533915$$

e. *Mel Frequency Warping*

Mel Frequency Warping adalah proses mengubah skala frekuensi linear (*hz*) menjadi skala *mel.* Berikut ada rumus perhitungan yang digunakan untuk proses *mel frequency warping:*

$$f_{mel} = \frac{2595 \log_{10}\left(1 + \frac{f}{700}\right)}{\frac{Si}{2}}$$

Ambil contoh sampel 8 awal hasil *FFT:*

$$H_0 = \frac{2595\ x\ Log_{10}\ (1\ +\ \frac{1000}{700})}{-0.16/2}$$
$$H_0 = \frac{2595\ x\ Log_{10}\ (1\ +\ 1.4286)}{-0.08}$$
$$H_0 = \frac{2595\ x\ Log_{10}(2.4286)}{-0.08}$$
$$H_0 = \frac{2595\ x\ 0.3858}{-0.08}$$
$$H_0 = \frac{1001.73}{-0.08} = -12499.82$$

Hitung nilai y0:

$Y_0 = 0.001533915\ x\ (-12499.82)$

$Y_0 = 19.1616$

f. *Discrete Consine Transform*

Digunakan untuk mengkompresi informasi frekuensi dan menghasilkan fitur yang lebih ringkas dan representative yaitu *MFCC.* Berikut adalah rumus perhitungan yang di pakai:

$$C_n = \sum_{k=1}^{K} (\log S_k) \cos\left[n\left(k - \frac{1}{2}\right)\frac{\pi}{K}\right]$$

Untuk perhitungan *DCT* pada nilai hasil *filterbank* = 19.1616, dengan nilai K (koefisien) = 40, untuk n = 0 maka,

$$c0 = log\ (19.1616)\ cos\ [0(0 - 1/2)3.14/40]$$
$$+ log\ (19.1616)\ cos\ [0(1 - 1/2)3.14/40] + ..$$
$$... + log\ (19.1616)\ cos\ [0(7 - 1/2)3.14/40 = 10.26$$

g. *Cepstral Liftering*

Proses ini dilakukan untun meningkatkan kualitas fitur *MFCC* dengan memperjelas informasi penting dan mengurangi pengaruh noise di koefisien tinggi. Berikut adalah rumus perhitungan untuk menghitung *Cepstral liftering:*

$$W[n] = \left(1 + \frac{L}{2}\sin\left(\frac{n\pi}{L}\right)\right) \times C_n$$

Hasil perhitungan untul nilai DCT = 10.26 maka, untuk *cepstral liftering* pada *W0* adalah sebagai berikut:

$$W0 = 10.26\ x\ 22/2\ x\ sin(3.14/22) = 16.06$$

Lakukan cara yang sama ke data suara yang lain yang nantinya didapatkan suatu vektor ciri yaitu koefisien *MFCC.* Setelah mendapatkan nilai koefisien *MFCC* niai tersebut akan dirata – ratakan menjadi vektor 1 dimensi untuk setiap kata, tujuannya agar mempermudah model *LVQ* menerima inputan *MFCC.*

4.6.4 Model *Learning Vector Quantization*

Tahapan selanjutnya adalah tahap pelatihan dan pengujian menggunakan metode *learning vector quantization.* Parameter yang digunakan pada layer input

adalah vektor rata – rata hasil ekstrasi fitur suara menggunakan *MFCC*. Pada proses latihan, data yang digunakan sebagai data latih sebanyak 350 data dan data uji sebanyak 150 data (dengan rasio pembagian dataset 7:3). Hasil klasifikasi menunjukan akurasi tertinggi sebesar 91.45% dengan parameter *LVQ* menggunkana *learning rate* 0.01 dan jumlah epoch sebanyak 200. Klasifikasi dilakukan untuk 5 kata yaitu, baik, bodoh, licik, rajin, dan sombong.

Langkah berikutnya tahap pengujian model *LVQ* yang sudah dilatih. Model yang sudah di latih akan di simpan menggunakan bantuan dari *libray python* yaitu *pickle*, setelah disimpan model akan bisa di panggil lagi tanpa melakukan pelatihan ulang lagi. Data yang di uji merupakan data yang berbeda dengan yang ada di dataset rekaman suara, dari 10 kata yang di uji model menunjukan tingkat *confidence score* paling tinggi 96% pada kata sombong untuk data suara laki – laki dan 80% sampai 87% pada kata baik, dan tingkat *confidencde score* paling rendah yang bisa didapat oleh model adalah 47% pada kata licik. Meskipun model menghasilkan tingkat *confidence socre* rendah dibawah 50% model masih tetap bisa mengindentifikasi kata yang benar.

