PENERAPAN METODE *MEL FREQUENCY CEPSTRUM COEFFICIENT* DAN *DYNAMIC TIME WARPING K NEAREST NEIGHBOUR* DALAM REKOGNISI AKSEN SUKU DI INDONESIA

SKRIPSI

Oleh :
ASRORI ZAINUR RIDHO
NIM. 15650007

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG

2019

PENERAPAN METODE *MEL FREQUENCY CEPSTRUM COEFFICIENT* DAN *DYNAMIC TIME WARPING K NEAREST NEIGHBOUR* DALAM REKOGNISI AKSEN SUKU DI INDONESIA

SKRIPSI

Diajukan kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S. Kom)

Oleh:
ASRORI ZAINUR RIDHO
NIM. 15650007

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2019

LEMBAR PERSETUJUAN

PENERAPAN METODE *MEL FREQUENCY CEPSTRUM COEFFICIENT* DAN *DYNAMIC TIME WARPING K NEAREST NEIGHBOUR* DALAM REKOGNISI AKSEN SUKU DI INDONESIA

SKRIPSI

Oleh:
ASRORI ZAINUR RIDHO
NIM. 15650007

Telah Diperiksa dan Disetujui untuk Diuji
Tanggal : 19 Juni 2019

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Suhartono, M.Kom	**M. Imamudin, Lc., MA**
NIP. 196805192003121001	**NIP. 19740602 200901 1 010**

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

iii

LEMBAR PENGESAHAN

PENERAPAN METODE *MEL FREQUENCY CEPSTRUM COEFFICIENT* DAN *DYNAMIC TIME WARPING K NEAREST NEIGHBOUR* DALAM REKOGNISI AKSEN SUKU DI INDONESIA

SKRIPSI

Oleh:
ASRORI ZAINUR RIDHO
NIM. 15650007

Telah Dipertahankan di Depan Dewan Penguji
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Pada Tanggal 19 Juni 2019

Susunan Dewan Penguji			Tanda Tangan
1. Penguji Utama	:	**Fachrul Kurniawan, M.MT** **NIP. 19771020 200912 1 001**	( )
2. Ketua Penguji	:	**Yunifa Miftachul Arif, M.T** **NIP. 19830616 201101 1 004**	( )
3. Sekretaris Penguji	:	**Dr. Suhartono, M.Kom** **NIP. 196805192003121001**	( )
4. Anggota Penguji	:	**M. Imamudin, Lc., MA** **NIP. 19740602 200901 1 010**	( )

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

iv

HALAMAN PERNYATAAN
ORISINALITAS PENELITIAN

Nama : Asrori Zainur Ridho
NIM : 15650007
Jurusan : Teknik Informatika
Fakultas : Sains dan Teknologi
Judul Skripsi : Penerapan Metode *Mel Frequency Cepstrum Coefficient* Dan *Dynamic Time Warping K Nearest Neighbour* Dalam Rekognisi Aksen Suku Di Indonesia.

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil dari karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 19 Juni 2019
Yang membuat pernyataan,

Asrori Zainur Ridho
NIM. 15650007

v

HALAMAN MOTTO

Pada setiap orang kesempatan hanya datang satu kali, namun kesempatan akan selalu datang untuk orang yang pantang menyerah dan terus mencoba

HALAMAN PERSEMBAHAN

Kupersembahkan karya ini kepada:

Kedua orang tuaku, Bapak Masdiannor dan Ibu Rainah yang telah membesarkanku dengan penuh kasih sayang dan penuh cinta yang tak terhingga. Berkali-kali aku terjatuh, berkali-kali aku ingin menyerah, tapi ingatanku akan kerinduan dan tangung jawab sebagai seorang anak membuatku tuk bangkit dan terus berjuang. Kusadari kebaikan yang kudapatkan tak lepas atas iringan do'amu yang selalu menyertaiku atas izinNya. Semoga Allah memanjangkan umur Bapak dan Ibu dalam kebaikan dunia dan akhirat.

Semua guru-guruku yang telah membimbing dan mengajariku dengan sabar dan ikhlas, semoga menjadi amal jariyah yang tidak terputus. Semua sahabat yang mau menghabiskan waktu bersama-sama. Semua kawan kawanku yang saling ulur tangan dalam menjalani rutinitas harian kita.

Kepada semua pihak yang tidak dapat disebutkan satu per satu, yang telah membantu penulis dalam menyelesaikan karya tulis ini hingga selesai tepat waktu. Semoga Allah SWT mempermudah urusan teman teman semua sehingga keinginannya dapat terpenuhi.

Terima kasih.

KATA PENGANTAR

Assalamu'alaykum wr.wb

Segala puji bagi Allah SWT. yang Maha Pengasih lagi Maha Penyayang atas Rahmat dan Karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penerapan Metode *Mel Frequency Cepstrum Coefficient* dan *Dynamic Time Warping K Nearest Neighbour* Dalam Rekognisi Aksen Suku Di Indonesia" dengan baik dan lancar. Sholawat serta salam semoga selalu tercurahkan kepada junjungan besar, Nabi Muhammad SAW. yang telah membimbing umatnya dari zaman kebodohan menuju Islam yang rahmatan lil alamin. Dalam menyelesaikan skripsi ini penulis menyadari bahwa tidak lepas dari bantuan, bimbingan, serta dukungan dari berbagai pihak. Atas bantuan yang telah diberikan, penulis ingin menyampaikan do'a dan ucapan terima kasih sebesar-besarnya kepada:

1. Prof. Dr. H. Abdul Haris, M. Ag, selaku Rektor UIN Maulana Malik Ibrahim Malang beserta staf.
2. Dr. Sri Harini M, Si, selaku Dekan Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang beserta staf.
3. Bapak Dr. Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika UIN Maulana Malik Ibrahim Malang, yang tak pernah bosan untuk terus memotivasi dan menginspirasi untuk menjadi pribadi yang lebih baik.
4. Bapak Dr. Suhartono, M. Kom, selaku Dosen Wali dan Pembimbing I yang telah bersedia meluangkan waktu untuk membimbing, memotivasi, dan

memberikan arahan kepada penulis sehingga dapat menyelesaikan skripsi ini hingga selesai.

5. Bapak M. Imamudin, Lc., MA., selaku pembimbing II yang juga senantiasa memberikan masukan dan nasihat dalam penyusunan skripsi ini.
6. Ayah, Ibu, Kakak, Adik, beserta keluarga tercinta yang selalu menjadi motivasi penulis dalam beraktivitas dan do'a yang senantiasa diberikan kepada penulis.
7. Seluruh Dosen Teknik Informatika yang telah memberikan keilmuannya kepada penulis.
8. Seluruh sahabat yang selalu membuat ceria, dan melengkapi kekurangan dalam beraktivitas.
9. Teman-teman seperjuangan Teknik Informatika 2015 "Interface" yang telah saling memotivasi, dan saling membantu dalam solidaritas.

Harapan penulis, semoga amal kebaikan dan jasa-jasa dari semua pihak yang telah membantu hingga skripsi ini dapat selesai dengan baik dan lancar diterima oleh Allah SWT, dan mendapatkan balasan yang berlipat ganda. Penulis menyadari bahwa skripsi ini jauh dari kesempurnaan karena terbatasnya pengalaman dan kemampuan penulis dalam menyelesaikan skripsi ini. Oleh karena itu, penulis mengharapkan segala bentu saran dan kritik yang membangun dari berbagai pihak. Semoga skripsi ini dapat bermanfaat dan mendorong peneliti berikutnya untuk melanjutkan dan menyempurnakan penelitian ini.

Wassalamu'alaykum Wr. Wb.

Malang, 17 Juni 2019

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Ridho, Asrori Zainur. 2019. ***Penerapan Metode Mel Frequency Cepstrum Coefficient dan Dynamic Time Warping K Nearest Neighbour Dalam Rekognisi Aksen Suku Di Indonesia***. Skripsi. Jurusan Teknik Informatika. Fakultas Sains dan Teknologi. Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. Suhartono, M. Kom, (II) M. Imamudin, Lc., MA

Kata Kunci: Aksen Suku, Rekognisi, *Mel Frequency Cepstrum Coefficient*, *Dynamic Time Warping*, *K Nearest Neighbour*.

Indonesia memiliki banyak suku, setiap suku memiliki banyak identitas, diantara identitas tersebut adalah aksen berbicara. Seseorang dapat diketahui asal sukunya hanya dengan mendengarkan bagaiamana orang tersebut berbicara dalam aksen tuturnya. Aksen bicara dalam setiap suku berbeda-beda, namun memiliki pola yang dapat diketahui sehingga dapat dikenali. Penerapan rekognisi suara belakangan ini digunakan untuk kemudahan akses, pengamanan, hingga uji forensik. Oleh sebab itu dibutuhkan suatu metode yang dapat mengukur suatu suara aksen suku dan dapat melakukan rekognisi terhadap suara aksen tersebut. Pada penelitian ini dibangun suatu sistem untuk merekognisi suara aksen suku menggunakan metode *Mel Frequency Cepstrum Coefficient* (MFCC) dan *Dynamic Time Warping K Nearest Neighbour* (DTW-KNN). Metode MFCC digunakan untuk mengekstraksi sinyal suara agar sinyal dapat berbentuk lebih sederhana, sedangkan metode DTW-KNN merupakan modifikasi dari metode DTW dan KNN yang digunakan untuk merekognisi hasil dari sinyal suara yang diuji. Pada hasil pengujian telah di dapatkan hasil bahwa dari pengujian data uji terhadap data koleksi menggunakan metode DTW-KNN mendapatkan nilai akurasi sebesar 88%, sedangkan pada metode KNN dan DTW mendapatkan hasil akurasi sebesar 50% dan 75%.

ABSTRACT

Ridho, Asrori Zainur. 2019. ***Application of Mel Frequency Cepstrum Coefficient And Dyynamic Time Warping K Nearest Neighbour to Recognition of Tribal Accents of Indonesia.*** Undergradute Thesis. Informatics Engineering Department. Science and Technology Faculty. Islamic State University Maulana Malik Ibrahim Malang. Supervisors: (I) Dr. Suhartono, M. Kom, (II) M. Imamudin, Lc., MA

Keywords : Tribal Accents, Recognition, *Mel Frequency Cepstrum Coefficient*, *Dynamic Time Warping*, *K Nearest Neighbour*.

Indonesia has many tribes, each tribe has many identities, among these identities is a speech accent. A person can be known to his tribe just by listening to how the person speaks in his speech accent. Accents speak in each tribe differently, but have patterns that can be identified. The application of speech recognition has recently been used for easy access, security and for forensic testing. Therefore, we need a method that can measure and recognize a tribal accent voice. In this study, a system is created to recognize tribal accent voice using the *Mel Frequency Cepstrum Coefficient* (MFCC) and *Dynamic Time Warping K Nearest Neighbor* (DTW-KNN) methods. The MFCC method is used to extract voice signals so that the signal can be simpler, while the DTW-KNN method is a modification of the DTW and KNN methods used to recognize the results of the voice signals being tested. The results of the test have obtained results that from testing the test data to the collection data using the DTW-KNN method, the accuracy value is 88%, while the KNN and DTW methods get an accuracy of 50% and 75%.

المستخلص

الرضى ,أسراري زينو. ٢٠١٩.التطبيق بطريقة *Mel Frequency Cepstrum Coefficient* و *Dynamic Time Warping K Nearest Neighbour* للاعتراف اللحجة في اندونسية. البحث الجامعي. قسم هندسة المعلومايتة. كلية العلوم التكنولوجا. جامعة مولانا مالك إبراهيم الإسلا مية الحكومية مالانج. المشرف الأول الدكتور سوهارتونو, الماجيستير, المشرف الثانى محمد إمام الدين, الماجيستير.

الكالمات الرئيسية : لهجة القبلية, اعتراف

يوجد في إندونيسيا العديد من القبائل ، ولكل قبيلة هويات عديدة ، ومن بين هذه الهويات لهجة الحديث. يمكن أن يكون الشخص معروفًا بأصله ببساطة من خلال الاستماع إلى الطريقة التي يتحدث بها الشخص في لهجته الخطابية. لهجات الحديث في كل قبيلة مختلفة، ولكن لكل لهجات نمط يمكن معرفته بحيث يمكن تحديده. تم استخدام تطبيق التعرف على الصوت مؤخرًا لسهولة الوصول والأمان باختبارها. لذلك نحتاج إلى طريقة يمكنها قياس صوت اللهجة القبلية ويمكنها التعرف على صوت اللهجة. يستخدم في هذه البحث بناء نظام للتعرف على لهجات الصوت باستخدام طريقة Mel *Frequency Cepstrum Coefficient* (MFCC) *& DynamicTime Warping K Nearest Neighbour* (DTW-KNN). طريقة MFCC تستخدم لاستخراج الإشارات الصوتية بحيث تكون الإشارات أكثر بساطة ، بينما الطريقة DTW-KNN هو تعديل الأسلوب DTW-KNN الذي يستخدم للتعرف على نتائج الإشارات الصوتية التي يجري اختبارها. حصلت نتائج الاختبار على نتائج من اختبار بيانات الاختبار على بيانات التجميع باستخدام الطريقة DTW-KNN الحصول على قيمة دقة 8% 8, بينما في هذه الطريقة DTW و KNN الحصول على دقة 50% و 75%.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Menurut sensus Badan Pusat Statistik pada tahun 2010, ada lebih dari 300 kelompok etnik atau suku bangsa di Indonesia, atau tepatnya 1.340 suku bangsa. Setiap suku tentunya memiliki karakteristik budayanya masing masing, satu diantaranya adalah bahasa daerah. Bahasa daerah memiliki ciri khasnya masing masing, dari sastra penggunaan bahasa, hingga halus atau lembutnya tata bahasa yang digunakan akan mempengaruhi pola (*pattern*) dari intonasi pembicaraan, hal ini disebut dengan aksen. Menurut KBBI, aksen adalah tekanan suara pada suku kata; pelafalan khas yang menjadi ciri khas seseorang. Walaupun orang Indonesia umumnya menggunakan bahasa Indonesia dalam berbicara sehari-hari, namun orang tersebut tentunya masih membawa pelafalan kata yang sesuai dengan bahasa daerah yang digunakan. Intonasi aksen yang didengar dapat membuat pendengarnya mengetahui bahwa seseorang tersebut berasal dari suku daerah asalnya. Hal ini sesuai dengan firman Allah dalam surah Al-Hujurat ayat 13:

يَا أَيُّهَا النَّاسُ إِنَّا خَلَقْنَاكُمْ مِنْ ذَكَرٍ وَأُنْثَى وَجَعَلْنَاكُمْ شُعُوباً وَقَبَائِلَ لِتَعَارَفُوا إِنَّ أَكْرَمَكُمْ عِنْدَ اللَّهِ أَتْقَاكُمْ إِنَّ اللَّهَ عَلِيمٌ خَبِيرٌ

Artinya:
"Hai manusia, sesungguhnya Kami menciptakan kamu dari seorang laki-laki dan seorang perempuan dan menjadikan kamu berbangsa-bangsa dan bersuku-suku supaya kamu saling mengenal. Sesungguhnya orang yang paling mulia di antara kamu di sisi Allah ialah orang yang paling bertakwa di antara kamu. Sesungguhnya Allah Maha Mengetahui lagi Maha Mengenal".
(Qs. al-Hujurat: 13)

Surah tersebut menjelaskan bahwa Allah telah menciptakan seorang laki-laki (Adam) dan seorang perempuan (Hawa) dan kemudian menjadikannya berbangsa-bangsa, bersuku-suku agar saling mengenal. Dalam mengenal tentunya ada perbedaan untuk membedakan yang satu dengan yang lainnya. Perbedaan tersebut muncul dari penampilan, kebiasaan, lingkungan, dan sebagainya, dari hal itulah perbedaan tersebut menjadi tolak ukur sebagai pembeda antara yang satu dengan yang lain. Namun bagaimanapun penampilan saat hidup di dunia ini, walaupun tercipta dengan penampilan berbeda-beda, di mata Allah Swt. semua itu sama, tolak ukur yang membedakan hanyalah iman dan taqwa.

Berkembangnya teknologi dalam bidang pengenalan suara (*speech recognition*) banyak dimanfaatkan dalam bidang kenyamanan pengguna dan studi forensik. Contohnya seperti untuk membuka kunci ponsel pintar (*smartphone*) ataupun melakukan tes forensik dalam kasus hukum yang terjadi. Hal ini dapat dilakukan karena suara/sinyal yang masuk (*input*) memiliki variasi atau pola yang dapat diukur. Sehingga komputer sebagai mesin dapat mengenali sinyal yang masuk tersebut sebagai angka-angka yang dapat diproses. Pemrosesan suara normalnya menggunakn 44.100 Hz *(sample rate*) dalam 1 detik untuk kecepatan pemutaran pada DVD, input tersebut merupakan kategori input dengan nilai yang sangat banyak, maka perlu melakukan ekstraksi agar sinyal tersebut menjadi lebih sedikit tanpa meninggalkan ciri atau pola yang dimiliki pada sinyal tersebut. Untuk melakukan pengenalan aksen maka diperlukan metode klasifikasi terhadap data latih yang dimiliki agar dapat dikenali sinyal yang masuk tergolong suatu kelas aksen yang telah terdefinisi. Pada bidang pengolah suara, metode ekstraksi yang populer adalah MFCC dengan berbagai macam jenis metode klasifikasi yang

digunakan dalam merekognisi sampel suara yang digunakan. Dalam penelitian terkait akurasi penelitian sudah sangat baik, yaitu dengan nilai ukur sebesar 90% kecocokan terhadap data koleksinya masing-masing penelitian. Oleh karena itu, penelitian ini akan mencoba mengukur keakuratan metode MFCC dan metode klasifikasi DTW-KNN terhadap data sampel aksen suku orang Indonesia dengan ekspektasi mendapatkan nilai akurasi kecocokan diatas 90% pada tahap rekognisinya.

Mel Frequency Cepstrum Coeficcient merupakan metode ekstraksi suara yang sangat umum digunakan. Dalam beberapa penelitian sebelumnya mengungkapkan bahwa menggunakan MFCC dalam melakukan ekstraksi memiliki efktivitas dan kecepatan yang sesuai dengan ekspektasi. Setelah hasil ekstraksi didapatkan maka klasifikasi akan menggunkan metode DTW-KNN (*Dynamic Time Warping- K Nearest Neighbour*), dimana DTW akan memproyeksikan matrik berukuran 1 x panjang data latih dan 1 x panjang data uji, dan mengakumulasi nilai dari nilai terkecil yang didapatkan, kemudian KNN akan menghitung nilai terdekat yang sesuai dari hasil DTW. Kelebihan DTW dari metode klasifikasi lainnnya adalah data uji dan data koleksi yang digunakan dapat menggunakan ukuran panjang yang berbeda.

Berdasarkan latar belakang diatas, maka penulis berniat mengajukan penelitian untuk membuat suatu algoritma proses ekstraksi suara dari data koleksi yang telah didapatkan menggunakan metode MFCC, dan mengklasifikasikannya menggunakan metode DTW-KNN. Hasil dari penelitian ini diharapkan mampu mendapatkan nilai akurasi yang diekspektasikan dari algoritma yang dibuat untuk merekognisi suku berdasarkan aksen suara.

1.2 Pernyataan Masalah

Bagaimana akurasi dari metode MFCC dan DTW-KNN dalam merekognisi suku aksen di Indonesia?

1.3 Tujuan Penelitian

Untuk mengukur nilai akurasi dari metode MFCC dan DTW-KNN dalam merekognisi aksen suku di Indonesia.

1.4 Batasan Masalah

Batasan masalah dari penelitian ini:

1. Banyaknya suku dalam data koleksi sebanyak 4 suku, yaitu; Jawa, Kaili, Madura, dan Sunda.
2. Setiap suku (kelas) memiliki 10 data suara, 5 suara laki-laki dan 5 suara perempuan dengan orang yang berbeda-beda, dengan usia responden dari umur 20 hingga 24 tahun.
3. Setiap sampel audio menggunakan pelafalan yang berbeda sesuai dengan bahasanya masing-masing.
4. Panjangnya durasi yang direkam sebanyak 3500 *milisecond*. Jika kurang akan diubah menjadi nilai 0, dan jika lebih akan dipotong terhadap panjang durasi yang telah ditentukan.
5. *File* suara hasil rekaman akan hanya menggunakan 1 kanal.

1.5 Hipotesis

Berdasarkan penelitian sebelumnya dapat dilakukan hipotesis jika menggabungkan metode DTW dan KNN untuk melakukan rekognisi terhadap data set penelitian ini, maka memungkinkan bahwa akurasi

maksimum bisa lebih tinggi dan memaksimalkan nilai dari akurasi minimum terhadap metode yang digunakan penelitian sebelumnya.

1.6 Manfaat Penelitian

Manfaat dari penelitian ini adalah:

Untuk mengetahui nilai akurasi dalam merekognisi suara aksen suku di Indonesia menggunakan metode MFCC dan DTW-KNN. Sehingga hasil akurasi yang didapatkan dapat dilanjutkan ke peneltian pengolah suara yang lebih intensif.

1.7 Sistematika Penelitian

Penulisan skripsi ini tersusun dalam 5 (lima) bab dengan sistematika penulisan:

BAB I PENDAHULUAN

Bab ini berisi tentang penjelasan penulis menyusun proposal penelitian ini beserta manfaat dan tujuan penelitian.

BAB II TINJAUAN PUSTAKA

Bab ini berisi berbagai landasan teori yang digunakan untuk memahami permasalahan yang ada pada penelitian ini, dimana teori-teori tersebut merupakan teori umum tentang ekstraksi suara MFCC dengan proses-prosesnya beserta metode klasifikasi DTW-KNN.

BAB III METODE PENELITIAN

Pada bab ini akan dibahas mengenai analisis permasalahan penelitian dan penjelasan tentang rancangan struktur program dan antarmuka dari aplikasi rekognisi yang dibuat.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi penjelasan pengimplementasian sistem berdasarkan gambaran antarmuka aplikasi yang dibuat dan pengujian aplikasi terhadap data uji untuk membuktikan bahwa aplikasi dapat berjalan dengan baik. Pada bab ini juga akan membahas hasil pengujian berupa presentase akurasi keberhasilan sistem serta efektivitas dari pengujian tersebut.

BAB V PENUTUP

Pada bab ini akan dijabarkan beberapa kesimpulan dari perancangan sistem dan saran untuk pengembangan penelitian lebih lanjut.

BAB II

TINJAUAN PUSTAKA

2.1 Rekognisi Aksen

Menurut KBBI aksen adalah tekanan suara pada kata atau suku kata, atau bisa juga diartikan pelafalan khas yang menjadi ciri khas seseorang; logat. Setiap daerah di Indonesia memiliki sukunya masing-masing. Setiap suku memiliki identitas. Identitas merupakan penggambaran dari suku yang digambarkan, misalnya seperti pakaian adat, rumah adat, bahasa daerah, senjata tradsional, budaya, tarian daerah, dan lain lainnya. Khususnya pada bahasa daerah, tentu memiliki pola kalimat yang digunakan, dari penggunaan kalimat ini jelas akan mempengaruhi aksen seseorang dalam berbicara. Misal suatu suku yang menggunakan bahasa Indonesia dalam berbicara, dikarenakan seseorang tersebut berasal dari sukunya tentu ia akan menggunakan aksen suku aslinya dalam berbicara walaupun dalam menggunakan bahasa yang lain. Aksen yang digunakan memiliki karakteristik/pola yang dapat dikenali. Dalam bidang pengolahan suara input sinyal suara tersebut digambarkan dalam bentuk gelombang transversal, sehingga jika ada dua suku yang berbeda melafalkan suatau kata/kalimat dalam bahasa yang sama (bahasa Indonesia), maka dapat terlihat perbedaan sinyal suara yang masuk tersebut.

