Implementasi Sistem Pengenalan Ucapan Bahasa Indonesia Menggunakan Kombinasi MFCC dan PCA Berbasis HMM

Fathurrohman Elkusnandi[1], Adiwijaya[2], Untari Novia Wisesty[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]fathurrohmane@students.telkomuniversity.ac.id, [2]adiwijaya@telkomuniversity.ac.id,
[3]untarinw@telkomuniversity.ac.id

Abstrak

Karya tulis ini membahas implementasi sistem pengenalan ucapan dalam bahasa Indonesia dimana suatu perangkat membaca *file* suara lalu ditranslasikan menjadi teks sesuai dengan kata yang diucapkan pada *file* suara tersebut didasarkan pada kata yang sudah dilatih ke dalam sistem. Metode MFCC digunakan untuk proses ekstraksi ciri dimana akustik vektor atau vektor ciri direduksi jumlah dimensinya menggunakan PCA, lalu hasil ektraksi ciri tersebut diklasterkan dengan algoritma Y. Linde, A. Buzo, dan R. Gray (LBG) dan diklasifikasikan menggunakan HMM. Pengurangan dimensi pada vektor akustik atau vektor ciri dilakukan karena jumlah dimensi data yang diekstrak dari sinyal suara menggunakan MFCC yang tinggi. Metode PCA dipilih karena PCA mampu memproyeksikan data ke *space* yang bervariansi tinggi sehingga data yang *redundant* atau kurang signifikan bisa direduksi. Selain itu pengurangan dimensi pada vektor ciri dapat meningkatkan performansi sistem dikarenakan jumlah dimensi yang berkurang akan mengurangi data yang harus dikalkulasi oleh sistem. Hasilnya sistem mampu mengenali kata dengan rata – rata akurasi sebesar 80,19%, namun performansi sistem tidak naik secara signifikan yaitu paling tinggi hanya sebesar 3,29% untuk proses pelatihan, dikarenakan hanya proses *kuantisasi vektor* yang jumlah data untuk dikalkukasinya berkurang, selain itu proses PCA menambah beban sistem yang sebelumnya tidak ada.

Kata kunci : pengenalan ucapan, MFCC, LBG, PCA, HMM

Abstract

This paper talks about the implementation of speech recognition in Bahasa Indonesia. The system will translate audio file into text according to the spoken word that has been trained into the system. MFCC method is used for feature extraction where feature vector dimension is reduced with PCA method, then it quantized using Y. Linde, A. Buzo, and R. Gray (LBG) and classified with HMM method. The reduction of the feature vector dimension is applied because the number of dimension in MFCC feature from MFCC method is very high. PCA is chosen because the PCA can project the data into a space where the variance is high with the order of the dimension, so the redundant and less important data can be reduced. Also, the dimension reduction can affect the system performance, because lesser dimension means lesser data to be calculated. The results show that the system can recognizes word with 80.19% accuracy, but there is no significant improvement in system performance, the highest improvement is at around 3.29% for training process, because only at vector quantization process where the number of data has decreased, also the PCA process add process time that wasn't there before.

Keywords : *speech recognition*, MFCC, LBG, PCA, HMM

1. Pendahuluan

Latar Belakang

Suara merupakan salah satu metode interaksi atau komunikasi antar manusia. Selain itu suara juga digunakan untuk berinteraksi antar manusia dengan komputer. Namun, agar perangkat komputer mampu berinteraksi melalui suara, perlu adanya suatu sistem pada perangkat yang dapat melakukan pengenalan suara manusia, sistem tersebut dinamakan *automatic speech recognition* [**1**].

Automatic speech recognition atau *speech recognition* adalah proses pengenalan suara manusia berupa kata atau kalimat agar bisa dikenali oleh perangkat komputer untuk berbagai macam kebutuhan, seperti aplikasi *speech-to-text* yang mampu merubah kata atau kalimat yang diucapkan oleh manusia menjadi teks sesuai dengan apa yang diucapkannya; aplikasi *voice command* yang mampu membuat ucapan manusia menjadi penghubung untuk mengendalikan perangkat komputer, telpon pintar atau yang lainnya. Perkembangan teknologi *speech recognition* ini sudah dikembangkan sejak tahun 1960-an [**1**] dan menghasilkan banyak metode dalam

pengembangannya, seperti untuk ekstraksi ciri *Mel-Scale Frequency Cepstral Coefficient* (MFCC) dan untuk klasifikasi *Hidden Markov Model* (HMM).

Dalam karya tulis ini telah diimplementasikan *automatic speech recognition* bahasa Indonesia menggunakan inputan suara *isolated word* atau satu kata per waktu [**1**] berupa nama – nama kota di Indonesia [**2**,**3**,**4**,**5**,**6**,**7**]. Metode yang digunakan untuk ekstraksi ciri adalah MFCC dan untuk klasifikasi adalah HMM. Kedua metode tersebut sudah banyak digunakan dalam pengembangan sistem pengenalan ucapan dan menghasilkan akurasi yang baik [**1**,**8**,**9**,**10**,**11**]**.** Namun terdapat penambahan metode *Principal Component Analysis* (PCA) setelah sinyal suara diubah menjadi vektor ciri untuk mereduksi jumlah dimensi pada vektor cirinya. PCA digunakan karena sifat data yang dihasilkan pada ekstraksi ciri MFCC yang kompleks atau berdimensi tinggi [**10**,**12**,**13**]. Meski demikian pada fitur MFCC tersebut terdapat banyak sekali redudansi pada datanya [**12**]. PCA akan membantu menyederhanakan data yang *redundant* dan mengurangi jumlah dimensi pada *dataset* [**10**,**11**,**13**,**14**] sehingga diharapkan akan dapat mengurangi beban sistem tanpa mengurangi akurasi pendeteksian kata secara drastis.