Dalam metode *Learning Vector Quantization*, proses klasifikasi dilakukan dengan membandingkan vektorr fitur dari data masukan (*input*) dengan vektor bobot (*prototype*) yang telah di inisialisasi sebelumnya. Berikut adalah tahapan perhitungan manualnya:

a. Inisialisasi Bobot Awal

Dalam penelitian ini menggunakan 2 prottotipe yang berarti di inisialisasikan 2 prototipe secara acak untuk setiap kelas:

Table 4. 10 Inisialisasi Bobot Awal Model

Kelas	Prot.	MFCC1	MFCC2	MFCC3	…	MFCC40
Baik	P1	-863.637	119.620	43.969	…	-25.817
Baik	P2	-1106.769	91.638	24.451	…	-17.583
Bodoh	P1	-851.845	174.421	68.524	…	-27.917
Bodoh	P2	-821.781	189.326	51.827	…	-27.069
Licik	P1	-989.756	77.213	35.721	…	-7.806

Kelas	Prot.	MFCC1	MFCC2	MFCC3	…	MFCC40
Licik	P2	-1037.119	88.649	24.000	…	-5.712
Rajin	P1	-912.062	136.052	42.055	…	-10.083
Rajin	P2	-855.273	181.106	118.262	…	-7. 638
Sombong	P1	-852.926	339.372	142.108	…	-33.692
Sombong	P2	-838.327	163.804	84.593	…	-15.353

b. Perhitungan Jarak Euclidean Untuk menentukan Pemenang

Ambil contoh bebarapa data fitur dari data latih sebagai contoh:

$$X = (-1106.769, 91.683, 24.451)$$

Jarak *Euclidean* dihitung dengan menggunakan rumus:

$$||X - W_i|| = \sqrt{\sum_{j=1}^{m} (X_j - W_{ij})^2}$$

Contoh perhitungan untuk kelas "Baik" prototipe 1:

$$d = \sqrt{(-1106.769 + 863.637)^2 + (91.683 - 119.620)^2 + (24.451 - 43.969)^2}$$

$$d = \sqrt{(-243.132)^2 + (-27.937)^2 + (-19.518)^2}$$

$$d = \sqrt{59116.96 + 780.47 + 381.00}$$

$$d = \sqrt{60278.43}$$

$$d = 245.53$$

Ulangi perhitungan ini untuk semua prototipe dan pilih prototipe dengan jarak terkecil. Misal, hasil perhitungan menunjukan bahwa P2 kelas "Baik" memiliki jarak terkecil, maka P2 dipilih sebagai pemenang.

c. Pembaruan Bobot LVQ

Jika kelas prototipe yang dipilih sama dengan kelas sebenarnya, bobot diperbarui menggunakan:

$$W_i(t+1) = W_i(t) + a \cdot (X - W_i(t))$$

Jika kelas prototipe yang dipilih berbeda dari kelas sebenarnya, bobot diperbarui dengan:

$$W_i(t+1) = W_i(t) - a \cdot (X - W_i(t))$$

Misal kelas sebenarnya adalah "Baik", dan *learning rate* yang digunakan adalah 0.01, maka pembaruan bobot untuk prototipe P2 kelas "Baik" dilakukan sebagai berikut:

$$w_1 = -863.637 + 0.01(-1106.769 - (-863.637))$$
$$w_1 = -863.637 + 0.01(-243.132)$$
$$w_1 = -863.637 - 2.431$$
$$w_1 = -866.068$$

$$w_2 = 119.620 + 0.01(91.683 - 119.620)$$
$$w_2 = 119.620 + 0.01(-27.937)$$
$$w_2 = 119.620 - 0.279$$
$$w_2 = 119.341$$

$$w_3 = 43.969 + 0.01(24.451 - 43.969)$$
$$w_3 = 43.969 + 0.01(-19.518)$$
$$w_3 = 43.969 - 0.195$$
$$w_3 = 43.774$$

d. Hasil Bobot Setelah Pembaruan

Table 4. 11 Table Update Bobot Prototipe

Kelas	Prot.	MFCC1	MFCC2	MFCC3
Baik	P2	-866.068	119.341	43.341

Proses ini diulang untuk setiap data latih hingga mencapai jumlah iterasi atau *epoch* tercapai dalam penelitian ini jumlah *epoch* adalah 200. Hasil akhir dari pelatihan ini adalah bobot prototipe yang sudah dilatih, yang nanti akan dibandingkan dengan data baru (bukan data dari dataset) untuk memprediksi kata data baru.