Rekognisi atau bisa disebut dengan pengenalan merupakan suatu proses untuk membedakan antara suatu subjek dengan subjek yang lainnya. Dalam penelitian terkait, banyak ditemukan judul tentang rekognisi ataupun deteksi, namun antara keduanya memiliki perbedaan yang signifikan. Jika pada pengolah citra

disebutkan rekognisi wajah, maka fokus penelitian tersebuat adalah bagaimana membedakan wajah dari setiap orangnya (wajah orang ke-satu dengan yang lainnya), sedangkan deteksi wajah adalah bagaimana membedakan suatu gambar itu adalah wajah atau bagian tubuh yang lainnya. Sama halnya dengan rekognisi aksen, fokus rekognisi berarti bagaimana membedakan aksen tersebut dari suku yang satu dengan suku yang lainnya. Berbeda halnya dengan deteksi suara, semisal deteksi gender pria atau wanita, deteksi suara binatang, dan lain semacamnya.

2.2 Ekstraksi Ciri Pada Suara

Menurut Sivaranjan Goswami (2013), Ciri pada pemrosesan sinyal suara dapat dibagi menjadi 6 ciri (*feautures*), yaitu:

a. Zero-Crossing Rate

Zero-crossing rate adalah ukuran frekuensi sinyal selama periode kecil. Zcr dapat diperoleh dengan mengukur berapa kali tanda sinyal berubah dan membaginya dengan dua. Dalam hal ini dapat diakatakan zcr adalah penggunaan dari frekuensi sinyal yang masuk.

b. Mean Square or Magnitude Value

Magnitudo dari suatu sinyal digunakan dalam proses transformasi fourier untuk mendapatkan ciri dari sinyal suara yang diinputkan. Magnitudo menggambarkan besarnya tingkat energi pada sinyal tersebut.

c. Voice Activity Detection

Selama sinyal suara diinputkan, ada banyak jeda (tidak ada suara) dalam perekamannya. Untuk melakukan pemrosesan ucapan, perlu dibedakan antara ada dan tidak adanya ucapan dalam klip audio. Ada atau tidaknya ucapan

dalam bingkai durasi pendek dapat dengan mudah ditentukan jika tidak ada kebisingan (*noise*). Jika ucapan tidak ada, nilai besarnya rata-rata atau nilai kuadrat rata-rata sangat kecil (nilai amplitudo). Di sisi lain, nilai magnitudo rata-rata yang tinggi atau nilai kuadrat rata-rata menunjukkan adanya ucapan.

d. Detection of Voiced and Unvoiced Speech

Suara yang disuarakan dan tidak disuarakan. Ciri ini merupakan bagian dari deteksi aktivitas suara. Tentunya digunakan dalam mengenalali ada atau tidaknya suara dalam kebisingan. Menjadi suatu yang penting bahwa sebagian besar fitur sinyal ucapan diekstraksi untuk ucapan suara (bukan kebisingan). Oleh karena itu identifikasi ucapan yang disuarakan dan tidak disuarakan adalah hal penting lainnya setelah pendeteksian aktivitas suara.

e. Pitch and Pitch Period Estimation

Pitch adalah frekuensi yang dirasakan dari sinyal suara. Dalam beberapa referensi ada yang menggunakan istilah frekuensi dan pitch secara bergantian, hanya saja dalam konteks ini pitch yang dimaksud adalah untuk mengenali seseorang yang sedang berbicara dari perubahan intonasinya. Setiap orang memiliki rentang nada yang khas yang keluar dari laring tenggorokannya. Pria dan wanita memiliki interval pitch yang berbeda. Faktor faktor yang mempengaruhi, yaitu tingkat stress, intonasi, dan emosi.

f. Phonemics and Phonetics of Speech Signal

Fonem adalah suatu istilah linguistik dan merupakan satuan terkecil dalam sebuah bahasa yang masih bisa menunjukan perbedaan makna. Pemisalan dalam bahasa Indonesia bunyi kata “k” dengan “g” terdengar hampir sama namun keduanya merupakan fonem yang berbeda, misal dalam kata cagar dan

cakar. Hal ini menyebabkan sinyal yang masuk akan sangat mirip, oleh karena itu dalam ciri ini masuk dalam pengolah suara yang lebih tinggi. Sama halnya dengan fonetik yang mempelajari bagaimana suatu alfabet dibunyikan dan diklasifikasikan berdasarkan suara yang bersuara (dikarenakan getaran pita suara) dan suara yang tidak bersuara.

g. Prosodic Features: Stress and Intonation of Speech

Dibutuhkan pendalaman materi yang lebih dalam ciri ini. Prosodi merupakan ciri sinyal suara berdasarkan intonasi, tekanan, rima, irama dari fonem yang terbentuk. Sinyal suara yang masuk sudah bercampur dengan emosi (*mood*) seseorang ketika berbicara, sehingga hal ini sudah sangat mempengaruhi pola dari sinyal yang terbentuk. Dibutuhkan mesin pembelajaran yang mampu menyimpan setiap input baru sebagai pembelajaran agar dapat menangani suatu sinyal prosodi.

2.3 *Mel Frequency Cepstrum Coefficient*

Mel Frequency Cepstrum Coefficient atau disingkat MFCC meruapakan metode ekstraksi yang menggunakan transformasi fourier dari sinyal suara yang masuk dalam bentuk gelombang transversal dan diproses hingga mendapatkan nilai cepstrum. Metode ekstraksi MFCC merupakan metode yang cukup populer dalam basis pengolah suara, hal ini dikarenakan metode tersebut dapat mempresentasikan sinyal dengan baik. Proses atau tahapan yang digunakan pada metode MFCC ini, yaitu Pra-Emphasis, Framing, Windowing, Power Spectrum, Filter Bank, dan Cepstrum.

2.4 *Dynamic Time Warping*

Dynamic Time Warping merupakan metode klasifikasi dengan memproyeksikan matriks dari data koleksi dengan data uji kemudian hasil akhir yang didapatkan merupakan akumulasi penjumlahan dari nilai diagonal terkecil. Dalam perhitungannya dynamic time warping merupakan pengembangan dari persamaan *euclidean distance* dimana dari data a dikurang data b kemudian diberikan fungsi positif. Namun dari persamaan *eucliden distance* ini hanya dapat digunakan jika ukuran dimensi dari data set tersebut adalah sama, maksudnya jika data koleksi memiliki ukuran 1 x 10 maka data uji juga harus memiliki ukuran 1 x 10. Oleh karena itu metode *dynamic time warping* hadir dengan menambahkan pembaharuan dimana data koleksi dan data uji dapat diukur walaupun berbeda ukuruan. Hal ini dikarenakan pembuatan proyeksi matriks dari masing masing panjang data dalam proses kalkulasinya, sehingga memecahkan permasalahan klasifikasi jika banyak data dalam 1 dataset tidak sama. Pembahasan lebih lanjut tentang alur dan proses perhitungan dari metode ini dibahas pada BAB III.

2.5 *K-Nearest Neighbour*

*K-Nearest Neighbo*ur juga merupakan metode klasifikasi yang lazimnya digunakan untuk melakukan klasterisasi. Maksud dari klusterisasi yaitu jika dari suatu data set namun tidak diketahui pengelompokan dari data data yang ada didalam data set tersbut. Proses *K-nearest neighbour* yaitu dengan menentukan titik centroid pada perseberan data yang digunakan, kemudian menghitung setiap data terhadap nilai centroid tersebut. Persamaan yang digunakan biasanya juga menggunakan persamaan *euclidean distanc*e. Selain melakukan klusterisasi, *K-nearest neighbour* juga dapat digunakan untuk melakukan klasifikasi jika kelas

dalam dataset telah ditentukan, yaitu dengan menentukan nilai terkecil terhadap hasil dari pengurangan data dan diklasifikasikan berdasarkan banyaknya tetangga K, output diambil berdasarkan banyaknya mayoritas kelas pada banyak tetangga K. Alur dan perhitungan detail telah dijabarkan pada BAB III.

2.6 Penelitian Terkait

Analisis metode MFCC dan DTW diukur pada penelitian verifikasi biometrika suara menggunakan metode MFCC dan DTW (Darma dan Adi, 2011). Pengujian dilakukan terhadap 35 orang, terdiri dari 27 orang laki-laki dan 8 orang perempuan, dimana setiap orang diberikan sampel sebanyak 7. 6 buah sampel sebagai data acuan dan sisa 1 sampel sebagai data uji dari setiap orangnya. Hasil pengujian memperlihatkan tingkat akurasi yang paling rendah adalah 59.664% dan akurasi tertinggi 93.254%. Hasil pengujian didapatkan dari perhitungan probabilitas terhadap rasio kesalahan kecocokan dan rasio kesalahan ketidakcocokan. Analisis lebih detail pada penelitian ini dijabarkan, seperti kecocokan terhadap kata "satu", "dua", hingga "lima", dan didapatkan hasil bahwa penyebutan kata "tiga" mendapatkan akurasi sebesar 87% dan terendah sebesar 66% untuk kata "empat". Kemudian menganalisis terhadap banyaknya data acuan yang diproses, bahwasanya semakin banyak data acuan yang dimasukan maka akurasi pencocokan semakin baik, hal ini dibuktikan dengan uji coba terhadap 1 sampel, 3 sampel, dan 6 sampel. Terlihat 6 sampel memiliki grafik paling tinggi. Namun untuk banyaknya jumlah pengguna dibagi menjadi 10 orang, 20 orang dan, 30 orang, terlihat 10 orang mendapatkan akurasi lebih baik dengan persentase 76.57% ketimbang 35 orang sebesar 76.07%. Setelah itu menganalisis jumlah koefisien MFCC. Banyak koefisen yang digunakan dibagi

menjadi 11, 15, 19, 23. Untuk bagian nilai koefisien yang digunakan didapatkan hasil bahwa 19 koefisien memiliki nilai terbaik dengan 78.06%. Terakhir membandingkan proses framing terhadap panjang sinyal dan sinyal yang dibuat overlap, yaitu N= 20 M=10, N= 30 M= 15, N= 30 M= 15, terhadap koefisien MFCC 23, 23, 25, dimana N adalah panjang sinyal, dan M adalah panjang overlap (dalam milisekon). Didapatkan hasil bahwa percobaan ketiga dengan akurasi tertinggi yaitu 88.50%. Oleh karena itu ditarik kesimpulan baik buruknya sistem dalam melakukan pengenalan dipengaruhi oleh panjang frame, panjang overlapping, jumlah koefisien filter bank, dan jumlah koefisien MFCC.

Implementasi MFCC dan GMM sebagai identifikasi bahasa pada Negara India (Koolagodi, Rastologi, Rao. 2012). Pada Negara India juga memiliki variasi bahasa, seperti Assamesse, Hindi, Kannada, Kashmiri, dan lain sebagainya. Dalam penelitian ini, peneliti melakukan pembandingan hasil ekstraksi menggunakan metode GMM (*Gaussian Mixture Model*) dengan variasi koefisien MFCC yang berbeda-beda. Hasil akurasi pengujian diambil dengan nilai rata-rata dari persentase setiap identifikasi bahasa yang diuji dari variasi koefisien MFCC yang digunakan. Koefisien MFCC sebagai analisis hasil ekstraksi yang digunakan adalah 6, 8, 13, 19, 21, 29, dan 35 dengan hasil (dalam persen) 87.9, 86.2, 87.4, 88.4, 88, 87.6, 88.4. Pada penelitian ini disimpulkan bahwa penggunaan koefisien MFCC sebanyak 19 memiliki akurasi yang paling baik dalam identifikasi bahasa di Negara India.

MFCC metode ekstraksi suara kemudian diterapkan untuk menganaisis suara binatang (A.D. Mane dkk, 2013). Pada pembahasan penelitian ini menggunakan beberapa suara anjing dalam penelitiannya. Setelah suara diekstraksi dengan

MFCC kemudian kecocokan data uji diproses dengan metode DTW dan mengasilkan nilai minimum 0.0 (data uji terdapat pada data koleksi) dan nilai terbesar sebesar 0.5637 dalam data sampel yang sejenis, Pada DTW jika nilai semakin kecil maka objek akan semakin mirip/sama. Metode DTW melakukan proyeksi matriks dengan ordo panjang data koleksi dengan panjang data uji, sehingga memungkinkan jika kedua panjang array masing-masing larik tidak sama dapat ditentukan nilai kemiripannya, hal ini merupakan kelebihan metode DTW. Dalam penelitian ini menekankan bahwa untuk melakukan ekstraksi suara diperlukan melakuakn *Noise Removal* atau dengan kata lain menghilangkan sinyal yang tidak diperlukan (noise). Dalam penelitian ini untuk menghilangkan noise menggunakan metode ZCR (*Zero Cross Rate*).

Jika penelitian sebelumnya mewajibkan perekaman suara pada keadaan yang hening, maka penelitian berikut mengidentifikasi suara katak pada lingkungan yang gaduh/berisik (Jaafar, Ramli, Shahrudin, 2013). Pada penelitian ini mendapatkan modifikasi pada bagian ekstraksinya dengan melakukan perhitungan sinyal awal terhadap SNR (*Signal to Noise Ratio*) sebesar 20dB dan 10dB. Pada analisis MFCC juga menggunakan variasi koefisien MFCC. Nilai yang digunakan adalah 8, 12, 15, 20, dan 25. Kemudian dilakukan identifikasi menggunakan metode KNN (*K- Nearest Neighbour*) pada umumnya. Ketika metode ekstraksi dan pengklasifikasiannya dimodifikasi agar lebih sesuai dengan suara katak akurasi keberhasilan sistem mencapai 59.33% hingga 85.78%.

Penelitian selanjutnya, MFCC dalam mengidentifikasi jenis kelamin seseorang (Yücesoy dan Nabiyev, 2013). Dalam penelitian ini dilakukan percobaan dalam mengenali jenis kelamin seseorang berdasarkan suaranya.

Dilakukan pemodelan GMM (*Gaussian Mixture Model*) dalam analisis ekstraksi sinyal dan menggunakan algoritma ekspektasi maksimum dalam identifikasi output (laki-laki/perempuan). Pada hasil pengujian didapatkan hasil dengan 760 kalimat dari 76 penutur akurasi sebesar 100%, sedangkan percobaan dengan 6100 kalimat dari 610 penutur mendapatkan akurasi keberhasilan sebesar 97.76%.

Setelah digunakan dalam identifikasi MFCC juga digunakan dalam rekognisi musik emosional (Nalini dan Palanivel, 2015). Dalam penelitian ini lebih berfokus dalam rekognisi musik emosional tersebut dimana setelah melakukan ekstraksi MFCC, input dibawa kedalam analisis metode klasifikasi rekognisi *Autoassociative Neural Network* (AANN), *Support Vector Machine* (SVM), dan *Radial Basis Function Neural Network* (RBFNN). Dalam penelitian data sampel yang digunakan sebanyak 200 data dimana setiap kelas memiliki 20 data sampel, kemudian dibagi secara acak 10 data untuk training dan 10 data untuk pengujian. Untuk hasil dari penelitian ini didapatkan hasil pengukuran dari metode AANN, SVM, RBFNN sebesar 96%, 99%, 95%.

Dibidang kesehatan, MFCC diterapkan dalam menganalisis suara penderita bibir sumbing (Anggoro, 2015). Dalam penelitian ini suara yang diucapkan oleh penderita bibir sumbinf dengan suara orang normal stelah diekstraksi akan dibandingkan dengan menggunakan metode DTW. Analisis yang dihasilkan terdapat 30 percobaan dimana masing-masing percobaan akan dianalisis dari metode penyusun MFCC itu sendri hingga metode perbandingan DTW, semisal pada tahap FFT, DCT, dll. Sampel suara yang digunakan merupakan pengucapan huruf tunggal alfabet. Pada akhir penelitian didapatkan hasil bahwa berdasarkan jarak DTW 2 sampel suara normal memiliki nilai perbedaan antara 0 hingga

8.53887 dan suara penderita dengan suara normal memiliki nilai perbedaan sebesar 1.7852 hingga 78.3919.

Ekstraksi MFCC dilakukan terhadap suara burung sebagai *Voice Activity Detection* (VAD) untuk menangkal serangan hama terhadap petani (Muhammad dan Suwito, 2016). Pada ekosistem sawah terdapat berbagai macam spesies burung, baik hama maupun non-hama. Burung non-hama dapat dimanfaatkan petani untuk melawan serangga sedangkan Burung dalam kategori hama dapat merugikan petani dalam hasil panen. Dari hal tersebut digunakan metode ekstraksi suara menggunakan metode MFCC dan JST dalam mengenali suara kicauan burung yang terdeteksi. Jika burung hama terdeteksi maka akan membunyikan suara ledakan senapan agar burun tersebut pergi dari sawah/kebun. Hasil yang diperoleh dari penelitian ini, didapatkan hasil 90% terhadap data sampel yang diunduh dari internet dan 70% ketika dilakukan sampling terhadap perekaman suara yang sebenarnya (realtime). Analisis dilanjutkan pada berbagai keadaan, seperti *outdoor* (taman), *indoor*, *outdoor* (pinggir jalan), *outdoor* (ramai, pasar burung). Dari keadan tersebut didapatkan akurasi yang berbeda-beda dan dipengaruhi oleh noise dan jarak suara yang terekam.

Akurasi pengukuran MFCC yang baik kemudian diimplementasikan terhadap perangkat keras, dimana input suara sebagai kontrol kendali pintu air (Fadli, 2016). Mesin yang dibuat hanya diperkenalkan kata "buka" dan "tutup" dengan data sampel masing-masing sebanyak 15 sebagai data latih dan masing-masing 10 sebagai data uji. Setelah sinyal di ekstraksi kemudian diproses dengan jaringan syaraf tiruan backpropagation. JST *backpropagation* pada tahap *training* melakukan epoch, pada epoch ke 423 dari 5000, epoch dihentikan dan didapatkan

nilai MSE 0.005565. Hasil yang diberikan dari JST *backpropagation* sebesar 1,01325 untuk kata "buka" dan 0.09309 untuk kata "tutup". Basis suara adalah langkah awal dalam penelitian ini, setelah output diidentifikasi maka dilanjutkan ke mikrokontroller sebagai hardware yang akan menggerakan pintu penutup air, dalam penelitian ini menggunakan servo sebagai simulasinya.

Dilakukan suatu percobaan dalam mengekstraksi suara alat musik menggunakan MFCC-KNN dengan alat musik yang berbeda yaitu piano, sruling, biola, trompet, dan cello (Nagawade dan Ratnaparkhe, 2017). Setiap kelas alat musik memiliki banyak sample 30, dengan 18 data untuk dilatih dan 12 data sebagai pengujian hasil. Metode Knn yang digunakan merupakan metode Knn pada umumnya tanpa modifikasi. Pada hasil akhir penelitian didapatkan bahwa piano, trompet, dan cello memiliki akurasi pengukuran sebesar 91.66%, sedangkan seruling dan biola mendapatkan nilai pengukuran sebesar 83.33%.

Berikut daftar dari penelitian terkait yang diringkas dalam bentuk tabel pada **Tabel 2.1.**

Tabel 2.1 Ringkasan penelitian terkait

Identitas Riset	Data Sampel	Metode	Hasil
Analisis metode MFCC dan DTW diukur pada penelitian verifikasi biometrika suara menggunakan metode MFCC dan DTW . Darma dan Adi - 2011	35 sampel. Terdiri: 27 laki-laki 8 perempuan	MFCC DTW	Hasil pengujian memperlihatkan tingkat akurasi yang paling rendah adalah 59.664% dan akurasi tertinggi 93.254%

Identification of Language Using Mel- Frequency Cepstrum Corfficient. Koolagodi, Rastologi, Rao. - 2012	15 sampel.	MFCC GMM	Koefisien MFCC sebagai analisis hasil ekstraksi yang digunakan adalah 6, 8, 13, 19, 21, 29, dan 35 dengan hasil 87.9, 86.2, 87.4, 88.4, 88, 87.6, 88.4 (dalam persen).
Identification & Detection System for Animals from their Vocalization. A.D. Mane dkk - 2013	4 sampel. Suara binatang yang digunakan suara anjing.	MFCC DTW	Hasil pengujian mengasilkan nilai 0.0 (data uji terdapat pada data koleksi) dan nilai terbesar sebesar 0.5637 dalam data sampel yang sejenis.
MFCC Based Frog Identification System In Noisy Environment. Jaafar, Ramli, Shahrudin - 2013	150 sampel dari 15 spesies berbeda.	MFCC KNN	Keberhasilan sistem mencapai 59.33% hingga 85.78%.

Gender Identification of a Speaker Using MFCC and GMM Yücesoy dan Nabiyev - 2013	-760 sampel untuk 76 orang -1600 sampel untuk 160 orang	MFCC GMM	Didapatkan hasil akurasi dengan 76 penutur sebesar 100%, sedangkan percobaan dengan 610 penutur mendapatkan akurasi keberhasilan sebesar 97.76%.
Music emotion recognition: The combined evidence of MFCC and residual phase Nalini dan Palanivel - 2015	200 sampel untuk 10 kelas	MFCC AANN SVM RBFNN	Hasil dari penelitian ini didapatkan hasil pengukuran dari metode AANN, SVM, RBFNN sebesar 96%, 99%, 95%.
Analis Karakter Suara Penderita Bibir Sumbing Terhadap Suara Normal Menggunakan Metode Mel Frequency Cepstrum Coeficient dan Dynamic Time Warping.	5 kelas dengan 30 parameter uji coba.	MFCC DTW	Didapatkan hasil bahwa berdasarkan jarak DTW 2 sampel suara normal memiliki nilai perbedaan antara 0 hingga 8.53887 dan suara penderita

Anggoro - 2015			dengan suara normal memiliki nilai perbedaan sebesar 1.7852 hingga 78.3919.
Pengenalan Suara Burung Menggunakan Mel Frequency Cepstrum Coefficient Dan Jaringan Syaraf Tiruan Pada Sistem Pengusir Hama Burung Muhammad dan Suwito - 2016	Tidak dicantumkan	MFCC JST	didapatkan hasil 90% terhadap data sampel yang diunduh dari internet dan 70% ketika dilakukan sampling terhadap perekaman suara yang sebenarnya (realtime)
Kendali Pintu Air Otomatis Berbasis Speech Recognition Menggunakan Metode MFCC dan Jaringan Syaraf Tiruan Fadli - 2016	60 data sampel. 20 data uji untuk masing -masing kelas	MFCC JST	Pada epoch ke 423 dari 5000 didapatkan nilai MSE 0.005565. Hasil yang diberikan dari JST *backpropagation* sebesar 1,01325 untuk kata “buka” dan 0.09309 untuk kata “tutup”.

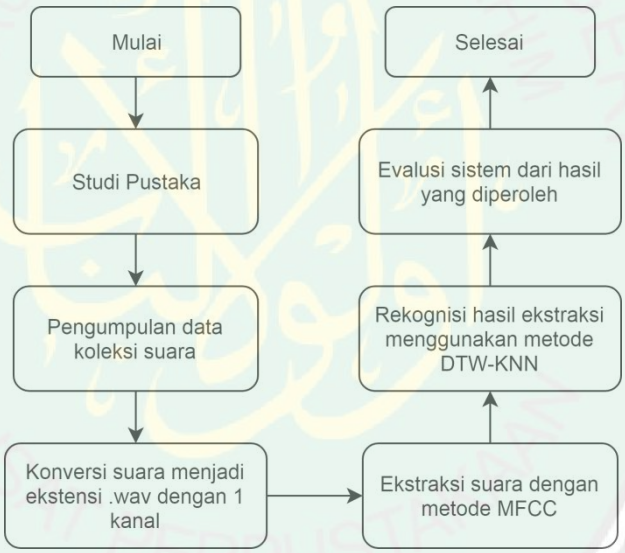
Musical Instrument Identification Using MFCC. Nagawade dan Ratnaparkhe - 2017	30 sampel. Terdiri dari: 18 data latih 12 data uji	MFCC KNN	Pada hasil akhir penelitian didapatkan bahwa piano, trompet, dan cello memiliki akurasi sebesar 91.66%, sedangkan seruling dan biola mendapatkan nilai sebesar 83.33%.

BAB III

METODE PENELITIAN

3.1 Prosedur Penelitian

Pada subab ini akan membahas tentang penelitian yang akan dilakukan. Diaram alur penelitian dapat dilihat pada Gambar 3.1. Penelitian ini terdiri dari 6 tahapan, yaitu: Studi pustaka, pengumpulan data koleksi, konversi suara menjadi ekstensi .wav dengan 1 kanal, ekstraksi suara dengan metode MFCC, rekognisi hasil ekstraksi menggunakan metode DTW-KNN, dan evaluasi sistem dari hasil yang diperoleh.



Gambar 3.1 Prosedur Penelitian

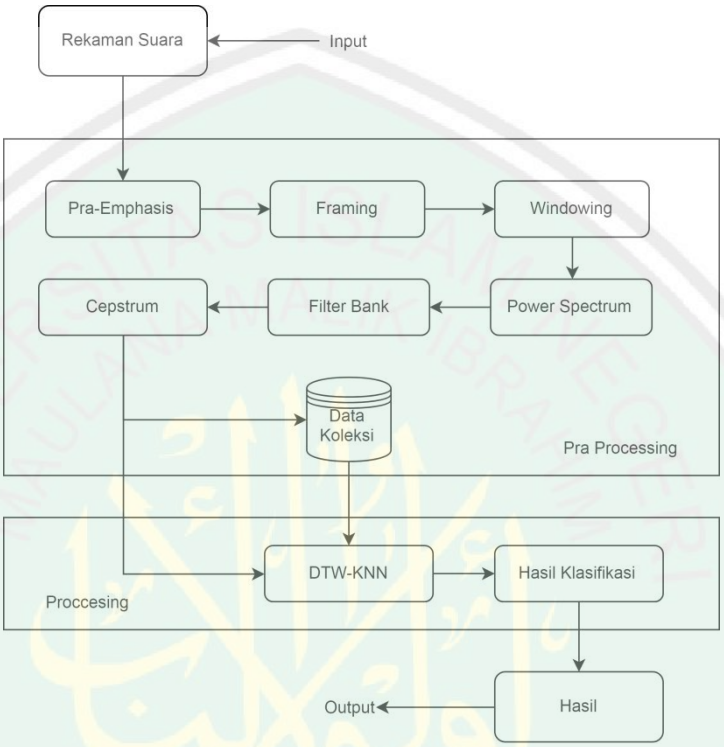
Penelitian ini dimulai dengan mengumpulkan studi pustaka dengan tujuan memahami dasar teori dan aplikasi penerapan yang telah dilakukan oleh peneliti-peneliti sebelumnya berupa metode dan teknologi yang digunakan dalam

menganalisis topik terkait terhadap hasil yang didapatkan. Kemudian Mengumpulkan data koleksi yang berupa suara dengan ucapan/tutur yang sama pada pengucapannya, misal suku A mengucapkan "Assalamu'alaykum", maka suku yang lainnya juga akan mengucapkan kata/kalimat yang sama. Koleksi rekaman suara yang dikumpulkan sebanyak 10 setiap sukunya dengan durasi rekaman selam 3.5 sekon. Tahap elanjutnya adalah mengkonversi *file* rekaman tersebut menjadi berekstensi *.wav* dengan 1 kanal. Kemudian Mengekstraksi suara.wav tersebut menjadi nilai MFCC dengan koefisien yang telah ditentukan. Setelah masing masing suara telah selesai di ekstraksi selanjutnya melakukan rekognisi terhadap suara yang telah ditentukan dengan menggunakan metode klasifikasi gabungan DTW dan KNN untuk mendapatkan *output*/hasil rekognisi yang diinginkan. Untuk menguji tingkat akurasi rekognisi pada sistem ini, maka penelitian ini dilakukan evaluasi dengan menggunakan probabilitas dari tetangga K dalam metode KNN terhadap K kelas paling dekat dengan kesesuain hasil dengan realita.

3.2 Desain Sistem

Desain pada penelitian rekognisi aksen suku Indonesia ini memiliki dua tahapan, yaitu tahap *preprocessing* (pra-proses) dan tahap *processing* (proses). Tahap *preproccesing* merupakan tahap untuk melakukan ekstraksi MFCC terhadap suara pada data koleksi dan data yang akan diuji. Pada tahap ini terdapat 6 proses, yaitu: *Pra-Emphasis*, *Framing*, *Windowing*, *Power Spectrum*, *Filter Bank*, dan *Cepstrum* (Nilai MFCC). *Output* dari MFCC adalah hasil ekstraksi dari sinyal suara yang diinputkan dalam bentuk matriks yang lebih sederahana. Tahap *processing* merupakan tahap untuk melakuan rekognisi data uji terhadap data

koleksi. Tahap *processing* terdiri dari 2 tahap, yaitu: Klasifikasi DTW-KNN, Probabilitas DTW-KNN. *Output* dari tahap *processing* merupkan hasil klasifikasi yang diinginkan berupa kelas/suku yang sesuai dengan data uji yang diinputkan. Berikut alur desian sistem pada penelitian ini.



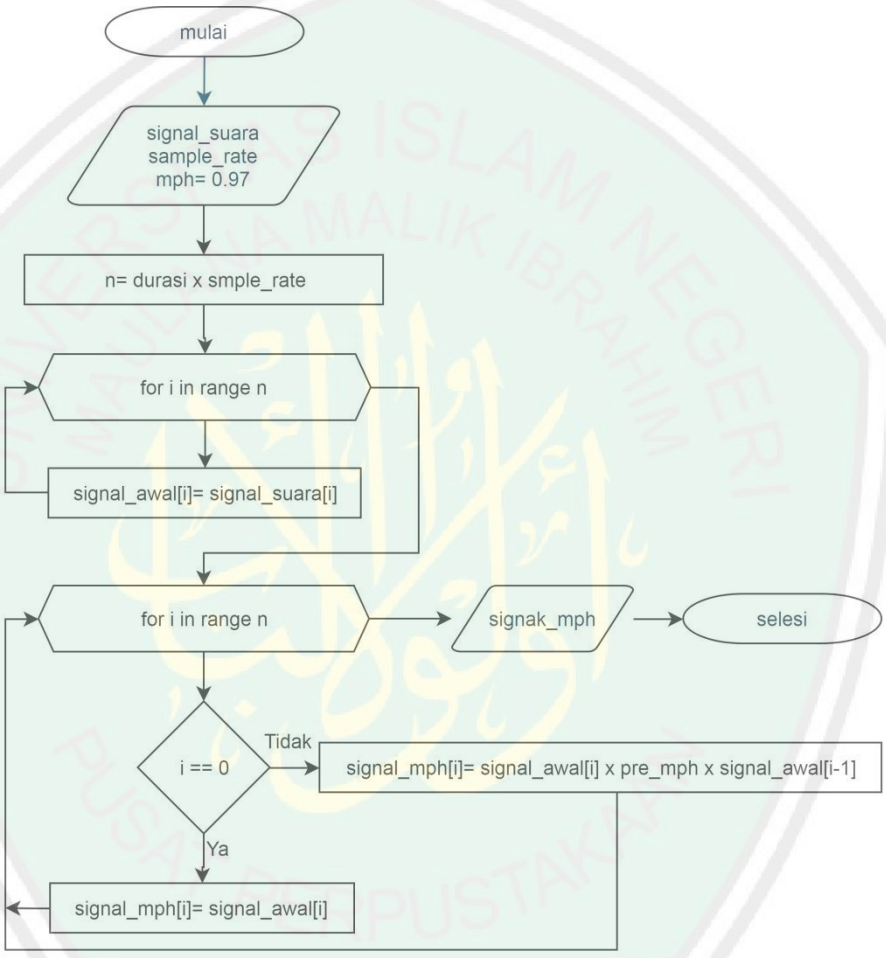
Gambar 3.2. Desain Sistem

3.2.1 *Pra-Emphasis*

Pra-emphasis adalah suatu jenis *filter* yang bertujuan untuk memperkuat/mempertahankan frekuensi tinggi. *Filter pra-emphasis* memiliki beberapa fungsi seperti menyeimbangkan spektrum frekuensi karena frerkuensi tinggi biasanya memiliki besaran yang lebih kecil dibandingkan dengan frekuensi rendah, menghindari masalah numerik selama operasi transformasi *Fourier* (FFT), dan juga meningkatkan sinyal Rasio terhadap Suara (SNR). Berikut persamaan yang digunakan pada tahap *pra-emphasis*.

$$signal\_mph(i) = signal(i) - \alpha \times signal(i-1) \quad (3.1)$$

Pada kebanyakan penelitian nilai populer yang mengisi nilai pada variabel alpa adalah 0,95 dan 0,97, namun nilai yang diusulkan pada penelitian ini adalah 0,97. Alur proses tahap pre-emphasis pada penelitian ini digambarkan pada diagram *flowchart* berikut:



Gambar 3.3 *Flowchart Pre-Emphasis*

Langkah pengerjaan pada tahap ini cukup mengalikan persamaan 3.1 terhadap nilai sinyal yang masuk (*Signal*[x]). Berikut contoh proses perhitungan pada tahap *Pra- emphasis*:

Diketahui sampel nilai :

$$signal = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]$$

$$signal\_mph[0] = signal[0]$$
$$signal\_mph[0] = 1$$

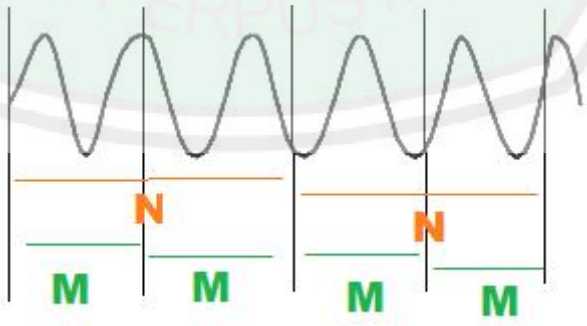
$$signal\_mph[1] = signal[1] - 0.97 \times signal[0]$$
$$signal\_mph[1] = 2 - 0.97 \times 1$$
$$signal\_mph[1] = 1.03$$

Nilai keseleruhan *signal* mph pada sampel data awal menjadi:

$$signal\_mph = [1,1.03,1.06,1.09,1.12.....,1.39,1.42]$$

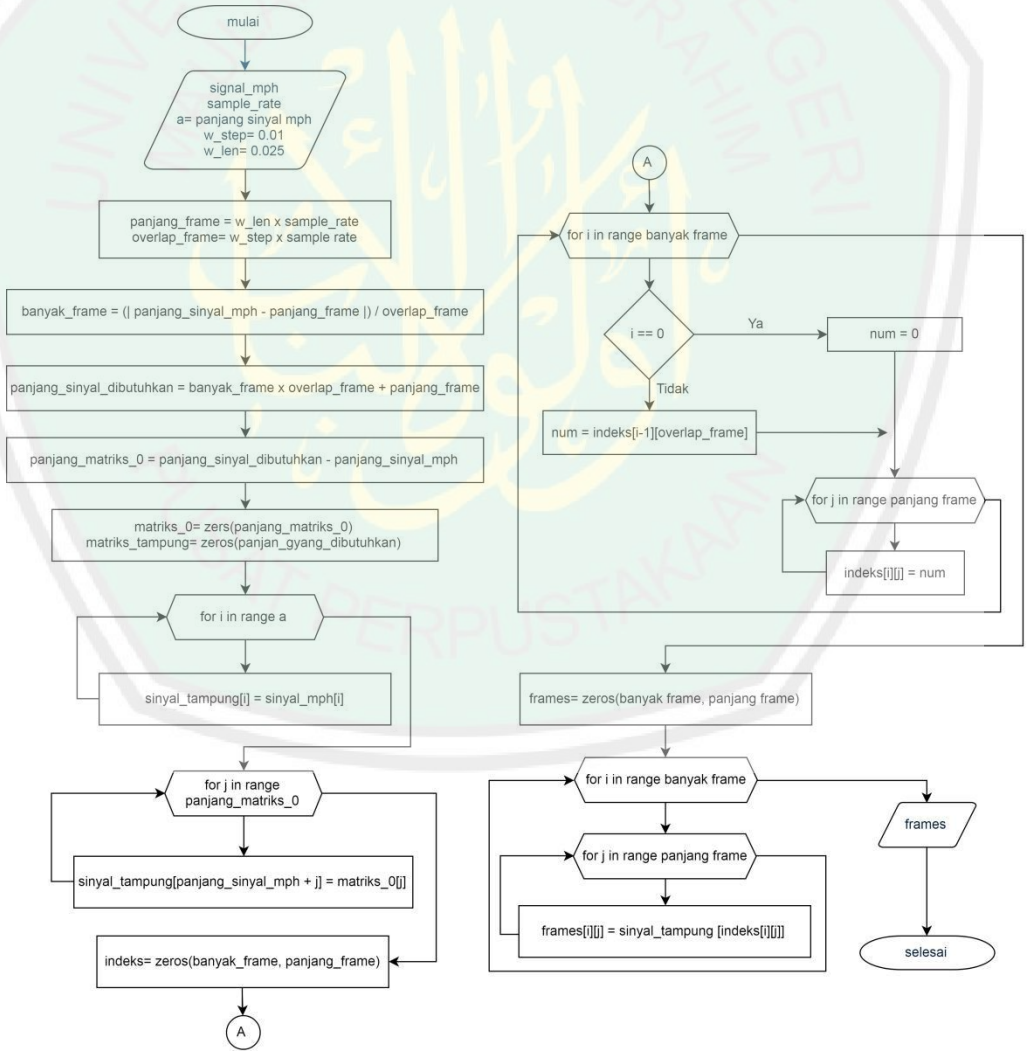
3.2.2 *Framing*

Setelah tahap *pra-emphasis*, maka sinyal perlu dibagi menjadi dalam bentuk *frame per frame*. Tujuan dari *framing* adalah frekuensi dalam sinyal berubah dari waktu ke waktu, sehingga jika dilakukan tranformasi *Fourier* terhadap sinyal yang berubah ubah dapat menyebabkan hilangnya kontur frekuensi sinyal dari waktu ke waktu. Oleh karena itu, untuk menghindarinya dapat dengan mengasumsikan bahwa frekuensi dalam suatu sinyal adalah diam selama periode waktu yang sangat singkat. Dengan melakukan transformasi *Fourier* pada *frame* yang singkat/pendek ini, dapat diperoleh perkiraan yang baik dari kontur frekuensi sinyal dengan menggabungkan *frame* yang berdekatan (*overlap frame*).



Gambar 3.4 Simulasi *framing* M= panjang *frame*, N= panjang *overlap*

Pada kebanyakan penelitian ukuran frame tipikal dalam rentang pemrosesan bicara dari 20 ms hingga 40 ms dengan 50% (+/-10%) tumpang tindih antara frame selanjutnya dengan berturut-turut. Framing yang dilakukan pada penelitian ini menggunkan panjang bingkai sebesar 25 ms, dan 10 ms tumpang tindih. Jika diketahui Sample Rate yang digunakan adalah 44.100hz (kecepatan pemutaran pada DVD dalam membaca 1 gelombang), maka panjang frame sebesar 0.025 x 44.100 = 1.102 dan panjang elemen frame yang overlap, yaitu 0.01 x 44.100= 441. Berikut proses *framing* pada penelitian ini yang digambarkan pada digram *flowchart*:



Gambar 3.5 *Flowchart Framing*

Setelah mengamati *flowchart* diatas, maka langkah-langkah yang harus dilakukan adalah:

a) Menentukan panjang sinyal *frame* yang akan dibuat.

b) Menentukan panjang sinyal yang *overlap* dalam setiap *frame*nya.

c) Menghitung nilai banyak *frame* yang terbentuk.

d) Menyesuaikan panjang sinyal yang sebenarnya dengan identitas *frame* yang telah dibuat, jika kurang akan dipotong, dan jika lebih akan diberikan nilai 0 pada indeks sisa.

Untuk contoh perhitungan *Sample rate* akan dibuat menjadi 360 (data dumi, untuk mempermudah pemisalan), dengan panjang *frame* 5, banyak *frame* 5, dan elemen *frame* yang *overlap* sebanyak 2. Berikut nilai *frame* dari nilai *signal_mph*.

$$frame = \begin{bmatrix} 1 & 1.03 & 1.06 & 1.09 & 1.12 \\ 1.09 & 1.12 & 1.15 & 1.18 & 1.21 \\ 1.18 & 1.21 & 1.24 & 1.27 & 1.3 \\ 1.27 & 1.3 & 1.33 & 1.36 & 1.39 \\ 1.36 & 1.39 & 1.42 & 0 & 0 \end{bmatrix}$$

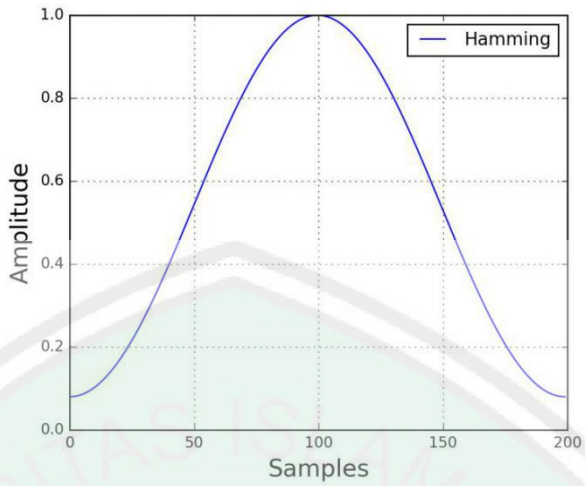
3.2.3 *Windowing*

Setelah mengiris sinyal menjadi *frame*, selanjutnya menggunakan hitung nilai *frame* sebelumnya menggunakan fungsi *window*. Ada banyak variasi jenis fungsi *Window*, namun pada penelitian ini akan menggunakan fungsi *Window Hamming* dengan persamaan:

$$w[i] = 0.54 - 0.46\cos\left(\frac{2\times\Pi\times i}{N-1}\right), \text{N = panjang frame} \qquad (3.2)$$

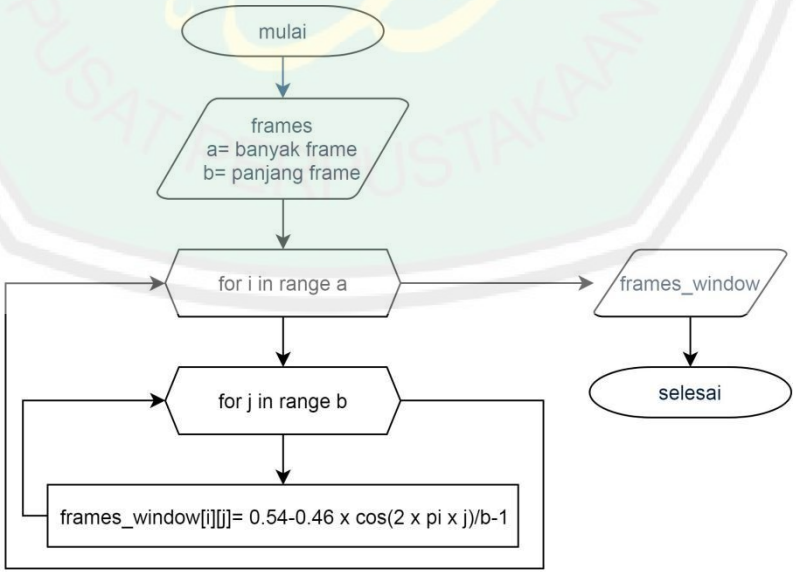
Pada tahap ini langkah-langkah yang digunakan juga sama seperti tahap *pra-emphasis*, yaitu mengalikan setiap nilai *frame* dengan persamaan 3.2 diatas. Sesuai dengan *flowchart* pada gamabar 3.7.

Jika dilakukan *plotting*, maka akan didapatkan grafik sebagai berikut:



Gambar 3.6 *Ploting* data sampel dengan fungsi *Window Hamming*

Penggunaan fungsi *window* memiliki beberapa alasan untuk diterapkan dalam ekstraksi sinyal suara. Terutama untuk menangkal asumsi yang dibuat oleh transformasi *fourier* dimana data tidak terbatas (dapat berubah) dan untuk mengurangi kebocoran spektral (ketidaksesuaian sinyal terhadap amplitudo sinyal berikutnya). Berikut proses *Window Hamming* pada penelitian ini dalam bentuk diagram *flowchart*:



Gambar 3.7 *Flowchart Windowing*

Berikut contoh perhitungan pada proses *Windowing*:

$$frame\_window[0] = frame[0] \times (0.54 - 0.46 \times \cos\left(\frac{2 \times 3.14 \times 0}{5-1}\right))$$

$$frame\_window[0] = 1 \times (0.54 - 0.46 \times \cos(0))$$

$$frame\_window[0] = 1 \times (0.54 - 0.46 \times 1)$$

$$frame\_window[0] = 1 \times (0.08)$$

$$frame\_window[0] = 0.08$$

Sehingga didapatkan nilai :

$$frame\_window = \begin{bmatrix} 0.08 & 0.5562 & 1.06 & 0.5886 & 0.0896 \\ 0.0872 & 0.6048 & 1.15 & 0.6372 & 0.0968 \\ 0.0944 & 0.6534 & 1.24 & 0.6858 & 0.104 \\ 0.1016 & 0.702 & 1.33 & 0.7344 & 0.1112 \\ 0.1088 & 0.7506 & 1.42 & 0 & 0 \end{bmatrix}$$

3.2.4 *Power Spectrum*

Tahap selanjutnya adalah melakukan transformasi *fourier*, untuk menghitung spektrum frekuensi, perhitungan ekstraksi menggunakan banyak nilai FFT *default* sebagai NFFT, biasanya nilai NFFT adalah 256 ataupun 512. NFFT untuk penelitian ini akan menggunakan 512. Berikut persamaan FFT:

$$FFT[i] = \sum_{j=0}^{N-1} frame\_window[j] \times e(-2 \times \Pi \times C \times j \times \left(\frac{i}{N}\right)) \quad (3.3)$$

Pada persamaan FFT diatas terdapat C yang berarti adalah bilangan kompleks, jika dimatlab disimbolkan dengan 1i sedangkan pada python disimbolkan dengan 1j. Bilangkan kompleks memiliki bentuk a+bj, dimana a adalah bilangan riil dan b adalah bilangan imajiner. Nilai N adalah hasil dari (NFFT/2)+1.

$$RFFT[i] = \left| \sum_{j=0}^{\frac{NFFT}{2}+1} frame\_window[j] \times e(-2 \times \Pi \times C \times j \times \left(\frac{i}{\frac{NFFT}{2}+1}\right)) \right| \quad (3.4)$$

Pada persamaan RFFT, RFFT merupakan nilai FFT yang diberikan nilai *absolute* dan dibatasi akumulasinya berdasarkan banyaknya nilai FFT yang digunakan (NFFT), namun nilai yang berada pada RFFT merupakan bilangan kompleks, sehingga bilangan tersebut harus mendapatkan perlakuan khusus. Berikut cara memberikan fungsi *absolute* terhadap bilangan kompleks:

$$|a+bi| = \sqrt{a^2+b^2}$$

$$nilai = |3+6i|$$
$$nilai = \sqrt{3^2+6^2}$$
$$nilai = \sqrt{9+36}$$
$$nilai = \sqrt{45} = 3\sqrt{5}$$

$$Periodogram = (1/N) \times (RFFT^2) \quad (3.5)$$

Terakhir untuk mendapatkan nilai *power spektrum*/spektrum daya dapat menggunakan persamaan diatas.