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan mengenai pemanfataan metode *Mel Frequency Cepstral Coefficient* (MFCC) dan *Learning Vector Quantization* (LVQ) dalam pengenalan ucapan bahasa Indonesia, dapat ditarik beberapa kesimpulan sebagai berikut:

1. Kombinasi metode MFCC sebagai ekstrasi fitur suara dalan LVQ sebagai metode klasifikasi telah menunjukan hasil yang cukup baik dalam mengenali ucapan bahasa Indonesia. Proses ekstrasi MFCC berhasil menghasilkan vektor fitur yang representatif terhadap karakteristik suara.
2. Hasil pelatihan model menunjukan bahwa kombinasi parameter *learning rate* sebesar 0.01 dan jumlah *epoch* sebanyak 200 dengan pembagian data latih dan uji 70:30 memberikan akurasi tertinggi sebesar 91.45%. Hal ini menunjukan bahwa metode LVQ dapat mengklasifikasikan data ucapan dengan baik.
3. Model memiliki peforma yang baik dalam mengenali kata “Bodoh” dan “Sombong” dengan nilai *f1-score* tertinggi, masing-masing 0.97 dan 0.96. namun pada kata “Rajin” menunjukan nilai *recall* yang rendah yaitu 0.76, mengindikasikan bahwa beberapa pengucapan kata tersebut diklasifikasikan sebagai kata yang lain atau kelas yang lain.
4. Evaluasi juga dilakukan dengan menggunakan *K-Fold Cross Validation* (k=5) menunjukan bahwa model memiliki rata-rata akurasi sebesar 82.53%, dengan akurasi tertinggi 88.12% dan terendah 78.00%. Ini menunjukan bahwa model memiliki tingkat generalisasi yang cukup baik terhadap data baru.
5. Model yang sudah dilatih dapat diimplementasikan dalam bentuk API menggunakan *flask,* memungkinkan penggunaan model secara langsung tanpa perlu melakukan pelatihan ulang. Pengujian API menunjukan bahwa model dapat mengenali kata dengan benar meskipun sudah diubah kedalam format API.

6. Meskipun model memiliki akurasi tinggi saat pelatihan, terdapat beberapa kendala, seperti saat pengujian data baru yang dimana tingkat *confidence score* dibeberapa kata data uji rendah pada kata "Licik" memiliki tingak *confidence score* 47%, yang mengindikasikan bahwa beberapa kata sulit dibedakan akibat kesamaan fitur suara. Selain itu, pengaruh kualitas rekaman dan kurang nya variasi data dalam dataset juga dapat mempengaruhi hasil klasifikasi.

5.2 Saran

Berdasarkan hasil dari penelitian yang telah dilakukan, terdapat beberapa saran yang dapat dijadikan pertimbangan untuk pengembangan lebih lanjut dalam penelitian pengenalan ucapan bahasa Indonesia menggunakan metode MFCC dan LVQ, yaitu sebagai berikut:

1. Penelitian ini menggunakan dataset yang terdiri dari lima sifat dengan jumlah sampel yang terbatas. Untuk meningkatkan kinerja model, disarankan untuk menggunakan dataset yang lebih besar dengan variasi kata yang lebih banyak.
2. Metode LVQ telah menunjukkan performa yang baik, tetapi masih memiliki keterbatasan dalam mengenali beberapa kata dengan akurasi yang lebih rendah. Oleh karena itu, disarankan untuk membandingkan metode ini dengan algoritma lain seperti *Support Vector Machine* (SVM), *Random Forest*, atau model berbasis Deep Learning seperti *Convolutional Neural Network* (CNN) atau *Recurrent Neural Network* (RNN).
3. Penelitian ini hanya berfokus pada pengenalan kata dalam kondisi rekaman yang terkendali. Untuk meningkatkan akurasi model, disarankan melakukan uji coba pada data suara dengan kondisi lingkungan yang lebih kompleks, seperti latar belakang bising atau variasi kecepatan pengucapan.
4. Model yang dikembangkan telah diuji dalam lingkungan penelitian. Untuk implementasi lebih lanjut, model dapat dikembangkan dalam bentuk aplikasi berbasis mobile atau web, sehingga pengguna dapat melakukan pengujian langsung terhadap sistem pengenalan ucapan.

DAFTAR PUSTAKA

Abdillah Alwi, A., & Pandu Adikara, P. (2020). *Pengenalan Jenis Kelamin dan Rentang Umur berdasarkan Suara menggunakan Metode Backpropagation Neural Network* (Vol. 4, Issue 7). http://j-ptiik.ub.ac.id

Aren, G., Dengan, A., Melisa, C., Manaor, A., Pardede, H., & Sihombing, M. (2022). Implementasi Learning Vector Quantization (LVQ) Dalam Mengidentifikasi. In *Sci-Tech Journal* (Vol. 1, Issue 1).