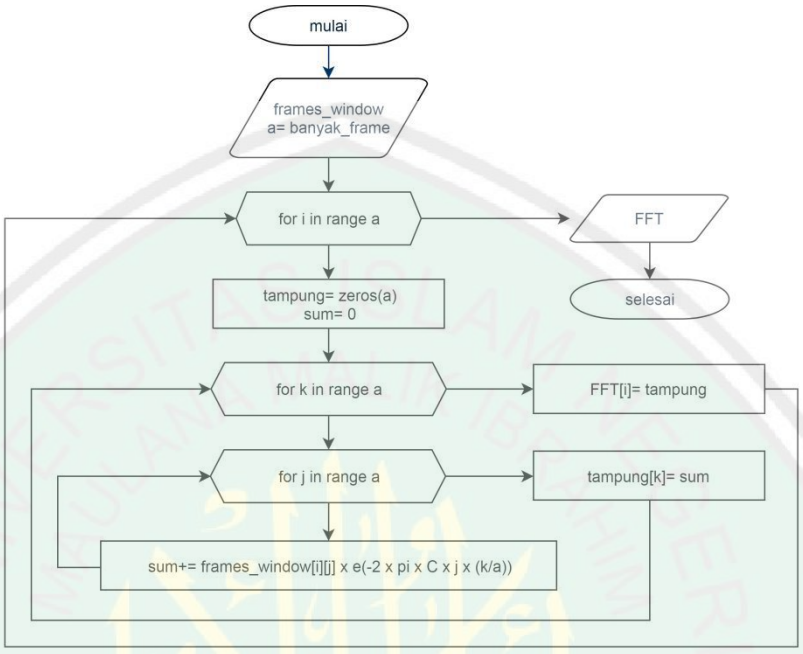
Pada tahap ini langkah-langkah yang dilakukan:

a) Menentukan nilai N dari NFFT.

b) Mendapatkan nilai baru dari persamaan RFFT pada persamaan 3.4 dengan contoh perhitungan yang akan dijelaskan dibagian akhir subab ini.

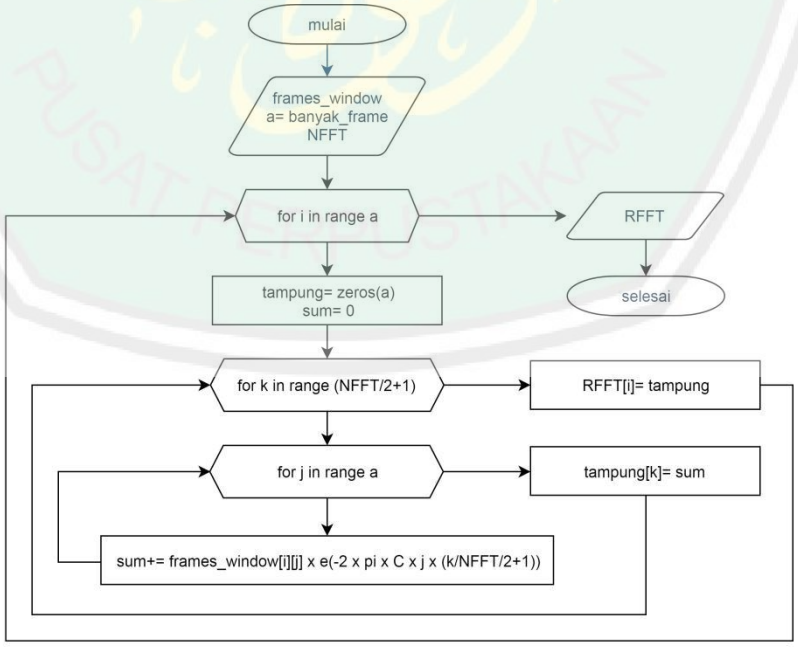
c) Tentukan nilai *periodogram* sesuai dengan persamaan 3.5.

Berikut adalah *flowchart* FFT, dalam penelitian ini FFT pada *flowchart* ini tidak digunakan, hanya saja sebagai pembanding dengan *flowchart* RFFT dibawah sebagai pembanding dimana didalam tahap RFFT sudah terdapat tahap FFT:



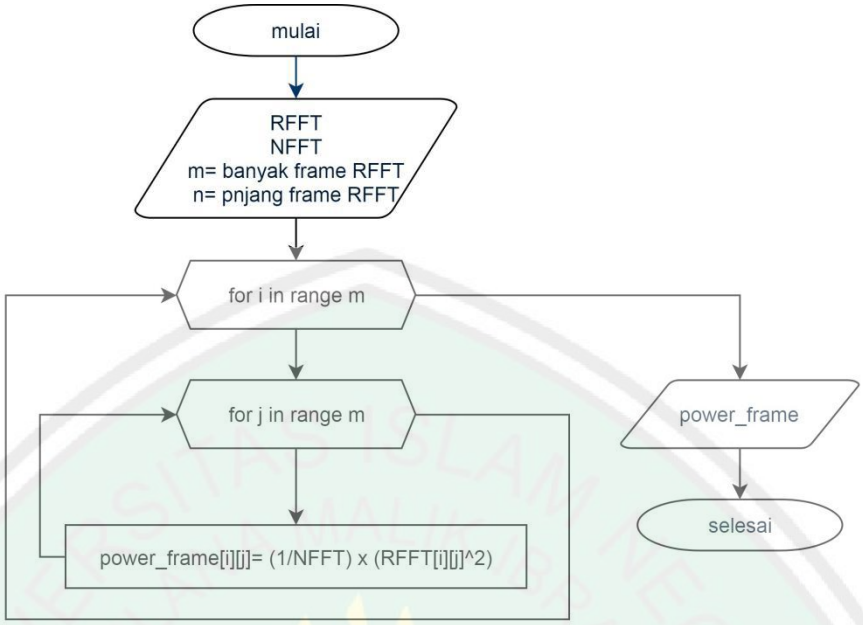
Gambar 3.8 *Flowchart FFT*

Berikut diagram *flowchart* pada tahap RFFT:



Gambar 3.9 *Flowchart RFFT*

Digambarkan diagram *flowchart* untuk mendapatkan nilai *periodogram*:



Gambar 3.10 *Flowchart Periodogram*

Perhitungan pada tahap ini adalah perhitungan yang paling panjang, contoh perhitungan ini dengan menggunakan nilai dummi NFFT sebesar 2, sehingga didapatkan hasil akhir *periodogram* dalam contoh perhitungan berikut:

Penelitian ini akan menggunakan RFFT sebagai langkah awal transformasi *fourier*:

NFFT= 2

N=(2/2)+1, N= 2

Hal ini berarti hanya akan menggunakan *frame_window*[1] dan *frame_window*[2]

Melanjutkan nilai dari *frame_window* dan mengacu pada persamaan 3.4 untuk mendapatkan nilai RFFT:

$$sum0 = 0.08 \times e(-2 \times 3.14 \times (1 + 0\,j) \times 0 \times \frac{0}{2})$$

$$sum0 = 0.08 \times e(0 + 0\,j)$$

$$sum0 = 0.08 \times (1 + 0\,j)$$

$$sum0 = 0.08 + 0\,j$$

$$sum1 = 0.5562 \times e(-2 \times 3.14 \times (1+0j) \times 1 \times \frac{0}{2})$$
$$sum1 = 0.5562 \times e(0+0j)$$
$$sum1 = 0.5562 \times (1+0j)$$
$$sum1 = 0.5562 + 0j$$

$$RFFT[0] = |(0.08+0j)+(0.5562+0j)|$$
$$RFFT[0] = |0.6362+0j|$$
$$RFFT[0] = \sqrt{0.6362^2+0^2}$$
$$RFFT[0] = \sqrt{0.6362^2}$$
$$RFFT[0] = 0.6362$$

Hasil akhir RFFT berupa:

$$RFFT = \begin{bmatrix} 0.6362 & -0.4762 \\ 0.692 & -0.5176 \\ 0.7478 & -0.559 \\ 0.8036 & -0.6004 \\ 0.8594 & -0.6418 \end{bmatrix}$$

Mengukur nilai *periodogram* dapat mengacu pada persamaan 3.5, sehingga didapatkan perhitungan:

$$Periodogram[0] = (\frac{1}{2}) \times (0.6362^2)$$
$$Periodogram[0] = 0.5 \times 0.40475044$$
$$Periodogram[0] = 0.20237522$$

Berikut hasil akhir dari nilai *periodogram*:

$$periodogram = \begin{bmatrix} 0.2024 & 0.1133 \\ 0.2394 & 0.1339 \\ 0.2796 & 0.1562 \\ 0.3228 & 0.1802 \\ 0.3692 & 0.2059 \end{bmatrix}$$

3.2.5 *Filter Bank*

Langkah selanjutnya adalah menghitung nilai *filter bank* yang terbentuk dari *periodogram*. *Filter bank* menerepakan sistem *filter* segitiga. Umunya nilai yang digunakan pada *bank filter* adalah 40 (*nfilt*= 40) pada skala mel ke spektrum daya untuk mengekstraksi frekuensi. Skala mel bertujuan untuk meniru persepsi suara telinga manusia *non-linear*, dengan menjadi lebih diskriminatif pada frekuensi yang lebih rendah dan kurang deskriminatif pada frekuensi yang lebih tinggi. Pada *filter bank* dapat mengkonversi antara *Hertz* dan *Mel* menggunakan persamaan berikut:

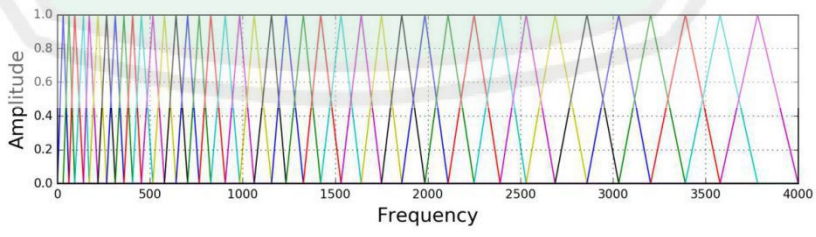
Hertz ke *Mel*:

$$m = 2595 \log 10(1 + \frac{f}{700}) \tag{3.6}$$

Mel ke *Hertz*:

$$f = 700(10^{m/2595} - 1) \tag{3.7}$$

Filter ini disebut menggunakan filter segitiga dikarenakan memiliki respon 1 pada frekuensi tengah dan menurun secara linear menuju 0 hingga mencapai frekuensi tengah dari dua filter yang berdekatan dimana responnya adalah 0, seperti ditunjukan pada gambar dibawah ini:



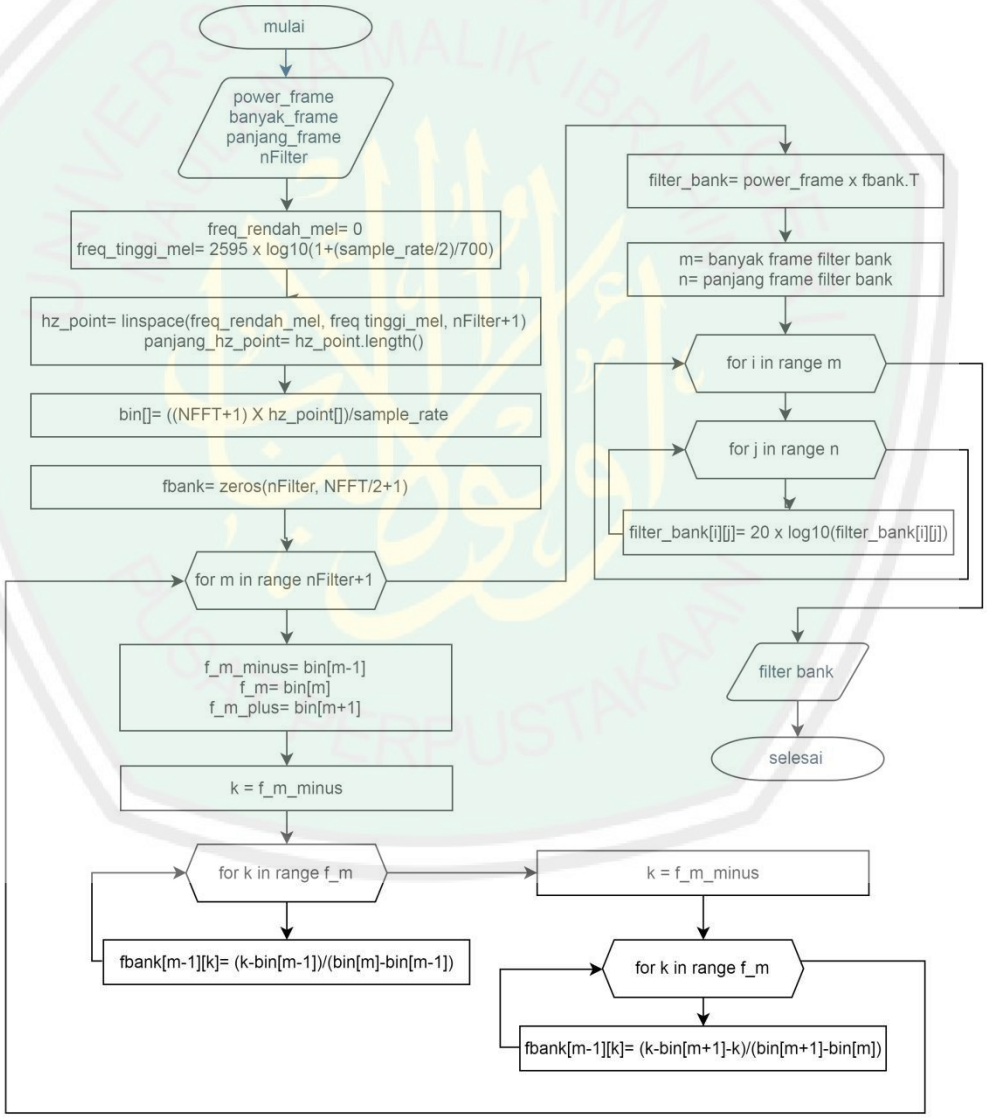
Gambar 3.11 *Filter* segittiga pada *Filter bank*

(Sumber gambar:https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html*)*

Untuk mendapatkan hasil yang demikin dapat menggunakan persamaan *filterbank* berikut:

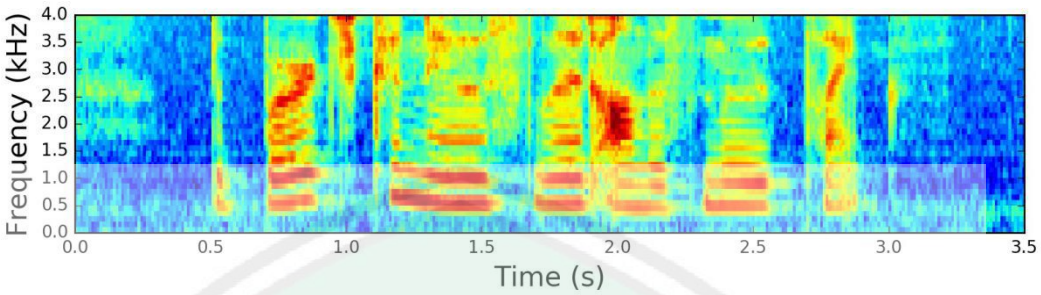
$$Hm(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) <= k < f(m) \\ 1 & k = f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) < k <= f(m+1) \\ 0 & k > f(m-1) \end{cases} \quad (3.8)$$

Berikut *flowchart* alur proses dari *filterbank* yang digunakan:



Gambar 3.12 *Flowchart Filterbank*

Jika *periodogram* telah diimplemtasikan *filterbank*, maka akan mendapatkan spektogram seperti pada gambar berikut:



Gambar 3.13 Spektogram *periodogram* dengan *filter bank*

Dari penjabaran *flowchart* tersebut, tahap *filterbank* juga cukup panjang, berikut tahapan yang dilakukan dalam mengimpementasi nilai *filterbank* terhadap nilai *periodogram*:

a) Tentukan nilai *nFilter*.
b) Tentukan nilai frequensi rendah dan frequensi tinggi.
c) Buat *interval* data antara nilai frequensi rendah dan frequensi tinggi dengan panjang data adalah *nFilter* + 1. Dalam bahasa pemrogramman, fungsi ini disebut *linspace*.
d) Tentukan nilai *hz_points* dengan persamaan 3.7, dimana nilai m adalah nilai interval pada tahap c.
e) Tentukan nilai *bin* dengan persamaan sesuai pada *flowchart*.
f) Dapatkan nilai *filter bank* dengan menggunakan persamaan 3.8.
g) Lakukan transpos matriks *filter bank* yang sudah didapatkan.
h) Tujuan transpos matriks adalah agar dimensi matrik *filter bank* dapat dikalikan dengan dimensi matriks *periodogram*.
i) Terakhir, *filter bank* merupakan hasil dari 20 x 10log(*filter_bank*[i][j]).

Sampai pada tahap ini didapatkan hasil pada contoh perhitungan:

$$filter\_bank = \begin{bmatrix} -13.6967 & 0 \\ -12.2362 & 0 \\ -10.889 & 0 \\ -9.6389 & 0 \\ -8.4727 & 0 \end{bmatrix}$$

3.2.6 *Cepstrum*

Tahap akhir dari MFCC adalah mendapatkan nilai cepstrum yang diperoleh dengan menggunakan metode DCT (*Discrite Cosine Transform*). Cepstrum merupakan transformasi Fourier dari logaritma spektrum suatu sinyal dengan operasi logaritma. Pada umumnya pada ASR (rekognisi suara otomatis) koefisien cepstrum berjumlah 2-13 yang dihasilkan dan dipertahankan, sedangkan sisanya dibuang. Alasan membuang koefisien lainnya adalah bahwa koefisien koefisien tersebut mewakili perubahan yang cepat dalam koefisien *filter bank*. Untuk menghitung nilai DCT yang dihasilkan dapat menggunakan persamaan berikut:

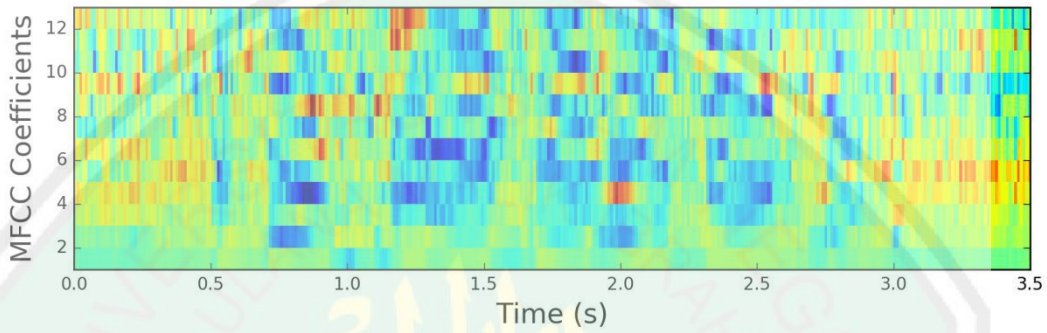
$$y[i] = 2 \times \sum_{n=0}^{N-1} filter\_bank[n] \times \cos(\Pi \times i \times \frac{(2n+1)}{(2 \times N)})$$

$$DCT[i] = \begin{Bmatrix} y[i] \times \sqrt{\frac{1}{4 \times N}}, i = 0 \\ y[i] x \sqrt{\frac{1}{2 \times N}}, lainnya \end{Bmatrix} \quad (3.9)$$

Kemudian untuk mengurangi penekanan MFCC yang lebih tinggi dapat menggunkan fungsi *Sinus* yang telak diklaim bahwa dapat meningkatkan pengenalan pada suara sinyal bising. Persamaan *sinus* yang diberikan berupa berikut:

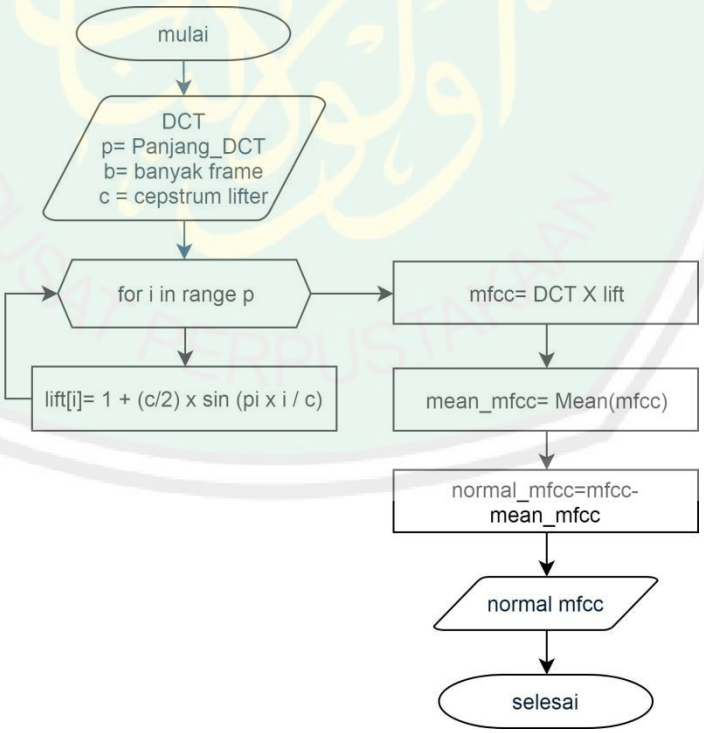
$$n = 1 : koefisiencepstrum$$
$$lift = 1 + (22/2) \times \sin(\Pi \times n \div 22)$$
$$mfcc = mfcc \times lift \quad (3.10)$$

Pada tahap akhir lakukan normalisasi dengan mengurangi nilai *mean* terhadap masing masing elemen nilai mfcc yang ada. Berikut spektogram setelah proses normalisasi mfcc:



Gambar 3.14 Spektogram mfcc normal

Berikut *flowchart* MFCC dalam penelitian ini:



Gambar 3.15 *Flowchart* MFCC

Pada *Cepstrum* merupakan tahap akhir dari MFCC, dimana nilai *cepstrum* merupakan hasi akhr dari tahp ini, beriukut penjabaran pengerjaan pada tahap ini:

a) Tentukan panjang koefisien yang akain diguankan, misal ada 4.
b) Tentukan nilai DCT dari nilai *filter bank* yang telah didapatkan pada persamaan 3.9, proses perhitungan ini hampir sama dengan perhitungan FFT ataupun RFFT.
c) Dari nilai DCT yang diperolah ambil nilai hanya pada indeks 1 hingga 3 (batas panjang koefisien - 1).
d) Tentukan nilai *lifter* sesuai dengan *flowchart* telah digambarkan atau sesuai dengan persamaan 3.10.
e) Nilai *cepstrum*/MFCC diperoleh dengan menggunakan persamaan 3.10.
f) Hasil akhir didapatkan dengan menormalisasi nilai *cepstrum*, yaitu dengan mengurangi masing masing nilai *cepstrum* terhadap nilai rata-rata *cepstrum*.

Pada hasil akhir contoh perhitungan didapatkan nilai:

$$mfcc\_normal = \begin{bmatrix} -2.0538 & 2.053891 \\ -0.232464 & 0.232464 \\ -0.77827 & 0.77827 \end{bmatrix}$$

3.2.7 *DTW-KNN*

DTW dan KNN adalah metode klasifikasi yang berfungsi untuk mengklasifikasi kecocokan data uji terhadap data koleksi. DTW memproyeksikan matrik dari data latih terhadap data uji dengan persamaan *euclidean distance*, kemudian menjumlahkan nilai secara diagonal dengan memilih nilai minimum setiap perpindahan indeks matriksnya. Sedangkan KNN

akan mengurutkan atau mencari kelas terdekat dan menyesesuaikannya terhadap kelas yang sebenarnya.

Berikut persamaan DTW dalam memproyeksikan matriks persamaan:

$$dist(i,j) = |koleksi[i] - uji[0]| + disc[i-1,0], j = 0$$
$$disc(i,j) = |koleksi[0] - uji[j]| + data[0, j-1], i = 0$$
$$disc(i,j) = |koleksi[i] - uji[j]| + \min(data[i-1, j-1], data[i-1, j], data[i, j-1]), lainnya$$

(3.11)

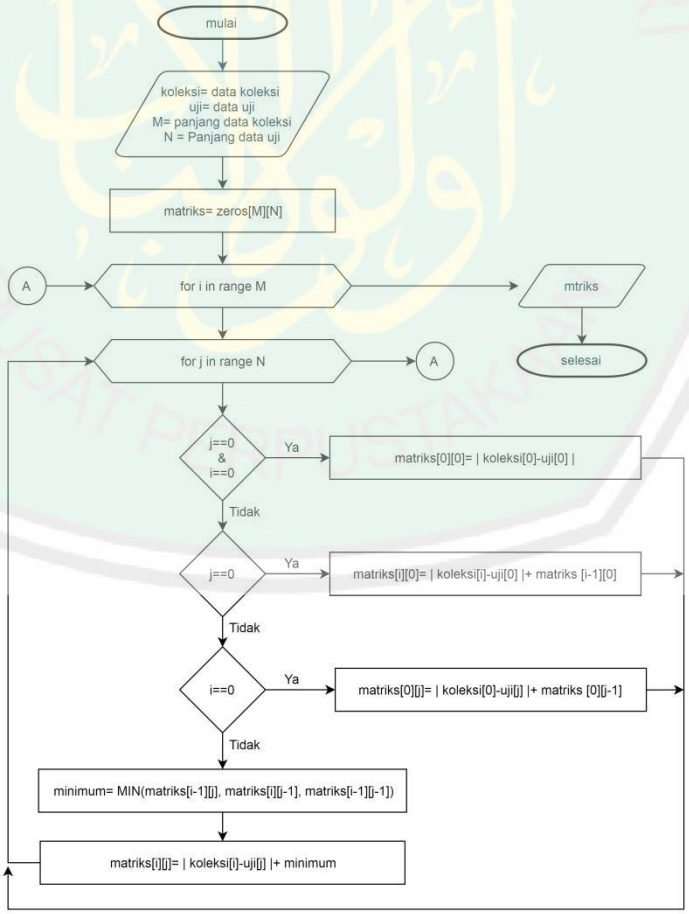
Berikut langkah-langkah dalam menjalankan metode Dynamic Time Warping:

a) Buat nilai a sebagai data uji dengan ukuran matriks 1 x panjang data uji.
b) Buat nilai b sebagai data koleksi/ data latih yang akan dibandingkan dengan data uji dengan ukuran matriks 1 x panjang data koleksi/latih.
c) Susun vektor a secara horisontal, dan vektor b secara vertikal seperti pada gambar 3.18.
d) Untuk memenuhi matriks yang telah dibuat, untuk koordinat x = 0 dan y =0, maka cukup kurangkan vektor a pada indeks 0 dan vektor b pada indeks 0, dengan fungsi absolute(mengubah bilangan menjadi positif).
e) Jika koordinat x bernilai 0, sedangkan koordinat y tidak bernilai 0, maka kurangkan nilai vektor a pada indeks 0 dan vektor b pada indeks y, kemudian berikan fungsi absloute dan jumlahkan dengan nilai pada koordinat x= 0 dan y = y-1 (nilai sebelumnya).
f) Sama halnya jika y bernilai 0, dan x bernilai tidak 0, maka kurangkan nilai vektor b pada indeks 0 dan vektor a pada indeks x, kemudian berikan fungsi absolute dan jumlahkan dengan nilai pada koordinat x= x-1 dan y = 0 (nilai sebelumnya).
g) Jika koordinat x dan y tidak sama dengan 0, maka sesuai dengan persamaan 3.11, didapatkan nilai dengan mengurangi nilai pada vektor a

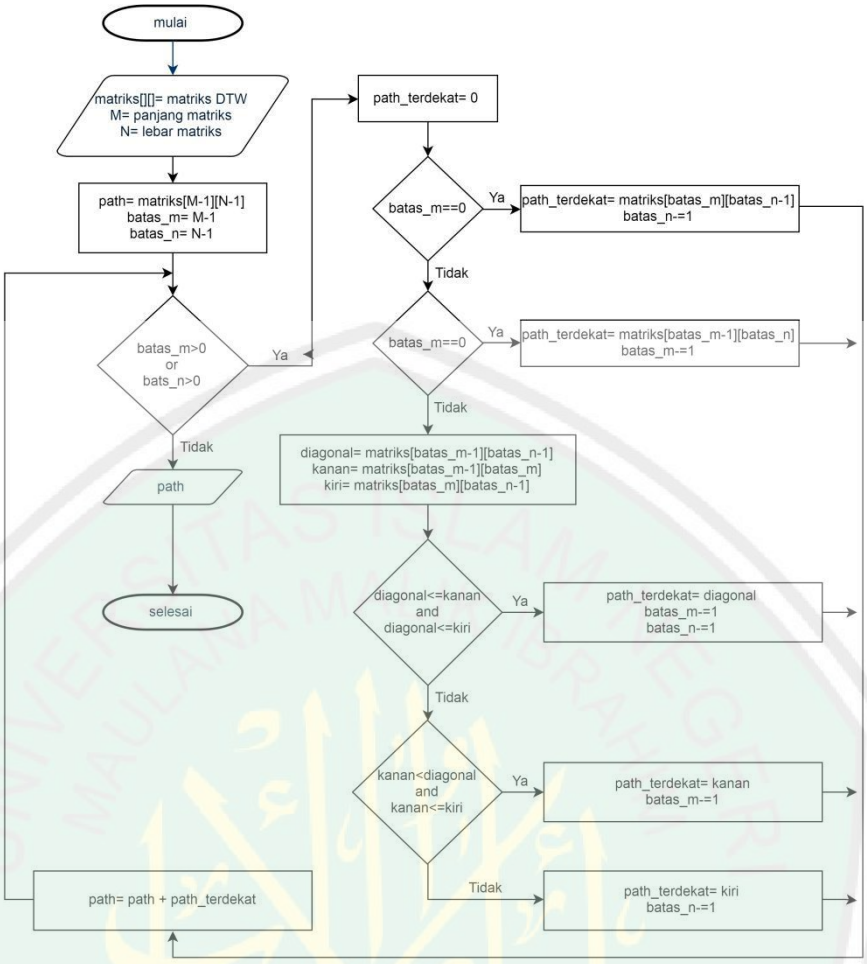
dengan indeks x dan vektor b pada indeks y dengan nilai absolute dan jumlahkan dengan nilai paling rendah/minimum dari nilai : samping kiri koordinat, samping kanan koordinat, dan samping kiri atas koordinat.

h) Jika nilai pada matriks telah terpenuhi, maka tahap selanjutnya adalah menghitung nilai akumulasi minimum secara diagonal dari indeks x = panjang vektor a dan indeks y = panjang vektor b. Maka didapatkan nilai DTW dari dua sampel data tersebut. Disimulasikan pada gambar 3.18.

Dalam Algoritma DTW terdapat 2 proses, yaitu proses untuk membuat matriks dan proses untuk mendapatkan nilai diagonal terkecil. Berikut algoritma DTW-KNN dalam bentuk diagram *flowchart* yang digunakan dalam penelitian ini, yang digambarkan pada **Gambar 3.16**, **Gambar 3.17**, dan **Gambar 3.18**:



Gambar 3.16 *Flowchart* DTW dalam membuat matriks



Gambar 3.17 *Flowchart* DTW dalam mencari nilai diagonl terkecil

Berikut tahapan pada metode KNN setelah nilai DTW didapatkan:

a) Didapatkan nilai DTW dalam bentuk array a 1 dimensi, pada tahap ini nilai yang terbentuk sesuai dengan urutan kelas yang dibuat pada saat inisialisasi data koleksi.

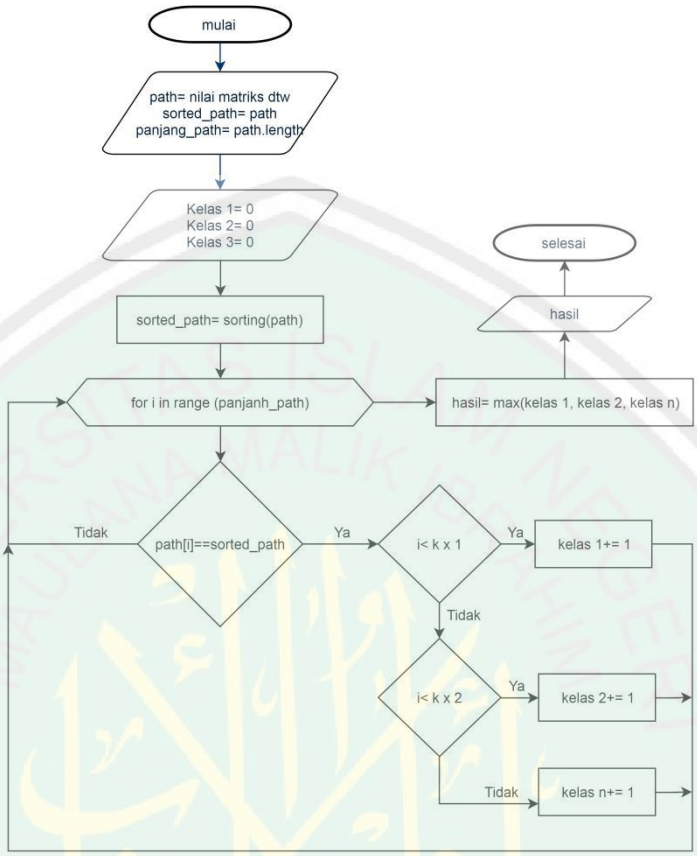
b) Buat array b dengan nilai *sorting* dari *array* a.

c) Tentukan Nilai K (banyak tetangga terdekat).

d) Sesuaikan indeks *array* a dan *array* b dari nilai yang diwakilkan, jika nilai sama dengan indeks data koleksi yang telah dibuat maka tambahkan nilai 1 dari kelas data koleksi.

e) Nilai kelas data koleksi terbesar merupakan hasil dari KNN.

Berikut algoritma KNN yang digambarkan dalam bentuk *flowchart* dalam penelitian ini:



Gambar 3.18 *Flowchart* KNN

Contoh Perhitungan:

$Kelas1 = [[2,1,2,4],[1,2,2,3]]$

$Kelas2 = [[5,3,7,7],[3,5,5,6]]$

$DataUji = [1,1,3,2]$

Kemudian proyeksikan matrik data kelas dengan data uji, dan jumlahkan nilai diagonal sesuai dengan **Gambar 3.18**.

Kelas 1	1	1	3	2		1	1	3	2
2	1	2	3	3	1	0	0	2	3
1	1	1	3	4	2	1	1	1	1
2	2	2	2	2	2	2	2	2	1
4	5	5	3	4	3	4	4	2	2
				8					3
kelas 2	1	1	3	2		1	1	3	2
5	4	8	10	13	3	2	4	4	5
3	6	6	6	7	5	6	6	6	7
7	12	12	10	11	5	10	10	8	9
7	18	18	14	15	6	15	15	11	12
				35					28

Gambar 3.19 Proyeksi Matriks DTW

Lanjut pada tahap KNN:

K= 2 (karena maksimal jumlah data per kelas ada 2)

Hasil DTW= kelas 1, kelas 1, kelas 2, kelas 2= 8, 3, 35, 28

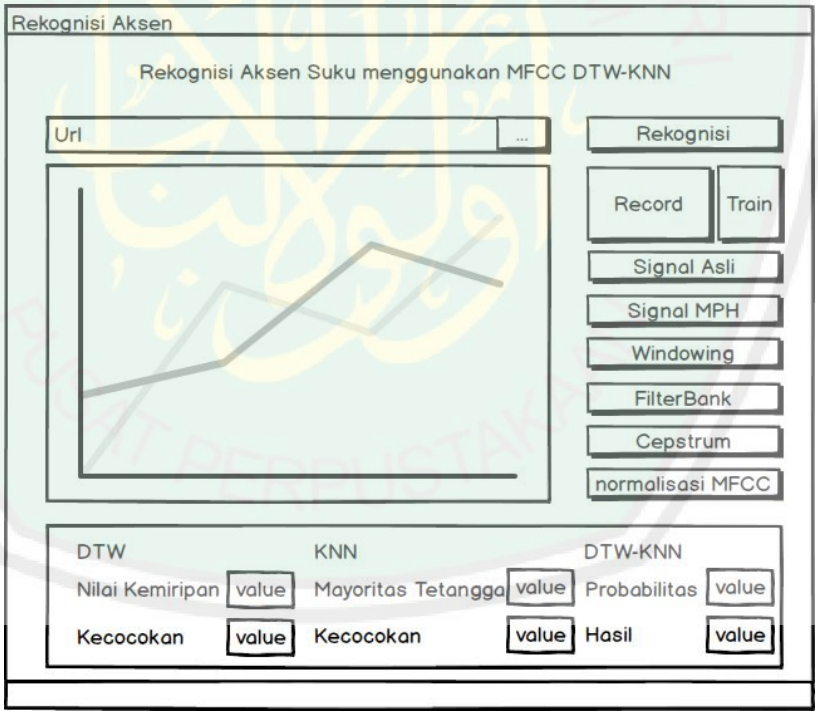
Sorting hasil DTW= 3, 8, 28, 35= kelas 1, kelas 1, kelas 2, kelas 2

Didapatkan hasil Kelas1 sebanyak 2

Probabilitas= 2/2 x 100%= 100%

3.3 Desain Tampilan

Sistem rekognisi aksen akan dibangun berbasis aplikasi *desktop*, khusunya pada sistem opersai *Windows*. Tampilan *user interface* (UI) yang digunakan hanya menggunakan 1 *layout,* yaitu *layout* yang dubuat untuk menangani pengujian terhadap data sampel uji.



Gambar 3.20 Tampilan desain aplikasi

Pada *layout* pengujian disediakan tombol untuk merekam suara ataupun meng*upload* file suara.*wav*, kemudian diproses dengan tombol rekognisi untuk

merekognisi dari data uji yang dimasukan. *Output* hasil rekognisi akan dijabarkan pada tampilan paling bawah. Masing-masing nilai dari metode klasifikasi akan ditampilkan sebagai pembanding. Sebagai pelengkap, ditambahkan tombol masing-masing proses dalam ekstraksi suara MFCC untuk menampilkan gambar grafik/plot/diagram dari sinyal yang didapatkan. Tentunya jika melakukan modifikasi pada data koleksi, telah disediakan tombol latih untuk mengekstraksi seluruh data koleksi dan menyimpannya dalam bentuk variabel *file*. Tombol untuk melakukan perekaman jika ingin dicoba secara langsung. Kemudian pada bagian bawah telah diberikan tempat sebagai hasil analisis rekognisi terhadap data yang ada di dalam *database,* analisis yang disajikan berupa hasil rekognisi menggunakan metode KNN, DTW, dan DTW-KNN dengan masing masing hasil analisis dan nilai kemiripannya.

3.4 Implementasi Sistem

Rancangan sistem yang telah dibuat dan diimplementasikan menggunakan bahasa pemrogramman Python 3.7.1. Untuk pengintegrasian anatara sistem dengan desain antarmuka menggunakan *library* Tkinter yang merupakan perancangan tampilan GUI Desktop. Untuk penyimpanan data koleksi/sampel menggunakan *library* Pickle untuk menyimpan dan membuka kembali pada suatu *file* yang berekstensi .m. Data yang disimpan dalam *file* tersebut berupa nilai angka.

3.4.1 *Import File*

Untuk melakukan import *file audio* untuk diekstraksi diperlukan suatu *library* agar dapat mencari *file audio* yang diinginkan pada *windows explorer*, *library* yang digunakan adalah *filedialog* yang merupakan bagian dari library Tkinter.

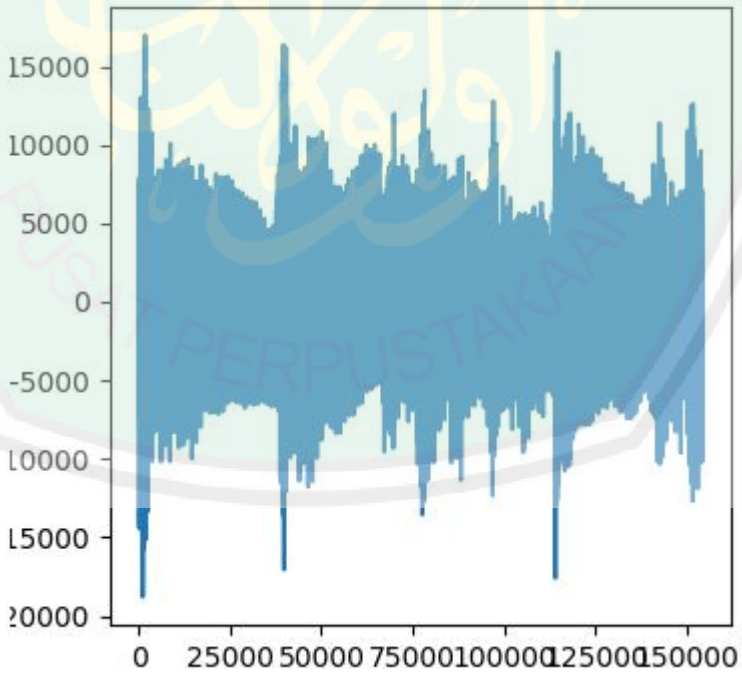
```
def browsefunc():
    global url_info
    filename = filedialog.askopenfilename()
    url_info.config(text= filename)
```

Gambar 3.21 Kode program *import file .wav*

Untuk mengilustrasikan setiap proses pada tahap ekstraksi MFCC, maka akan digunakan file *guitar.wav*. Pada tahap awal sinyal yang masuk akan dipotong dengan batasan 3500ms x *sample rate*. Kemudian wujud sinyal yang masuk dapat dilihat dalam bentuk gelombang *transversal* seperti pada **Gambar 3.23**, sedangkan proses pada tahap ini menggunakan kode program seperti pada **Gambar 3.22**.

```
def preparasi(text):
    global sample_rate, signal
    sample_rate, signal= scipy.io.wavfile.read(text)
    signal= signal[0:int(3.5*sample_rate)]
```

Gambar 3.22 Kode Program proses memotong sinyal



Gambar 3.23 grafik *Input* Sinyal

```
signal
[ -200  -267  -292 ... -8364 -9287 -9559]
```

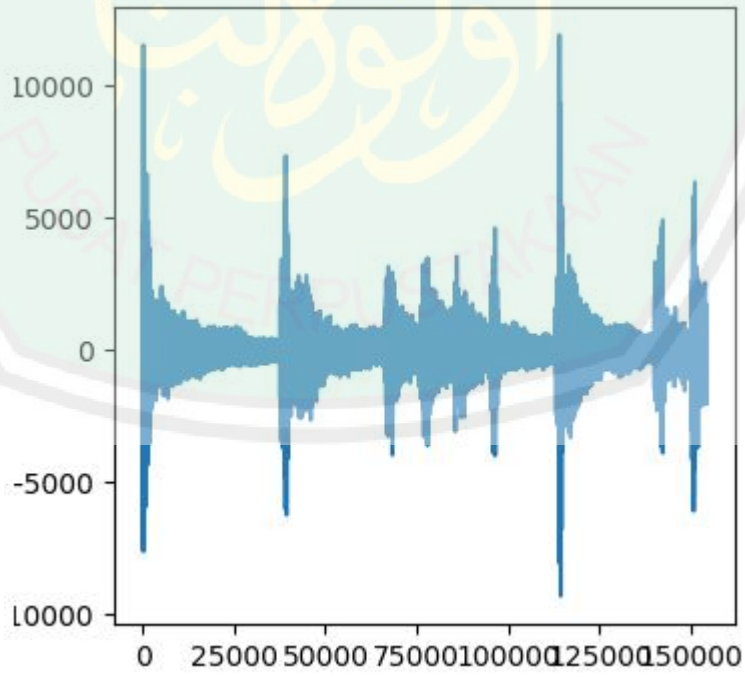
Gambar 3.24 Sinyal yang terbentuk dalam *console Python*

3.4.2 *Pra Emphasis*

Pada proses ini sinyal yang masuk di berikan filter dengan persamaan 3.1 dengan tujuan agar dapat memperrtahankan frekuensi tinggi dan rendah. Koefisien nilai *emphasis* yang digunakan pada program ini sebesar 0.97. Kode profram pada tahap ini dapat dilihat pada **Gambar 3.25**.

```
def pre_emphasised():
    global emphasized_signal
    pre_emphasis= 0.97
    emphasized_signal= []
    for i in range(signal):
        if(i==0):
            emphasized_signal.append(signal[0])
        else:
            signal_mph= signal[i]-0.97*signal[i-1]
            emphasized_signal.append(signal_mph)
```

Gambar 3.25 Kode Progam *Pre Emphasis*



Gambar 3.26 grafik sinyal mph

```
pra emphasized
[ -200.     -73.     -33.01 ... -1617.65 -1173.92  -550.61]
```

Gambar 3.27 sinyal mph yang terbentuk pada *console Python*

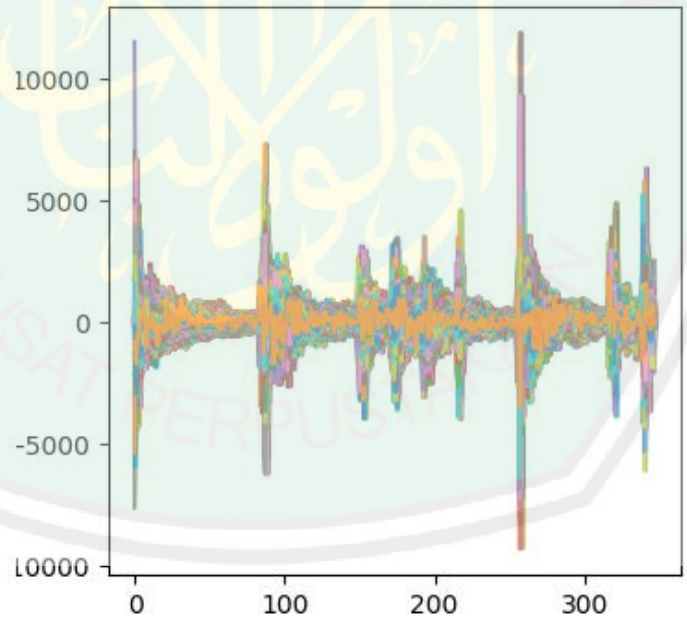
3.4.3 *Framing*

Pada proses Framing sinyal yang masuk akan dipecah-pecah dan dibuat saling tumpang tindih, sesuai dengan *flowchart* pada **Gambar 3.5** beserta teknik perhitungannya dalam melakukan *framing*. Hasil *framing* dapat dilihat dalam *plot* grafik pada **Gambar 3.29** berikut. Terlihat pada gambar tersebut bahwa *plot* garis memiliki warna yang berbeda-beda, hal tersebut menandakan bahwa sinyal sudah dipecah-pecah (*frame*). berbeda dengan gambar yang sebelumnya dimana *line* yang yang menghubungkan setiap titik berwarna biru, hal ini menandakan hanya ada satu sinyal, sedangkan pada tahap *framing* memiliki banyak sinyal dengan panjang sinyal yang sudah terpotong disetiap *frame*nya. **Gambar 3.28** merupakan kode program yang dibangun untuk memecah dan melakukan tumpang tindih pada proses ini.

```
def framing():
    global frames
    global frame_length, num_frames

    frame_size= 0.025
    frame_stride= 0.01

    frame_length, frame_step= frame_size*sample_rate, frame_stride*sample_rate

    signal_length= len(emphasized_signal)
    frame_length=int(round(frame_length))
    frame_step= int(round(frame_step))

    num_frames= int(numpy.ceil(float(numpy.abs(signal_length-frame_length))/frame_step))

    pad_signal_length= num_frames*frame_step+frame_length

    z= numpy.zeros(pad_signal_length-signal_length)
    pad_signal= numpy.append(emphasized_signal, z)

    a= []
    for i in range (len(pad_signal)):
        a.append(i)
```

```
z= (num_frames, frame_length)
indices= numpy.zeros(z)

for i in range(num_frames):
        if(i==0):
                num=0
        else:
                num= indices[i-1][frame_step]

        for j in range(frame_length):
                indices[i][j]= num
                num= num+1

z= (num_frames, frame_length)

b= numpy.zeros(z)

for i in range(num_frames):
        for j in range(frame_length):
                b[i][j]= a[int(indices[i][j])]

frames= pad_signal[b.astype(numpy.int32, copy= False)]
```