Dyarbirru, Z., & Hidayat, S. (2020). *Metode Wavelet-MFCC dan Korelasi dalam Pengenalan Suara Digit (Wavelet-MFCC Method and Correlation in Digit Voice Recognition)*. *2*(2), 100–108. https://github.com/Jakobovski/free-spoken-digit-

Fitria, L., Muttaqin, K., & Nasution, M. S. (2021). *Implementasi Speech Recognition pada Kata Kerja Dasar Menggunakan Metode MFCC*. *02*(01), 43–50. https://ejurnalunsam.id/index.php/jicom/

Kristina, L. S., Fadila Fitriana, G., & Prasetiadi, A. (2020). Pemisahan Suara Manusia Berdasarkan Jenis Kelamin Menggunakan Fast Fourier Transform (FFT). *Jurnal Teknik Informatika Dan Sistem Informasi*, *7*(3). http://jurnal.mdp.ac.idTelp

Mustikarini, W., Hidayat, R., & Bejo, A. (2019). Real-Time Indonesian Language Speech Recognition with MFCC Algorithms and Python-Based SVM. In *IJITEE* (Vol. 3, Issue 2).

Negeri, P., & Rahmadani, L. (2020). Klasifikasi Gender Berdasarkan Suara Dengan Naive Bayes Dan Mel Frequency Cepstral Coefficient. *Technology Journal*, *2*(1), 19–26. https://doi.org/10.15575/jw.xxx.xxx

Rekayasa, K. K., Dharma Adhinata, F., Putra Rakhmadani, D., Jala, A., & Segara, T. (2021). Terbit online pada laman web jurnal: http://journal.ittelkom-pwt.ac.id/index.php/dinda JURNAL DINDA Pengenalan Jenis Kelamin Manusia Berbasis Suara Menggunakan MFCC dan GMM. In *Data Institut Teknologi Telkom Purwokerto* (Vol. 1, Issue 1). https://research.google.com/audioset.

Sasilo, A. A., Saputra, R. A., & Ningrum, I. P. (2022). Sistem Pengenalan Suara Dengan Metode Mel Frequency Cepstral Coefficients Dan Gaussian Mixture Model. *Komputika : Jurnal Sistem Komputer*, *11*(2), 203–210. https://doi.org/10.34010/komputika.v11i2.6655

Seminar, M., Akhir, T., Setiawan, A., Hidayatno, A., & Rizal Isnanto, R. (2012). *MEL-FREQUENCY CEPSTRUM COEFFICIENTS (MFCC) MELALUI JARINGAN SYARAF TIRUAN (JST) LEARNING VECTOR QUANTIZATION (LVQ) UNTUK MENGOPERASIKAN KURSOR KOMPUTER*.

Sidik Permana, I., Nurhasanah, Y. I., & Zulkarnain, A. (2018). IMPLEMENTASI METODE MFCC DAN DTW UNTUK PENGENALAN JENIS SUARA PRIA DAN WANITA. *MIND Journal | ISSN*, *3*(1), 49–63. https://doi.org/10.26760/mindjournal

Tawakal, F., Azkiya, A., Teknik Informatika, J., Informatika, M., & Manajemen Informatika dan Komputer Dumai Jl Utama Karya Bukit Batrem Dumai - Riau, A. (2020). Diagnosa Penyakit Demam Berdarah Dengue (DBD) menggunakan Metode Learning Vector Quantization (LVQ) Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Dumai, (2). In *JANUARI* (Vol. 4, Issue 3).

Tridarma, P., & Endah, S. N. (n.d.). Pengenalan Ucapan Bahasa Indonesia Menggunakan MFCC dan Recurrent Neural Network. In *Jurnal Masyarakat Informatika* (Vol. 11, Issue 2).

Tri Handoko, I. (2019). *Klasifikasi Gender dan Usia berdasarkan Suara Pembicara Menggunakan Hidden Markov Model*. https://doi.org/10.21108/indojc.2019.4.3.375

Wirawan Setialaksana, Dwi Rezky Anandari Sulaiman, Shabrina Syntha Dewi, Chairuunnisa Ar Lamasitudju, & Nini Rahayu Ashadi, M. A. (2020). *Model Jaringan Syaraf Tiruan dalam Peramalan Kasus Positif Covid-19 di Indonesia*. *3*.

LAMPIRAN

Lampiran 1 Dokumentasi Pengambilan Data Suara Laki-Laki

Lampiran 2 Dokumentasi Pengambilan Data Suara Perempuan