Gambar 3.28 Kode program *framing*



Gambar 3.29 Grafik hasil *framing*

```
frames
[[ -200.     -73.     -33.01 ...  1585.46  -167.34  -813.2 ]
 [  988.62  1694.23  2120.12 ... -1372.62 -2925.81 -1696.27]
 [ -659.73 -1208.61 -1670.33 ...  -472.79  -635.97  -532.86]
 ...
 [ -451.04  -746.     957.76 ... -1035.26  -774.59  -248.01]
 [  253.23   177.67   -36.47 ...   177.97   421.54   759.13]
 [ -669.46  -691.42  -547.56 ...  -246.61    31.16   -39.52]]
```

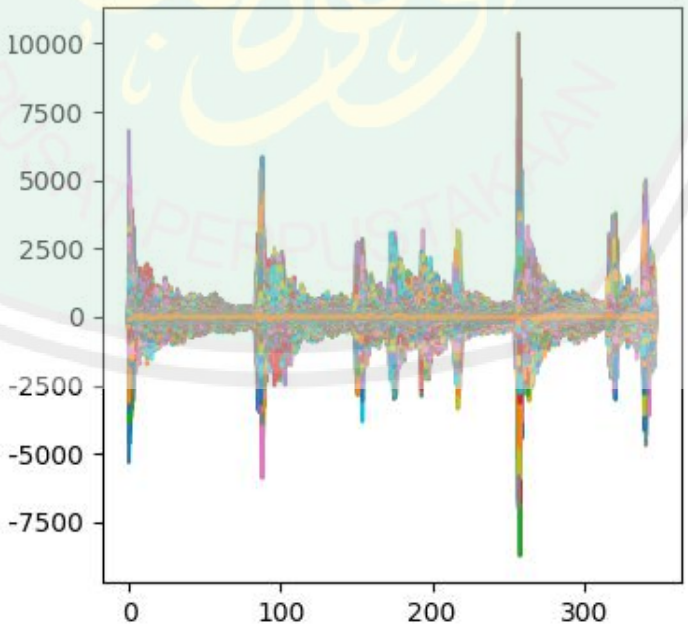
Gambar 3.30 *Output framing* pada *console python*

3.4.4 *Windowing*

Pada proses ini diberikan persamaan *Window Hamming* sesuai dengan persamaan 3.2. Sehingga akan mempengaruhi grafik *frame* sebelumnya, untuk perbedaan hasil dapat dilihat pada **Gambar 3.32** untuk grafik, dan **Gambar 3.33** untuk *output* nilai yang terbentuk pada tahap ini.

```
def windowing():
    global frames, frames_window
    for i in range(num_frames):
        for j in range(frame_length):
            frames[i,j]*=0.54-0.46*numpy.cos((2*numpy.pi*j)/(frame_length-1))
    frames_window= frames
```

Gambar 3.31 Kode program *windowing*



Gambar 3.32 Grafik hasil *windowing*

```
windowing
[[ -16.          -5.84054681   -2.64178904 ...  126.88430327
   -13.38845346  -65.056     ]
 [  79.0896     135.55109066  169.67312266 ... -109.8507262
  -234.08671583 -135.7016    ]
 [ -52.7784     -96.69785311 -133.67644613 ...  -37.83736565
   -50.88236374  -42.6288    ]
 ...
 [ -36.0832     -59.68558793   76.64949623 ...  -82.85181828
   -61.97300208  -19.8408    ]
 [  20.2584      14.21493084   -2.91869271 ...   14.24293231
    33.72635755   60.7304    ]
 [ -53.5568     -55.31877909  -43.82120589 ...  -19.73618888
     2.4930334    -3.1616    ]]
```

Gambar 3.33 *Output windowing* pada *console python*

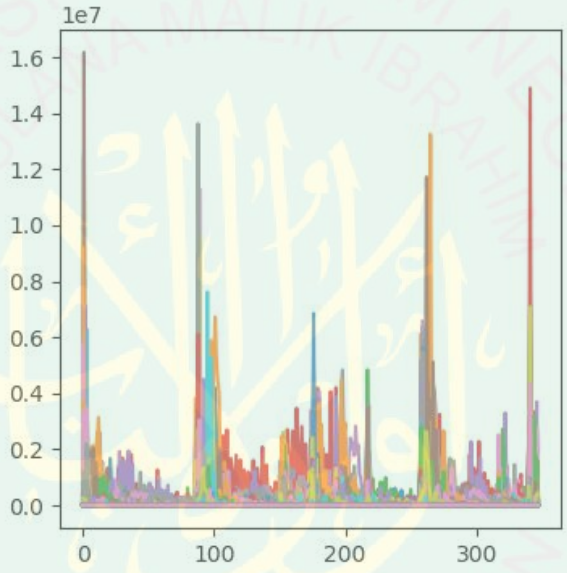
3.4.5 *Power Spectrum*

Pada Power Spektrum digunakan proses FFT (*Fast Forier Transform*), dimana terdapat proses perhitungan menggunakan bilangkan kompleks pada persamaannya (dalam python = 1j), kemudian pada proses nilai *Periodogram/Power Spectrum* (Spektrum daya) digunakan persamaan 3.5 untuk mengubah bilangan kompleks menjadi bilangan desimal.

```
def FFT_PowerSpectrum():
    global NFFT, pow_frames
    NFFT= 512

    nilai_FFT= dataFFT(pow_frames, NFFT)
    mag_frames= magFrames(nilai_FFT, NFFT)
    pow_frames= ((1.0/NFFT)*((mag_frames)**2))

def dataFFT(frames, num_frames):
    fftFinal= []
    for i in range(num_frames):
        fft= FFT(frames[i], num_frames)
        fftFinal.append(fft)

    hasil= fftFinal
    return hasil

def FFT(data, num_frames):
    n= num_frames
    z= numpy.zeros(n, dtype=complex)
    sum= 0
    for k in range(n):
        for j in range(n):
            sum= sum+data[j]*numpy.exp(-2*numpy.pi*1j*(j)*(k)/n)
        z[k]= sum
```

```
            sum= 0
        hasil= z
        print(z)
        return hasil

def magFrames(frames_FFT, NFFT):
        dimensi= (int(len(frames_FFT)), int(math.floor((NFFT/2)+1)))
        data= numpy.zeros(dimensi)

        for i in range(len(frames_FFT)):
                for j in range(int(math.floor((NFFT/2)+1))):
                        data[i][j]= frames_FFT[i][j]

        hasil= data
        return hasil
```

Gambar 3.34 Kode program *power spectrum*



Gambar 3.35 Grafik *power spectrum*

```
power
[[2.23662971e+05 3.47950894e+05 8.95114110e+01 ... 6.94050686e+02
  7.31813294e+02 5.58911745e+02]
 [3.24499111e+04 1.43245352e+05 1.74131370e+05 ... 2.55611740e+03
  2.91651200e+03 2.69507964e+03]
 [4.70894835e+04 2.94284374e+00 9.81890756e+05 ... 3.32019757e+02
  4.42412440e+02 4.14166767e+02]
 ...
 [2.50412433e+04 6.33701341e+04 3.84053193e+05 ... 4.40888599e+02
  3.43312521e+02 9.82156534e+02]
 [6.90886301e+04 1.39757748e+00 8.78979378e+05 ... 3.78402956e+01
  1.09988077e+02 7.22281647e+01]
 [1.82214546e+05 7.54293788e+05 8.66778308e+04 ... 6.03134780e+02
  9.86745408e+02 5.94725742e+02]]
```

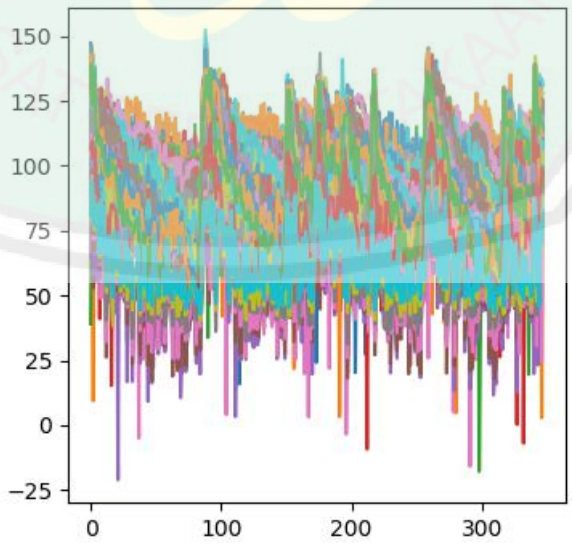
Gambar 3.36 Output spektrum daya pada *console python*

3.4.6 *Filterbank*

Filterbank merupkan *filter* yang dimasukan pada *periodogram* sebelum diproses kedalam nilai *cepstrum*, berikut kode program yang digunakan untuk membuat dan mengubah nilai pada spektrum daya.

```
global filter_banks
    nfilt= 40
    low_freq_mel= 0
    high_freq_mel= (2595*numpy.log10(1+(sample_rate/2)/700))
    mel_points= numpy.linspace(low_freq_mel, high_freq_mel, nfilt+2)

    hz_points= (700*(10**(mel_points/2595)-1)) # convert mel to hz

    tampung= numpy.floor((NFFT+1)*hz_points/sample_rate)

    fbank= numpy.zeros((nfilt, int(numpy.floor(NFFT/2+1))))

    for m in range(1, nfilt+1):
        f_m_minus= int(tampung[m-1]) #left
        f_m= int(tampung[m]) #center
        f_m_plus= int(tampung[m+1]) #right

        for k in range(f_m_minus, f_m):
            fbank[m-1, k]= (k-tampung[m-1])/(tampung[m]-tampung[m-1])
        for k in range(f_m, f_m_plus):
            fbank[m-1, k]= (tampung[m+1]-k)/(tampung[m+1]-tampung[m])

    filter_banks= numpy.dot(pow_frames, fbank.T)
    filter_banks= 20*numpy.log10(filter_banks)
```

Gambar 3.37 Kode program *filterbank*



Gambar 3.38 Grafik *filterbank* setelah diproses dengan spektrum daya

```
filter
[[106.9918818  110.83035913  39.03756806 ...  83.78731916  81.80572941
  81.07613094]
 [ 90.22427022 103.12161079 104.81754036 ...  95.00693008  94.76131946
  94.55848342]
 [ 93.45847854   9.37534404 119.84126343 ...  75.06686905  76.97747362
  80.22091452]
 ...
 [ 87.97311774  96.03769253 111.6878276  ...  76.06520381  76.47848443
  77.63120683]
 [ 96.78813163   2.90751786 118.87957372 ...  68.23825534  67.25266445
  69.83053166]
 [105.21166085 117.55081062  98.75816068 ...  81.43807079  82.27913065
  82.86074239]]
```

Gambar 3.39 *Output filterbank* pada *console python*

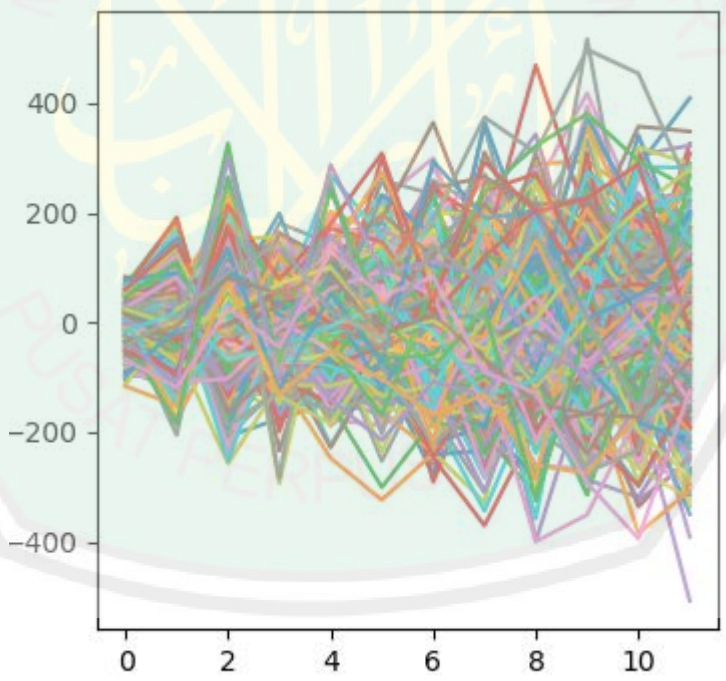
3.4.7 *Cepstrum*

Pada tahap akhir ekstraksi, proses perhitungan cepstrum sesuai dengan *flowchart* pada **Gambar 3.15**. *Cepstrum* atau MFCC merupakan nilai yang akan digunakan untuk proses rekognisi selanjutnya. Dalam perhitungan nilai cepstrum terdapat metode DCT *(Discriete Cosine Transform*) dalam memnuhi perhitungan tersebut. Pada akhir perhitungan dilakukan normalisasi dengan memberikan fungsi mean pada nilai *cepstrum* yang terbentuk.

```
def mfcc():
    global mfcc
    num_ceps= 12
    cep_lifter= 22
    mfcc= nilaiDCT(filter_banks, num_ceps)
    mfcc= numpy.nan_to_num(mfcc)
    panjang_frame_mfcc= len(mfcc[0])
    n= numpy.arange(panjang_frame_mfcc)
    lift= 1+(cep_lifter/2)*numpy.sin(numpy.pi*n/cep_lifter)
    mfcc*= lift
    normalisasi_mfcc()

def nilaiDCT(data, limit):
    banyak_data= len(data)
    banyak_isi= len(data[0])
    isi= (banyak_data, banyak_isi)
    matriks= numpy.zeros(isi)
    for i in range(banyak_data):
        for j in range(banyak_isi):
            sum= 0
            hasil= 0
            for k in range(banyak_isi):
                sum= sum+data[i][k]*math.cos((math.pi*j*(2*k+1))/(2*banyak_isi))
```

```
                if(j==0):
                    hasil= 2*sum*math.sqrt(1/(4*banyak_isi))
                else:
                    hasil= 2*sum*math.sqrt(1/(2*banyak_isi))

                matriks[i][j]= hasil

    hasil= matriks[:, 1:(limit+1)]
    print("hasil mfcc "+str(len(hasil)))
    print("hasil mfcc frame "+str(len(hasil[0])))
    return hasil

def normalisasi_mfcc():
    global mfcc, mfcc_normal, data_ekstraksi
    mfcc_mean= (numpy.mean(mfcc, axis= 0)+1e-8)
    mfcc_normal= mfcc-mfcc_mean
    mfcc_normal= mfcc_normal.T
    print(mfcc_normal)
    data_ekstraksi= []
    for i in range(len(mfcc_normal)):
        for j in range(len(mfcc_normal[0])):
            data_ekstraksi.append(mfcc_normal[i][j])
```

Gambar 3.40 Kode program *cepstrum*



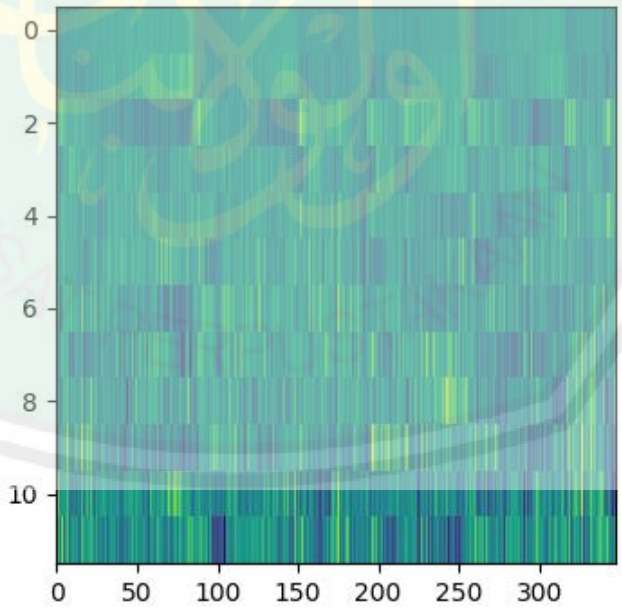
Gambar 3.41 Grafik *cepstrum*

```
mfcc
[[-100.06368239  -90.15459604  -77.93988015 ...  -36.44223448
   -67.30141631  -43.85420257]
 [ -75.55401773  -62.7337796  -133.59834334 ...   -5.75146641
  -118.37559607   12.04135392]
 [ 174.05123802  103.84285863  161.53238489 ...   81.22047811
    27.41924194   92.19972543]
 ...
 [  54.82923097 -104.66949581   42.81038316 ... -166.77759789
  -292.57843445    6.36680465]
 [ 285.03318081  284.36527791   18.04345946 ... -172.37067774
  -394.28819318 -159.84486793]
 [ 409.30546097  173.71916064  -10.09482553 ...  -67.73952537
  -124.63610769   65.91747844]]
```

Gambar 3.42 *Output cepstrum* pada *console python*

3.4.8 *Spektogram*

Spektogram merupakan grafik bunyi yang memberikan informasi tentang perubahan dalam rentang waktu, frekuensi, dan intensitas gelombang bunyi menurut sumbu waktu, berikut gambar dari spektogram yang terbentuk dari *file audio wav* tersebut:



Gambar 3.43 Spektogram cepstrum

3.4.9 *KNN*

Pada metode KNN atau *K-Nearest Neighbour*, contoh perhitungan telah dijelaskan pada subab sebelumnya. Dalam implementasi KNN pada penelitian ini data sampel yang digunakan terdiri dari laki-laki dan perempuan sehingga kedua data tersebuat harus dibuat terpisah agar terjadi keseimbangan nilai antara *gender* tersebut. Misal suatu data pelafal suara adalah seorang laki laki, pada proses input sinyal ternyata suara laki laki tersebut memiliki sinyal yang mendekati sinyal suara perempuan, sehingga menyebabkan suara perempuan lebih banyak daripada suara laki laki. Oleh karena itu, untuk menghindari hal tersebut, sampel data laki laki dan perempuan dipisah dan dalam perhitungan KNN akan diambil K/2+1 tetangga terdekat dalam sampel data laki laki dan sampel data perempuan dalam setiap sukunya, kemudian kembali ke perhitungan KNN yang sebenarnya dimana kedua sampel tersebut digabungkan dan diurutkan kemudian diambil K tetangganya dengan indeks yang mereferensikan kelas suku dari sampel tersebut. Kelas dengan sampel terbanyaklah output dari metode ini.

```
def knn_recog():
    global data_ekstraksi, nilai_hasil_knn, nilai_kemiripan_knn, kemiripan_knn, suku
    mfcc_data_laki_laki=[]
    mfcc_data_perempuan=[]

    with open('hasil_ekstraksi_mfcc_laki_laki', 'rb') as data1:
        mfcc_data_laki_laki = pickle.load(data1)
    with open('hasil_ekstraksi_mfcc_perempuan', 'rb') as data2:
        mfcc_data_perempuan = pickle.load(data2)

    nilai= []
    for i in range(len(mfcc_data_laki_laki)):
        hasil= numpy.zeros(len(mfcc_data_laki_laki[0]))
        for j in range(len(mfcc_data_laki_laki[0])):
            hasil[j]= (mfcc_data_laki_laki[i][j])-(data_ekstraksi[j])
        nilai.append(hasil)

    for i in range(len(nilai)):
        for j in range(len(nilai[0])):
```

```
            nilai[i][j]= nilai[i][j]**2

distance= numpy.zeros(len(nilai))
distance= numpy.sum(nilai, axis= 1)

#BubbleSort
distancesort= list(distance)
for i in range(len(distancesort)):
    for j in range(len(distancesort)-1):
        if(distancesort[j]>distancesort[j+1]):
            swap= distancesort[j]
            distancesort[j]= distancesort[j+1]
            distancesort[j+1]= swap

k= numpy.zeros(4)
print(distance)
print(distancesort)

kelas1= 0
kelas2= 0
kelas3= 0
kelas4= 0

for i in range(len(k)):
    for j in range(len(distance)):
        if(distancesort[i]==distance[j]):
            if(j<5):
                kelas1+=1
                break
            elif(j<10):
                kelas2+=1
                break
            elif(j<15):
                kelas3+=1
                break
            elif(j<20):
                kelas4+=1
                break

print("kelas 1= "+str(kelas1))
print("kelas 2= "+str(kelas2))
print("kelas 3= "+str(kelas3))
print("kelas 4= "+str(kelas4))

nilai2= []
for i in range(len(mfcc_data_perempuan)):
    hasil2= numpy.zeros(len(mfcc_data_perempuan[0]))
    for j in range(len(mfcc_data_perempuan[0])):
        hasil2[j]= (mfcc_data_perempuan[i][j])-(data_ekstraksi[j])
    nilai2.append(hasil2)

for i in range(len(nilai2)):
    for j in range(len(nilai2[0])):
        nilai2[i][j]= nilai2[i][j]**2

distance2= numpy.zeros(len(nilai2))
distance2= numpy.sum(nilai2, axis= 1)
#BubbleSort
```

```
distancesort2= list(distance2)
for i in range(len(distancesort2)):
    for j in range(len(distancesort2)-1):
        if(distancesort2[j]>distancesort2[j+1]):
            swap= distancesort2[j]
            distancesort2[j]= distancesort2[j+1]
            distancesort2[j+1]= swap

k2= numpy.zeros(4)
print(distance2)
print(distancesort2)

for i in range(len(k2)):
    for j in range(len(distance)):
        if(distancesort2[i]==distance2[j]):
            if(j<5):
                kelas1+=1
                break
            elif(j<10):
                kelas2+=1
                break
            elif(j<15):
                kelas3+=1
                break
            elif(j<20):
                kelas4+=1
                break

print("kelas 1= "+str(kelas1))
print("kelas 2= "+str(kelas2))
print("kelas 3= "+str(kelas3))
print("kelas 4= "+str(kelas4))

suku= ''
skor= max(kelas1, kelas2, kelas3, kelas4)
if(kelas1==skor):
    suku= 'Jawa'
elif (kelas2==skor):
    suku= 'Kaili'
elif(kelas3==skor):
    suku= 'Madura'
else:
    suku= 'Sunda'

nilai_hasil_knn.config(text=suku)

Kemiripan_knn= skor/8*100
str_kemiripan= str(kemiripan_knn)+'%'
nilai_kemiripan_knn.config(text= str_kemiripan)
```

Gambar 3.44 Kode program KNN

3.4.10 *DTW*

DTW atau *Dynamic Time Warping* juga dalam penelitian ini tidak menggunakan perbedaan gender dalam klasifikasinya. DTW akan langsung memproyeksikan matriks sebesar 1 x panjang data koleksi dengan 1 x panjang data uji. Untuk memenuhi setiap elemen matriksnya digunakan persamaan 3.11 untuk memnuhinya, sehingga menyebabkan proses komputasi yang begitu lama jika input yang dimasukkan tergolong besar. Setelah matriks yang dibuat selesai maka hal yang dilakukan selanjutnya menyeleksi nilai minimal secara diagonal dan mengakumulasikannya hingga menjadi nilai yang tunggal. Jika semua data koleksi hasil ekstrasi sudah selesai diproses dengan data uji, maka nilai terkecil dari kumpulan nilai tersebut yang merupakan output dari metode klasifikasi ini.

```
def dtw_recog():
    global data_ekstraksi, nilai_hasil_dtw, nilai_kemiripan_dtw, data_perbandingan
    global data_perbandingan2, skoring_dtw, suku_

    with open('hasil_ekstraksi_mfcc_laki_laki', 'rb') as data1:
        mfcc_data = pickle.load(data1)

    with open('hasil_ekstraksi_mfcc_perempuan', 'rb') as data2:
        mfcc_data_perempuan = pickle.load(data2)

    data_latih= mfcc_data
    data_uji= data_ekstraksi

    panjang_data_latih= len(data_latih[0])
    banyak_data_latih= len(data_latih)
    panjang_data_uji= len(data_uji)

    data_perbandingan=[]
    for i in range(banyak_data_latih):
        ordo= (panjang_data_latih, panjang_data_uji)
        tampung= numpy.zeros(ordo)

        #tahap 1
        for j in range(panjang_data_uji):
            if(j==0):
                tampung[0, j]= numpy.absolute(data_latih[i][0]-data_uji[j])
            else:
                tampung[0, j]= numpy.absolute(data_latih[i][0]-data_uji[j])+tampung[0, j-1]

        #tahap 2
        for j in range(panjang_data_latih):
```

```
                if(j>0):
                    tampung[j, 0]= numpy.absolute(data_latih[i][j]-data_uji[0])+tampung[j-1, 0]

            #tahap 3
            for j in range(panjang_data_latih):
                for k in range(panjang_data_uji):
                    if(j>0 and k>0):
                        tampung[j][k]=
numpy.absolute(data_latih[i][j]-data_uji[k])+nilai_minimum(tampung[j-1][k], tampung[j][k-1],
tampung[j-1][k-1])

            m= panjang_data_latih-1
            n= panjang_data_uji-1
            jarak = tampung[m][n]
            while(m>0 or n>0):
                jarak_terdekat= 0
                if(m==0):
                    jarak_terdekat= tampung[m][n-1]
                    n=n-1
                elif(n==0):
                    jarak_terdekat= tampung[m-1][n]
                    m=m-1
                else:
                    nilai_diagonal=tampung[m-1][n-1]
                    nilai_kanan= tampung[m-1][n]
                    nilai_kiri= tampung[m][n-1]
                    if(nilai_diagonal<=nilai_kanan and nilai_diagonal<=nilai_kiri):
                        jarak_terdekat= nilai_diagonal
                        m=m-1
                        n=n-1
                    elif(nilai_kanan<nilai_diagonal and nilai_kanan<=nilai_kiri):
                        jarak_terdekat= nilai_kanan
                        m= m-1
                    elif(nilai_kiri<nilai_diagonal and nilai_kiri<=nilai_kanan):
                        jarak_terdekat= nilai_kiri
                        n=n-1

                #print("jarak_terdekat "+str(jarak_terdekat))
                jarak= jarak+jarak_terdekat
                #print("jarak "+str(jarak))
            data_perbandingan.append(jarak)
        print("hilih")
        print(data_perbandingan)

#=============================================================================#
        data_latih_perempuan= mfcc_data_perempuan
        data_uji= data_ekstraksi

        panjang_data_latih= len(data_latih_perempuan[0])
        banyak_data_latih= len(data_latih_perempuan)
        panjang_data_uji= len(data_uji)

        data_perbandingan2=[]
        for i in range(banyak_data_latih):
            print("loop ke- "+str(i))
            ordo= (panjang_data_latih, panjang_data_uji)
            tampung= numpy.zeros(ordo)
```

```
            #tahap 1
            for j in range(panjang_data_uji):
                if(j==0):
                    tampung[0, j]= numpy.absolute(data_latih_perempuan[i][0]-data_uji[j])
                else:
                    tampung[0, j]=
numpy.absolute(data_latih_perempuan[i][0]-data_uji[j])+tampung[0, j-1]

            #tahap 2
            for j in range(panjang_data_latih):
                if(j>0):
                    tampung[j, 0]=
numpy.absolute(data_latih_perempuan[i][j]-data_uji[0])+tampung[j-1, 0]

            #tahap 3
            for j in range(panjang_data_latih):
                for k in range(panjang_data_uji):
                    if(j>0 and k>0):
                        tampung[j][k]=
numpy.absolute(data_latih_perempuan[i][j]-data_uji[k])+nilai_minimum(tampung[j-1][k],
tampung[j][k-1], tampung[j-1][k-1])

            m= panjang_data_latih-1
            n= panjang_data_uji-1
            jarak = tampung[m][n]
            while(m>0 or n>0):
                jarak_terdekat= 0
                if(m==0):
                    jarak_terdekat= tampung[m][n-1]
                    n=n-1
                elif(n==0):
                    jarak_terdekat= tampung[m-1][n]
                    m=m-1
                else:
                    nilai_diagonal=tampung[m-1][n-1]
                    nilai_kanan= tampung[m-1][n]
                    nilai_kiri= tampung[m][n-1]
                    if(nilai_diagonal<=nilai_kanan and nilai_diagonal<=nilai_kiri):
                        jarak_terdekat= nilai_diagonal
                        m=m-1
                        n=n-1
                    elif(nilai_kanan<nilai_diagonal and nilai_kanan<=nilai_kiri):
                        jarak_terdekat= nilai_kanan
                        m= m-1
                    elif(nilai_kiri<nilai_diagonal and nilai_kiri<=nilai_kanan):
                        jarak_terdekat= nilai_kiri
                        n=n-1

                jarak= jarak+jarak_terdekat
            data_perbandingan2.append(jarak)

        panjang_data_latih= 20

        skor_laki= min(data_perbandingan)
        skor_perempuan= min(data_perbandingan2)
        skoring_dtw

        indeks= 0
```

```
if(skor_laki<skor_perempuan):
    skoring_dtw= skor_laki
    nilai_kemiripan_dtw.config(text= str(skor_laki))
    for i in range(panjang_data_latih):
        if(skor_laki==data_perbandingan[i]):
            indeks= i
elif(skor_perempuan<skor_laki):
    skoring_dtw= skor_perempuan
    nilai_kemiripan_dtw.config(text= str(skor_perempuan))
    for i in range(panjang_data_latih):
        if(skor_perempuan==data_perbandingan2[i]):
            indeks= i
suku_ = ''
if(indeks<5 ):
    suku_ = 'Jawa'
elif(indeks<10 ):
    suku_ = 'Kaili'
elif(indeks<15):
    suku_ = 'Madura'
elif(indeks<20 ):
    suku_ = 'Sunda'

nilai_hasil_dtw.config(text= suku_)
```

Gambar 3.45 Kode program DTW

3.4.11 *DTW-KNN*

Pada metode DTW-KNN perlu melakukan pembedaan gender pada prosesnya seperti metode KNN sebelumnya. Perbedaannya adalah proses KNN yang awalnya menggunakan *Euclidean Distance* diubah dengan metode DTW. Pada metode ini, penulis juga menambahkan *handling* dari metode sebelumnya yaitu KNN dan DTW, jadi ketika metode DTW-KNN ingin mengambil hasil, metode tersebut akan mempertimbangkan hasil dari metode sebelumnya dengan metode DTW-KNN itu sendiri. Petimbangan tersebut, yaitu:

1. Diasumsikan bahwa telah mendapatkan tiga nilai berupa bilangan hasil pada metode DTW dan nilai presentase probabilitas dari metode KNN dan DTW-KNN.
2. Jika nilai DTW adalah 0, maka nilai hasil dari DTW-KNN adalah nilai DTW.

3. Jika nilai hasil DTW bukan 0 dan nilai antara KNN dan DTW-KNN sama, walaupun kelas sama ataupun berbeda makan nilai DTW-KNN merupakan nilai dari DTW-KNN itu sendiri.
4. Jika nilai DTW bukan 0 dan KNN ataupun DTW-KNN berada dibawah 60% maka jawaban adalah DTW-KNN adalah DTW.
5. Jika nilai DTW bukan 0 dan nilai KNN berada diatas 60% dan lebih tinggi dari nilai DTW-KNN maka jawaban DTW-KNN adalah hasil dari KNN.
6. Jika nilai DTW bukan 0 dan DTW-KNN berada diatas 60% dan lebih tinggi dari nilai KNN maka jawaban adalah nilai asli dari DTW-KNN.

```
def dtw_knn_recog():
    global data_perbandingan, data_perbandingan2, nilai_hasil_dtw_knn
    global    nilai_kemiripan_dtw_knn, kemiripan_knn, skoring_dtw

    mfcc_data_laki_laki= data_perbandingan
    mfcc_data_perempuan= data_perbandingan2

    #BubbleSort
    distancesort= list(mfcc_data_laki_laki)
    for i in range(len(distancesort)):
        for j in range(len(distancesort)-1):
            if(distancesort[j]>distancesort[j+1]):
                swap= distancesort[j]
                distancesort[j]= distancesort[j+1]
                distancesort[j+1]= swap

    k= numpy.zeros(4)
    print(mfcc_data_laki_laki)
    print(distancesort)

    kelas1= 0
    kelas2= 0
    kelas3= 0
    kelas4= 0

    for i in range(len(k)):
        for j in range(len(mfcc_data_laki_laki)):
            if(distancesort[i]==mfcc_data_laki_laki[j]):
                if(j<5):
                    kelas1+=1
                    break
                elif(j<10):
                    kelas2+=1
                    break
                elif(j<15):
                    kelas3+=1
```

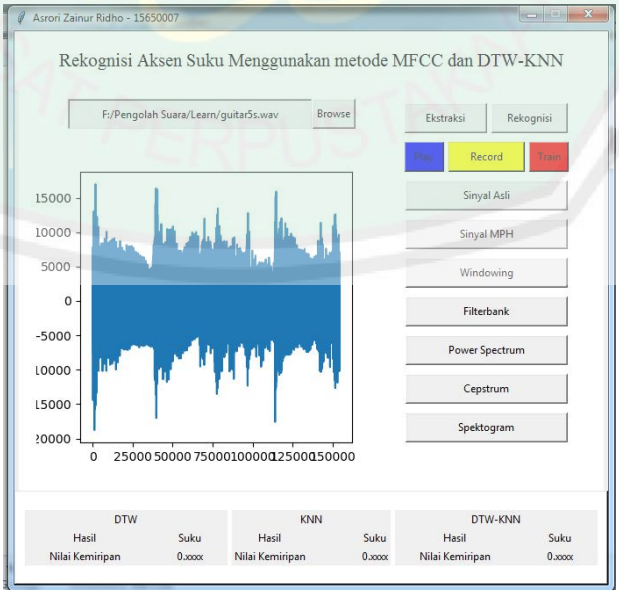
```
                    break
                elif(j<20):
                    kelas4+=1
                    break

print("kelas 1= "+str(kelas1))
print("kelas 2= "+str(kelas2))
print("kelas 3= "+str(kelas3))
print("kelas 4= "+str(kelas4))

#========================== perempuan========== ==========--#

#BubbleSort
distancesort2= list(mfcc_data_perempuan)
for i in range(len(distancesort2)):
    for j in range(len(distancesort2)-1):
        if(distancesort2[j]>distancesort2[j+1]):
            swap= distancesort2[j]
            distancesort2[j]= distancesort2[j+1]
            distancesort2[j+1]= swap

k2= numpy.zeros(4)
print(mfcc_data_perempuan)
print(distancesort2)

for i in range(len(k2)):
    for j in range(len(mfcc_data_perempuan)):
        if(distancesort2[i]==mfcc_data_perempuan[j]):
            if(j<5):
                kelas1+=1
                break
            elif(j<10):
                kelas2+=1
                break
            elif(j<15):
                kelas3+=1
                break
            elif(j<20):
                kelas4+=1
                break

print("kelas 1= "+str(kelas1))
print("kelas 2= "+str(kelas2))
print("kelas 3= "+str(kelas3))
print("kelas 4= "+str(kelas4))

suku= ''
skor= max(kelas1, kelas2, kelas3, kelas4)
if(kelas1==skor):
    suku= 'Jawa'
elif (kelas2==skor):
    suku= 'Kaili'
elif(kelas3==skor):
    suku= 'Madura'
else:
    suku= 'Sunda'

nilai_hasil_dtw_knn.config(text=suku)
```

```
kemiripan= skor/8*100
#Handling Modified

if(skoring_dtw<1):
    kemiripan= skoring_dtw
    str_kemiripan= str(kemiripan)+'%'
    nilai_kemiripan_dtw_knn.config(text= str_kemiripan)
elif(skoring_dtw>1 & kemiripan==kemiripan_knn):
    str_kemiripan= str(kemiripan)+'%'
    nilai_kemiripan_dtw_knn.config(text= str_kemiripan)
elif(skoring_dtw>1 & kemiripan<60 & kemiripan_knn<60):
    kemiripan= skoring_dtw
    str_kemiripan= str(kemiripan)+'%'
    nilai_kemiripan_dtw_knn.config(text= str_kemiripan)
elif(skoring_dtw>1 & kemiripan>=60):
    str_kemiripan= str(kemiripan)+'%'
    nilai_kemiripan_dtw_knn.config(text= str_kemiripan)
elif(skoring_dtw>1 & kemiripan_knn>=60):
    kemiripan= kemiripan_knn
    str_kemiripan= str(kemiripan)+'%'
        nilai_kemiripan_dtw_knn.config(text= str_kemiripan)
```

Gambar 3.46 Kode program DTW-KNN

3.5 Implementasi Desain Antarmuka

Pada aplikasi ini, desain antarmuka dibangun menggunakan *library* tkinter dalam lingkungan program python versi 3.7.1. Tkinter merupakan *library* yang memungkinkan untuk membuat GUI Desktop. Untuk lebih jelasnya hasil dari desain antarmuka dapat dilihat pada **Gambar 3.47.**



Gambar 3.47 Desain antarmuka aplikasi

Pada desain tersebut telah disediakan tombol browse yang berfungsi untuk mencari file audio yang akan direkognisi dan disamping tombol browse terdapat url dari file yang kita cari. Pada bagian kanan GUI terdapat beberapa tombol. Tombol play berfungsi memainkan audio yang sudah dipilih melalui tombol browse, tombol record untuk merekan suara secara langsung, dan tombol train untuk memulai ekstraksi data sampel pada direktori yang telah disediakan. Untuk melakukan ekstraksi dapat langsung menekan tombol ekstraksi. Menekan tombol ekstraksi otomatis akan mengaktifkan tombol sinyal asli hingga tombol spektogram, sehingga pengguna dapat melihat bentuk grafik dari audio yang telah dimasukan. Untuk melakukan rekognisi dapat dilakukan dengan menekan tombol rekognisi. Pada bagian kiri terdapat gambar grafik yang secara dinamis akan menampilkan model grafik berdasarkan langkah proses yang ditekan pada tombol disamping kanan. Terakhir pada bagian bawah, terdapapa hasil analisis rekognisi dari file audio yang diinputkan.

BAB IV

HASIL DAN PEMBAHASAN

Bab ini membahas tentang uji coba sistem yang telah dirancang dan dibangun. Uji coba dilakukan untuk mengetahui keberhasilan sistem dalam menjalankan algoritma yang telah dibuat dalam skenario pengujian. Uji coba pada sistem ini juga untuk mengetahui akurasi dari hasil rekognisi terhadap data uji yang dimasukan.

4.1 Skenario Pengujian

Terdapat 40 data koleksi di dalam database untuk memproses pengujian terhadap data uji yang diinputkan. Setiap kelas akan dicoba sebanyak 8 kali percobaan dimana menggunakan 4 data yang ada didalam data koleksi dan 4 data yang tidak ada di dalam data koleksi. 4 data tersebut dibagi lagi menjadi 2 data, yaitu laki laki dan 2 data perempuan. Banyaknya pengujian setiap metode sebanyak 32 pengujian, sehingga jika data-data tersebut diuji dengan 3 metode, maka proses pengujian berjumlah 96 kali pengujian. Langkah pertama yang dilakukan yaitu dengan mengimport suara ke dalam sistem. Kemudian diikuti proses ekstraksi dan rekognisi terhadap metode yang telah dibuat. Hasil yang didapatkan setelah pengujian berupa hasil rekognisi kelas (prediksi suku) dan nilai ukur kemiripan yang berupa persentase untuk metode KNN dan DTW-KNN serta nilai desimal untuk metode DTW dimana semakin kecil nilai maka akan semakin mirip. Kemudian setiap hasil rekognisi akan direkap dan diukur nilai akurasi berdasarkan metode klasifikasi masing-masing, kemudian menentukan nilai recall dan presisi terhadap kelas/suku dari masing-masing hasil pengujian.

Hasil akurasi, *recall,* dan *presisi* dari setiap metode akan diukur menggunakan metode ROC (*Receiver Operating Charateristic)*. Dalam ROC terdapat 4 komponen yang perlu diketahui, yaitu:

1. *True Positive* : Jumlah data positif yang terklasifikasi dengan benar oleh sistem.
2. *True Negative* : Jumlah data negatif yang terklasifikasi dengan benar oleh sistem.
3. *False Positive* : Jumlah data negatif namun terklasifikasi salah oleh sistem.
4. *False Negative* : Jumlah data positif namun terklasifikasi salah oleh sistem.

Tabel 4.1 Klasifikasi klaster data hasil ROC

Kelas	*True*	*Negative*
True	*True Positive (TP)*	*False Negative (FN)*
Negative	*False Positif (FP)*	*True Negative (TN)*

Akurasi didefiniskan sebagai tingkat kedekatan anatara nilai prediksi dengan nilai yang sebenarnya. Persamaan akurasi didefiniskan sebagai berikut:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} x100\% \quad (4.1)$$

Recall adalah tingkat keberhasilan sistem dalam menemukan kembali informasi dalam sebuah sistem. Persamaan Recall didefinisikan sebagai berikut:

$$recall = \frac{TP}{TP + FN} x100\% \quad (4.2)$$

Presision adalah tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem. Persamaan Precision didefinsikan sebagai berikut:

$$precision = \frac{TP}{TP + FP} x100\% \quad (4.3)$$

Proses perancangan, pembangunan, dan uji coba sistem dilakukan pada *hardware* dan *software* sebagai berikut:

1. Hardware
    - Processor Intel Core i5-2410M, 2.3 GHz
    - 6 GB RAM
    - VGA Intel HD Graphic Family
    - Hard Drive 500 GB
2. Software
    - Sistem Operasi Windows 7 64-bit
    - Visual Studio Code versi 1.32.3
    - Audacity versi 2.2.2

4.2 Hasil Pengujian

Pengujian dalam penelitian mengukur akurasi dari hasil rekognisi setiap data uji yang dimasukan. Berikut daftar perekaman *audio* yang dijadikan sebagai data koleksi dan data uji:

Tabel 4.2 Daftar data koleksi

No	Nama File Audio
1	Jawa laki laki 1
2	Jawa laki laki 2
3	Jawa laki laki 3
4	Jawa laki laki 4
5	Jawa laki laki 5
6	Jawa laki laki 6
7	Jawa laki laki 7
8	Jawa perempuan 1
9	Jawa perempuan 2
10	Jawa perempuan 3
11	Jawa perempuan 4
12	Jawa perempuan 5
13	Jawa perempuan 6
14	Jawa perempuan 7
15	Kaili Laki laki 1
16	Kaili Laki laki 2
17	Kaili Laki laki 3
18	Kaili Laki laki 4
19	Kaili Laki laki 5
20	Kaili Laki laki 6

No	Nama File Audio
21	Kaili Laki laki 7
22	Kaili Perempuan 1
23	Kaili Perempuan 2
24	Kaili Perempuan 3
25	Kaili Perempuan 4
26	Kaili Perempuan 5
27	Kaili Perempuan 6
28	Kaili Perempuan 7
29	Madura Laki laki 1
30	Madura Laki laki 2
31	Madura Laki laki 3
32	Madura Laki laki 4
33	Madura Laki laki 5
34	Madura Laki laki 6
35	Madura Laki laki 7
36	Madura Perempuan 1
37	Madura Perempuan 2
38	Madura Perempuan 3
39	Madura Perempuan 4
40	Madura Perempuan 5

No	Nama File Audio
41	Madura Perempuan 6
42	Madura Perempuan 7
43	Sunda Laki-laki 1
44	Sunda Laki-laki 2
45	Sunda Laki-laki 3
46	Sunda Laki-laki 4
47	Sunda Laki-laki 5
48	Sunda Laki-laki 6

No	Nama File Audio
49	Sunda Laki-laki 7
50	Sunda Perempuan 1
51	Sunda Perempuan 2
52	Sunda Perempuan 3
53	Sunda Perempuan 4
54	Sunda Perempuan 5
55	Sunda Perempuan 6
56	Sunda Perempuan 7

Tabel 4.3 Daftar pengujian *file audio*

No	Nama File Audio
1	Koleksi Jawa laki laki 1
2	Koleksi Jawa laki-laki 2
3	Koleksi Jawa perempuan 1
4	Koleksi Jawa perempuan 2
5	Uji Jawa laki laki 1
6	Uji Jawa laki laki 2
7	Uji Jawa perempuan 1
8	Uji Jawa perempuan 2
9	Koleksi Kaili laki laki 1
10	Koleksi Kaili laki-laki 2
11	Koleksi Kaili perempuan 1

No	Nama File Audio
12	Koleksi Kaili perempuan 2
13	Uji Kaili laki laki 1
14	Uji Kaili laki laki 2
15	Uji Kaili perempuan 1
16	Uji Kaili perempuan 2
17	Koleksi Madura laki laki 1
18	Koleksi Madura laki-laki 2
19	Koleksi Madura perempuan 1
20	Koleksi Madura perempuan 2
21	Uji Madura laki laki 1
22	Uji Madura laki laki 2

No	Nama File Audio
23	Uji Madura perempuan 1
24	Uji Madura perempuan 2
25	Koleksi Sunda laki laki 1
26	Koleksi Sunda laki-laki 2
27	Koleksi Sunda perempuan 1

No	Nama File Audio
28	Koleksi Sunda perempuan 2
29	Uji Sunda laki laki 1
30	Uji Sunda laki laki 2
31	Uji Sunda perempuan 1
32	Uji Sunda perempuan 2

Data yang sudah diinputkan akan diekstraksi dan dilakukan rekognisi terhadap masing masing metode klasifikasi, berikut hasil dari rekognisi masing masing metode klasifikasi:

1. Hasil rekognisi dengan KNN

Hasil rekognisi dari pengujian data terhadap metode KNN dilampirkan pada **Tabel 4.4** berikut:

Tabel 4.4 Hasil rekognisi menggunakan KNN

No	Pengujian	Target	Prediksi	Nilai Prediksi
1	Jawa laki-laki 1	Jawa	Jawa	75.00%
2	Jawa laki-laki 2	Jawa	Jawa	75.00%
3	Jawa perempuan 1	Jawa	Jawa	62.50%
4	Jawa perempuan 2	Jawa	Sunda	50.00%
5	Uji Jawa laki-laki 1	Jawa	Sunda	62.50%
6	Uji Jawa laki-laki 2	Jawa	Jawa	75.00%
7	Uji Jawa perempuan 1	Jawa	Jawa	50%
8	Uji Jawa perempuan 2	Jawa	Jawa	37.50%

9	Kaili laki-laki 1	Kaili	Kaili	37.50%
10	Kaili laki-laki 2	Kaili	Sunda	38%
11	Kaili perempuan 1	Kaili	Jawa	50.00%
12	Kaili perempuan 2	Kaili	Kaili	50.00%
13	Uji Kaili laki-laki 1	Kaili	Kaili	37.50%
14	Uji Kaili laki-laki 2	Kaili	Kaili	37.50%
15	Uji Kaili perempuan 1	Kaili	Jawa	38%
16	Uji Kaili perempuan 2	Kaili	Sunda	62.50%
17	Madura laki-laki 1	Madura	Sunda	37.50%
18	Madura laki-laki 2	Madura	Jawa	50%
19	Madura perempuan 1	Madura	Kaili	37.50%
20	Madura perempuan 2	Madura	Kaili	37.50%
21	Uji Madura laki-laki 1	Madura	Kaili	37.50%
22	Uji Madura laki-laki 2	Madura	Kaili	37.50%
23	Uji Madura perempuan 1	Madura	Kaili	38%
24	Uji Madura perempuan 2	Madura	Jawa	37.50%
25	Sunda laki-laki 1	Sunda	Sunda	63%
26	Sunda laki-laki 2	Sunda	Sunda	63%
27	Sunda perempuan 1	Sunda	Sunda	50%
28	Sunda perempuan 2	Sunda	Sunda	63%
29	Uji Sunda laki-laki 1	Sunda	Kaili	38%
30	Uji Sunda laki-laki 2	Sunda	Kaili	50%
31	Uji Sunda perempuan 1	Sunda	Sunda	63%

32	Uji Sunda perempuan 2	Sunda	Sunda	50.00%

2. Hasil rekognisi dengn DTW

Hasil rekognisi dari pengujian data terhadap metode DTW dilampirkan pada **Tabel 4.5** berikut:

Tabel 4.5 Hasil rekognisi menggunakan DTW

No	Pengujian	Target	Prediksi	Nilai Prediksi
1	Jawa laki-laki 1	Jawa	Jawa	0
2	Jawa laki-laki 2	Jawa	Jawa	0
3	Jawa perempuan 1	Jawa	Jawa	0
4	Jawa perempuan 2	Jawa	Jawa	0
5	Uji Jawa laki-laki 1	Jawa	Kaili	770910121
6	Uji Jawa laki-laki 2	Jawa	Jawa	576080721
7	Uji Jawa perempuan 1	Jawa	Jawa	584669838
8	Uji Jawa perempuan 2	Jawa	Jawa	788155239
9	Kaili laki-laki 1	Kaili	Kaili	0
10	Kaili laki-laki 2	Kaili	Kaili	0
11	Kaili perempuan 1	Kaili	Kaili	0
12	Kaili perempuan 2	Kaili	Kaili	0
13	Uji Kaili laki-laki 1	Kaili	Kaili	772491163
14	Uji Kaili laki-laki 2	Kaili	Madura	777133627
15	Uji Kaili perempuan 1	Kaili	Madura	718273709
16	Uji Kaili perempuan 2	Kaili	Kaili	729148722

17	Madura laki-laki 1	Madura	Madura	0
18	Madura laki-laki 2	Madura	Madura	0
19	Madura perempuan 1	Madura	Madura	0
20	Madura perempuan 2	Madura	Madura	0
21	Uji Madura laki-laki 1	Madura	Madura	777133627
22	Uji Madura laki-laki 2	Madura	Madura	771000413
23	Uji Madura perempuan 1	Madura	Madura	787867460
24	Uji Madura perempuan 2	Madura	Madura	571095766
25	Sunda laki-laki 1	Sunda	Sunda	0
26	Sunda laki-laki 2	Sunda	Sunda	0
27	Sunda perempuan 1	Sunda	Sunda	0
28	Sunda perempuan 2	Sunda	Sunda	0
29	Uji Sunda laki-laki 1	Sunda	Madura	713471100
30	Uji Sunda laki-laki 2	Sunda	Jawa	247369902
31	Uji Sunda perempuan 1	Sunda	Kaili	760877067
32	Uji Sunda perempuan 2	Sunda	Madura	758981396

3. Hasil rekognisi dengan DTW-KNN

Hasil rekognisi dari pengujian data terhadap metode DTW-KNN berserta hasil rekognisi setelah pemberian penanganan terhadap metode tersebut dilampirkan pada **Tabel 4.6** berikut:

Tabel 4.6 Hasil rekognisi dengan DTW-KNN

No	Pengujian	Target	Prediksi	Nilai Prediksi	Handling
1	Jawa laki-laki 1	Jawa	Jawa	50%	Jawa
2	Jawa laki-laki 2	Jawa	Jawa	50%	Jawa
3	Jawa perempuan 1	Jawa	Sunda	25.00%	Jawa
4	Jawa perempuan 2	Jawa	Jawa	50.00%	Jawa
5	Uji Jawa laki-laki 1	Jawa	Sunda	37.50%	Sunda
6	Uji Jawa laki-laki 2	Jawa	Jawa	63%	Jawa
7	Uji Jawa perempuan 1	Jawa	Jawa	37.50%	Jawa
8	Uji Jawa perempuan 2	Jawa	Jawa	50.00%	Jawa
9	Kaili laki-laki 1	Kaili	Sunda	62.50%	Kaili
10	Kaili laki-laki 2	Kaili	Kaili	25%	Kaili
11	Kaili perempuan 1	Kaili	Sunda	50%	Kaili
12	Kaili perempuan 2	Kaili	Sunda	63%	Kaili
13	Uji Kaili laki-laki 1	Kaili	Sunda	63%	Sunda
14	Uji Kaili laki-laki 2	Kaili	Kaili	37.50%	Kaili
15	Uji Kaili perempuan 1	Kaili	Sunda	50.00%	Madura
16	Uji Kaili perempuan 2	Kaili	Sunda	62.50%	Sunda
17	Madura laki-laki 1	Madura	Madura	25%	Madura
18	Madura laki-laki 2	Madura	Madura	37.50%	Madura
19	Madura perempuan 1	Madura	Madura	50%	Madura
20	Madura perempuan 2	Madura	Sunda	50%	Madura
21	Uji Madura laki-laki 1	Madura	Sunda	37.50%	Madura

22	Uji Madura laki-laki 2	Madura	Sunda	50.00%	Madura
23	Uji Madura perempuan 1	Madura	Kaili	50%	Madura
24	Uji Madura perempuan 2	Madura	Kaili	37.50%	Madura
25	Sunda laki-laki 1	Sunda	Sunda	63%	Sunda
26	Sunda laki-laki 2	Sunda	Sunda	75.00%	Sunda
27	Uji Sunda perempuan 1	Sunda	Sunda	75.00%	Sunda
28	Sunda perempuan 2	Sunda	Sunda	75%	Sunda
29	Uji Sunda laki-laki 1	Sunda	Sunda	50.00%	Madura
30	Uji Sunda laki-laki 2	Sunda	Sunda	63%	Sunda
31	Uji Sunda perempuan 1	Sunda	Kaili	50.00%	Sunda
32	Uji Sunda perempuan 2	Sunda	Sunda	50.00%	Sunda

Untuk perbedaan hasil rekognisi dari setiap metode dalam melakukan pengujian didapatkan hasil sesuai pada **Tabel 4.7** berikut.

Tabel 4.7 Perbandingan hasil setiap metode

	KNN	DTW	DTW-KNN
Pengambilan hasil	Membandingkan data uji dengan semua data yang ada di data koleksi kemudian mencari K tetangga terdekat dengan banyak K nilai terkecil menggunkan metode *eucliden*	Membandingkan data uji dengan semua data yang ada di data koleksi dengan cara membuat matriks dengan ukuran 1 x data uji dengan 1	Sama halnya dengan metode KNN, hanya saja pada metode ini metode *eucliden distance* pada metode KNN diubah menjadi

	distance	x data koleksi dan memnehui setiap sel matriks dengan persamaan yang telah didefiniskan pada subab DTW, nilai terkecil merupakan hasil dari metode ini.	metode DTW, serta dalam pengambilan hasil akhir juga diberikan sebuah penagngan guna meminimalisirkan hasil kesalahan rekognisi.
Hasil rekognisi	Terdapat 13 data yang tidak sesuai dengan hasil yang sebenarnya	Terdapat 6 data yang tidak sesuai dengan hasil yang sebenarnya	Terdapat 10 data yang tidak sesuai dengan hasil yang sebenarnya, namun setelah pemberian penanganan (handling), data kesalahan dapat diminamalisir hingga menyisikan sebanyak 5 data yang tidak sesuai
Kecepatan	2 menit	1 jam 17 menit	1 jam 20 menit (menambahkan 2

			menit dari hasil DTW namun keduanya merupakan 1 tahapan dalam satu kali peroses)

Setelah hasil rekognisi didapatkan, langkah selanjutnya adalah mengukur nilai *accuracy, precision*, dan *recall* menggunakan ROC. Sebelum mendapatkan hasil ROC, berikut hasil pengujian dari setiap masing masing metode dalam menyelesaikan hasil pengujiannya:

Tabel 4.8 Hasil ROC Metode KNN

KNN	Jawa	Kaili	Madura	Sunda
Jawa	6	0	0	2
Kaili	2	4	0	2
Madura	2	5	0	1
Sunda	0	2	0	6

Pada **Tabel 4.8** diatas merupakan hasil ROC dari pengujian data terhadap metode KNN, dimana hasil dari nilai uji merupakan nilai probabilitas dari banyaknya tetangga terdekat dari data uji. ROC pada KNN menggambarkan nilai nilai yang telah dijelaskan pada pembahasan ROC sebelumnya pada **Tabel 4.1**, sehingga pada tabel ini dapat disimpulkan bahwa pengujian terhadap suku jawa menghasilkan 6 nilai *True Positive* dan 2 *False Positive*, dan 1 *False Negative*.

Untuk Suku Kaili, mendapatkan 4 *True Positive*, 4 *False Positive*, 7 *False Negative*. Suku Madura, mendapatkan 0 *True Positive*, 8 *False Positive* 0 *False Negative*. Suku Sunda, mendapatkan 6 *True Positive*, 2 *False Positive*, dan 5 *False Negative*.

Tabel 4.9 Hasil ROC Metode DTW

DTW	Jawa	Kaili	Madura	Sunda
Jawa	7	1	0	0
Kaili	0	5	1	2
Madura	0	0	8	0
Sunda	1	1	2	4

Pada **Tabel 4.9** diatas merupakan hasil ROC dari pengujian data terhadap metode DTW, dimana hasil dari nilai uji merupakan nilai terkecil dari pengambilan path dengan nilai terkecil dari matriks yang terbentuk. ROC pada DTW menggambarkan nilai nilai yang telah dijelaskan pada pembahasan ROC sebelumnya pada **Tabel 4.1**, sehingga pada tabel ini dapat disimpulkan bahwa pengujian terhadap suku jawa menghasilkan 7 nilai *True Positive* dan 1 *False Positive*, serta 1 *False Negative*. Untuk Suku Kaili, mendapatkan 5 *True Positive*, 3 *False Positive*, 2 *False Negative*. Suku Madura, mendapatkan 8 *True Positive*, 0 *False Positive*, 3 *False Negative*. Suku Sunda, mendapatkan 4 *True Positive*, 4 *False Positive*, dan 3 *False Negative*.

Tabel 4.10 Hasil ROC Metode DTW-KNN

DTW-KNN	Jawa	Kaili	Madura	Sunda
Jawa	7	0	0	1
Kaili	0	6	2	0
Madura	0	0	8	0
Sunda	0	1	0	7

Pada **Tabel 4.10** diatas merupakan hasil ROC dari pengujian data terhadap metode DTW-KNN, dimana hasil dari nilai uji ini sama halnya dengan metode KNN, hanya saja pada proses pencarian path terdekat dengan eucliden distance diubah dengan proses DTW. ROC pada DTW menggambarkan nilai nilai yang telah dijelaskan pada pembahasan ROC sebelumnya pada **Tabel 4.1**, sehingga pada tabel ini dapat disimpulkan bahwa pengujian terhadap suku jawa menghasilkan 7 nilai *True Positive* dan 1 *False Positive*, serta 0 *False Negative*. Untuk Suku Kaili, mendapatkan 6 *True Positive*, 2 *False Positive*, 1 *False Negative*. Suku Madura, mendapatkan 8 *True Positive*, 0 *False Positive*, 2 *False Negative*. Suku Sunda, mendapatkan 7 *True Positive*, 1 *False Positive*, dan 1 *False Negative*.

Dari tabel ROC diatas maka didapatkan hasil *accuracy*, *presision*, dan, *recall* di **Tabel 4.11** menggunakan persamaan yang telah dijabarkan diatas:

Tabel 4.11 Hasil akurasi pengujian ROC

ROC	*Accuracy*	*Precission*	*Recall*
KNN	50%	37%	50%
DTW	75%	74%	75%
DTW-KNN	88%	88%	87.5%

4.3 Pembahasan

Setelah diperoleh nilai pengujian menggunakan ROC pada **Tabel 4.9**, diperoleh nilai akurasi dari metode DTW-KNN yang dibangun oleh penulis sebesar 88%, dengan perolehan nilai presisi sebesar 88% dan *recall* sebesar 87.5%. Hal ini menunjukan bahwa nilai akurasi dari metode DTW-KNN lebih tinggi dibandingkan metode KNN ataupun DTW.

Jika ditinjau secara mendalam metode DTW-KNN memberikan akurasi yang hampir sama dengan KNN. Hal ini disebabkan karena harus melihat nilai dari tetangga terdekat, sehingga harus melakukan pengelompokan dari setiap nilai yang ada. Berbeda halnya dengan DTW yang hanya mencari nilai terkecil dari path yang terbentuk. Oleh karena itu untuk membuat algoritma DTW-KNN agar bisa mendapatkan hasil yang lebih baik perlu ditambahkan penanganaan (handler) dengan membandingkan hasil yang diperoleh dari KNN dan DTW. Terbukti pada hasil akhir dengan menambahkan handler terhadap DTW-KNN, metode tersebut mampu mendapatkan akurasi sebesar 88%, tertinggi jika dibandingkan dengan metode KNN ataupun DTW.

Ditinjau dari efisiensi, penggunaan metode DTW yang memproyeksikan matriks sebesar ukuran data koleksi dan data uji membuat estimasi waktu berjalan dengan lama (pengujian pada penelitian ini menghabiskan waktu selama 1 jam 17 menit untuk 1 kali pengujian). Hal tersebut disebabkan perhitungan nilai setiap sel yang ada di dalam matriks tersebut dalam jumlah yang besar, semisal dalam penelitian ini sebanyak 4000 x 4000 matriks. Kelebihan dari DTW itu sendiri sebenarnya dapat mengukur kemiripan data jika panjang datanya berbeda, namun menjadi kelemahan tersendiri jika data yang diinputkan masuk dalam kategori data yang besar.

Allah SWT. berfirman bahwa setiap manusia telah diciptakan berbeda-beda dan saling berkelompok untuk saling mengenal (QS. Al-Hujurat/49:13).

يَا أَيُّهَا النَّاسُ إِنَّا خَلَقْنَاكُمْ مِنْ ذَكَرٍ وَأُنْثَى وَجَعَلْنَاكُمْ شُعُوباً وَقَبَائِلَ لِتَعَارَفُوا إِنَّ أَكْرَمَكُمْ عِنْدَ اللَّهِ أَتْقَاكُمْ إِنَّ اللَّهَ عَلِيمٌ خَبِيرٌ

Artinya:
"Hai manusia, sesungguhnya Kami menciptakan kamu dari seorang laki-laki dan seorang perempuan dan menjadikan kamu berbangsa-bangsa dan bersuku-suku supaya kamu saling mengenal. Sesungguhnya orang yang paling mulia di antara kamu di sisi Allah ialah orang yang paling bertakwa di antara kamu. Sesungguhnya Allah Maha Mengetahui lagi Maha Mengenal".
(Qs. al-Hujurat/49:13)

Dalam tafsir Ibnu Katsir dijelaskan bahwa Allah subhanahu wa ta'ala menceritakan kepada manusia bahwa Dia telah menciptakan mereka dari diri yang satu dan darinya Allah menciptakan istrinya, yaitu Adam dan Hawa, kemudian Dia menjadikan mereka berbangsa-bangsa. Pengertian bangsa dalam bahasa Arab adalah *sya'bun* yang artinya lebih besar daripada kabilah, sesudah kabilah terdapat tingkatan-tingkatan lainnya yang lebih kecil seperti *fasa-il* (puak), *'asya-ir* (Bani), *'ama-ir*, *Afkhad*, dan lain sebagainya. Mujahid telah mengatakan sehubungan

dengan makna firman-Nya: supaya kamu saling kenal-mengenal. (Al-Hujurat: 13) Seperti disebutkan si Fulan bin Fulan dari kabilah anu atau bangsa anu. Sufyan As-Sauri mengatakan bahwa orang-orang Himyar menisbatkan dirinya kepada sukunya masing-masing, dan orang-orang Arab Hijaz menisbatkan dirinya kepada kabilahnya masing-masing.

Allah SWT. juga berfirman bahwa Dialah yang menciptakan langit, bumi dan manusia yang berlain-lainan dari bahasa hingga warna kulit (QS. Ar-Rum/30:22).

وَمِنْ آيَاتِهِ خَلْقُ السَّمَاوَاتِ وَالأَرْضِ وَاخْتِلافُ أَلْسِنَتِكُمْ وَأَلْوَانِكُمْ إِنَّ فِي ذَلِكَ لآيَاتٍ لِلْعَالِمِينَ

Artinya:
" Dan di antara tanda-tanda kekuasaan-Nya ialah menciptakan langit dan bumi dan berlain-lainan bahasamu dan warna kulitmu. Sesungguhnya pada yang demikan itu benar-benar terdapat tanda-tanda bagi orang-orang yangi mengetahui".

(Ar-Rum/30:22)

Dijelaskan dalam tafsir Ibnu Katsir bahwa Dia menciptakan langit yang tinggi, luas, tembus pandang, tampak berkilauan bintang-bintangnya, baik yang beredar maupun yang tetap. Dan Dia menciptakan bumi yang datar lagi padat berikut gunung-gunungnya, lembah-lembahnya, lautannya, padang pasirnya, hewan-hewannya, dan pepohonannya. "Dan berlain lainan bahasamu" yakni berbeda-beda bahasa, ada yang berbahasa Arab, ada yang berbahasa Tartar, ada yang berbahasa Kurdi, ada yang berbahasa Indian, ada yang berbahasa Afrika, ada yang berbahasa Etiopia, ada yang berbahasa Inggris. Mereka -selain yang pertama- adalah orang-orang yang berbahasa 'ajam (non-Arab). Mereka terdiri dari berbagai bangsa, antara lain Sicilia, Armen, Kurdi, Tartar, dan lain sebagainya. Jumlah bahasa Bani Adam banyak sekali, begitu pula perbedaan warna kulitnya, masing-masing mempunyai ciri khas tersendiri. Semua penduduk bumi sejak Allah menciptakan Adam sampai hari kiamat, masing-masing

mempunyai sepasang mata, sepasang alis, hidung, kelopak mata, mulut, pipi, dan seseorang dari mereka tidak serupa dengan yang lain. Tetapi masing-masing pasti mempunyai sesuatu ciri yang membedakan yang seorang dari yang lainnya, baik itu dalam hal rupa, bentuk, ataupun bahasa. Perbedaan itu ada yang jelas dan ada yang samar, yang hanya diketahui setelah dilihat dengan teliti.

BAB V

KESIMPULAN DAN SARAN

6.1 Kesimpulan

Dari hasil implementasi dan uji coba sistem yang telah dilakukan oleh penulis dapat disimpulkan bahwa terdapat banyak metode untuk melakukan rekognisi aksen, diantaranya adalah metode klasifikasi yang telah dimodifikasi oleh penulis, yaitu *Dynamic Time Warping K Nearset Neighbour* (DTW-KNN). sebelum melakakun rekognisi tentunya sinyal audio yang masuk harus diekstraksi hal ini dilakukan untuk mengurangi banyaknya sinyal input yang akan diproses. *Mel Frequencey Cepstrum Coefficient* (MFCC) merupakan satu diantara metode ekstraksi suara yang populer digunakan dalam penelitian pengolah suara. Setelah sinyal diekstraksi kemudian penulis mencoba melakukan rekognis dengan 3 metode yang berbeda, yaitu *K-Nearest Neighbour* (KNN), *Dynamic Time Warping* (DTW), dan DTW-KNN. Dari 40 data sampel dilakukan pengujian terhadap 4 suku dimana masing masing suku terdapat 8 data uji. Dengan menggunakan metode pengujian *Receiver Operating Charateistic* (ROC) didapatkan hasil bahwa metode modifikasi DTW-KNN memiliki akurasi sebesar 88%, lebih tinggi jika dibandingkan dengan metode DTW ataupun KNN yang mendapatkan akurasi sebesar 75% dan 50%. Hasil akurasi dari setiap metode tersebut berbeda karena dalam operasinya memiliki proses yang berbeda-beda. Pada proses metode KNN hasil merupakan mayoritas dari banyaknya tetangga K terdekat dengan menggunakan persamaan *Euclidean Distance* dalam hasil outputnya, sehingga pada metode ini waktu yang dibuthkan hanya berkisar 2 menit untuk mendapatkan output yang diinginkan. Pada proses metode DTW hasil

merupakan nilai yang paling kecil dari *path* yang terbentuk yang didapatkan dengan memproyeksikan matriks panjang data uji dengan panjang data koleksi satu per satu, kemudian pengisian nilai pada matriks yang terbentuk memiliki persamaan dengan memperhatikan nilai pada sel matriks sebelumnya kemudian menarik path terpendek secara diagonal, sehingga pada proses ini menyebabkan komputasi yang begitu lama dengan menghabiskan waktu sebesar 1 jam 17 menit. Sedangkan pada proses DTW-KNN merupakan proses KNN dimana persamaan Euclidean Distance diubah menjadi proses DTW, kemudian dilanjutkan dengan proses KNN. Pada metode DTW-KNN juga menghabiskan waktu sebesar 1 jam 20 menit (meneruskan proses DTW sebelumnya).

6.2 Saran

Untuk pengembangan sistem ini di kemudian hari diperlukan beberapa perbaikan dan penambahan untuk mendapatkan hasil yang lebih baik, diantaranya:

a. Penggunaan metode DTW-KNN mendapatkan nilai akurasi tertinggi namun sangat tidak efisien jika digunakan secara *real-time*, mengingat waktu pengujian yang menghabiskan waktu selama 1 jam 17 menit, sehingga perlu menggunakan metode yang dapat memberikan *output* dengan cepat tanpa mengabaikan nilai akurasi keberhasilan rekognisi.
b. Penelitian ini dapat diteruskan dengan menambahkan aksen suku lainnya ataupun melakukan peningkatan pada metode dengan merekognisi suku dan bahasa yang sama namun memiliki pola aksen yang berbeda dikarenakan perbedaan geografis.

DAFTAR PUSTAKA

Abdullah, M. 2003. *Tafsir Ibnu Katsir Jilid 6*. Bogor: Pustaka Imam Asy- Syafi'I.

Abdullah, M. 2003. *Tafsir Ibnu Katsir Jilid 7*. Bogor: Pustaka Imam Asy- Syafi'I.

Anggoro, A. 2015. *Analisis Karakter Suara Penderita Bibir Sumbing Terhadap Suara Normal Menggunakan Metode Mel Frequency Cepstrum Coefficient dan Dynamic Time Warping*[Skripsi]. Malang (ID). Fakultas Sains dan Tekonologi, Teknik Informatika. Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Fadli, I. 2016. *Kendali Pintu Air Otomatis Berbasis Speech Recognition Menggunakan Metode MFCC dan Jaringan Syaraf Tiruan*. Politeknosains, Vol. XV, No. 2. ISSN: 1829-6181.

Goswani, S. 2013. *An Introduction to Various Features of Speech Signal*. Dept. Of ECE, Gauhati University. Gunwahati, India.

Jaafar, H., Ramli D.A., Shahrudin, S., 2013. *MFCC Based Frog Identification System In Noisy Environtment*. International Conference On Signal Image Proccessing Application (ICSIPA). Universiti Sains Malaysia.

Koolagudi, S.G., Rastologi, D., Rao, K.S. 2012. *Identification of Language Using Mel- Frequency Cepstrum Corfficient (MFCC)*. International Conference On Modelling, Optimation, And, Computing (ICMOC). Published by Elsevier Ltd. Doi: 10.1016/j.proeng.2012.06.392.

Mane, A.D, A, Rashmi. R, Tade, S.L. 2013. *Identification & Detection System For Animal From Their Vocalization*. International Journal of Advanced Computer Research. Vol. 03. No. 03. ISSN(print): 2249-7277, ISSN(online): 2277-7970.

Nagawade, M.S., Ratnaparkhe, V.R. 2017. *Musical Instrument Identification Using MFCC*. 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT), India.

Nalini, N.J., Palanivel, S. 2015. *Music Emotion Recognition: The Combine Evidence of MFCC and Residual Phase*. Egyptian Informatics Journal. Dept of Computer Science and Engineering, Annamalai University, India. Doi:10.1016/j.eij.2015.05.004.

Putra, Darma & Resmawan, Adi. 2011. *Verifikasi Biometrika Suara Menggunakan Metode MFCC dan DTW*. Jurnal Lontar Komputer Vol. 2 No. 1. ISSN: 2088-1541.

Rivai, Muhammad & Suwito. 2016. *Pengenalan Suara Burung Menggunakan Mel Frequency Coefficient Cepstrum Dan Jaringan Syaraf Tiruan Pada Sistem Pengusir Hama*. Jurnal Nasional Elektro Vol. 5 No. 1. ISSN: 2302-2949.

Yücesoy, E., Nabiyev, V. 2013. *Gender Identification Of A Speaker Using MFCC and GMM*. 8th International Conference On Electrical and Electronics Engineering (ELECO), Turkey. doi: 10.1109/ELECO.2013.6713992