Topik dan Batasannya

Sinyal suara yang digunakan pada sistem pengenalan ucapan tidak bisa digunakan secara langsung. Untuk dapat diolah, sinyal suara harus diekstrak cirinya terlebih dahulu menggunakan algoritma MFCC. Algoritma ini sudah banyak sekali digunakan dalam proses ekstraksi sinyal suara dan menghasilkan data yang baik [**8**]. Algoritma MFCC secara sederhana melakukan analisis sinyal suara pada rentang waktu yang singkat lalu dikonversikan ke dalam skala *mel* dan difilter menggunakan *Mel-filter bank*. Namun, hasil fitur dari proses MFCC ini menghasilkan data dengan jumlah dimensi yang tinggi [**10**,**12**,**13**]. Ini dikarenakan banyak sekali variabel – variabel yang dapat diekstraksi pada suatu sinyal suara. Secara matematis data tersebut mempunyai banyak redudansi antar datanya [**12**,**15**]. Maka dari itu algoritma PCA digunakan untuk mereduksi jumlah dimensi pada fitur MFCC. PCA mampu membuang data – data yang *redundant* dan tidak relevan terhadap sifat suara yang diucapkan dengan mentransformasikan data ke dalam *space* dimana dimensi diurutkan berdasarkan variansi datanya. Pengurangan dimensi ini berpengaruh terhadap akurasi dan performansi dari sistem.

Sistem hanya mampu mengenali ucapan berupa *isolated word* atau satu kata dalam satu waktu yang telah dilatih sebelumnya pada proses latih. Selain itu sistem hanya menerima inputan data suara berupa *file*, tidak mampu menerima data suara secara langsung misal menggunakan mikrofon. Data yang akan diinputkan sebatas suara orang baik laki – laki maupun perempuan yang menyebutkan sepuluh nama – nama kota yang ada di Indonesia yaitu “Aceh”, “Bandung”, “Bengkulu”, “Cilacap”, “Garut”, “Jakarta”, “Klaten”, “Majalengka”, “Padang”, dan “Solo”. Selain itu data yang digunakan bersifat *environment controlled* dimana datanya direkam pada studio tertutup sehingga *noise* pada data suara sangatlah minimal dan intonasi suara formal tanpa ada penekanan suara.

Tujuan

Mengimplementasikan sistem pengenalan ucapan menggunakan algoritma MFCC, LBG dan HMM, lalu menganalisa dampak dari pengurangan dimensi pada fitur MFCC dengan algoritma PCA terhadap akurasi dan performansi sistem. Pengurangan fitur MFCC akan berdampak pada perubahan akurasi sistem baik mengalami peningkatan maupun penurunan. Pengurangan fitur MFCC juga berdampak pada peningkatan performansi sistem dikarenakan penurunan dimensi fitur MFCC akan berdampak pada pengurangan data yang harus dikalkulasi oleh sistem.

Organisasi Tulisan

Studi pustaka menjelaskan teori – teori atau algoritma yang digunakan pada sistem yang dibangun. Bagian *Sistem yang Dibangun* menjelaskan secara rinci proses – proses yang terjadi pada sistem baik alur data maupun algoritma yang digunakan. Selanjutnya pada bagian *Evaluasi* menjelaskan mengenai hasil pengujian terhadap sistem yang telah dibangun beserta analisanya. Lalu yang terakhir bagian *Kesimpulan* menyimpulkan bagaimana tujuan awal karya tulis ini terhadap hasil analisa keluaran sistem yang telah dibangun.

2. Studi Pustaka

2.1 Mel-Frequency Cepstral Coefficient (MFCC)

Mel-Frequency Cepstral Coefficient adalah suatu koefisien yang merepresentasikan sinyal suara dimana didasarkan pada presepsi pendengaran manusia yang tidak bisa mempersepsikan frequensi suara diatas 1000 hz [**8**,**14**]. MFCC mempunyai dua buah filter yaitu filter linear yang berfrequensi dibawah 1000 hz dan filter logaritmik yang berfrequensi diatas 1000 hz. Proses dari MFCC yaitu :

a. Pre-emphasis

Pre-emphasis digunakan untuk meningkatkan suara pada frekuensi tinggi, dikarenakan terdapat lebih banyak energi pada frekuensi yang lebih rendah. Persamaannya adalah sebagai berikut :

$$a[n] = a[n] - b * a[n-1], \qquad 1 < n < c \tag{2.1}$$

Dimana *a* merupakan sinyal suara pada *n*; *b* merupakan koeffiseien untuk *pre-emphasis*; dan *c* merupakan panjang sinyal suara.

b. Framing

Sinyal suara bersifat stationer pada rentang waktu yang singkat [**8**]. Maka dari itu agar dapat diolah sinyal suara harus dipotong – potong dengan rentang waktu yang singkat, konsep ini disebut juga *short time analysis* [**16**].

c. Windowing

Proses *framing* akan mengakibatkan sinyal suara *discontinue* antar satu *frame* dengan *frame* selanjutnya. Maka dari itu perlu diperbaiki dengan mengaplikasikan *Hamming Window* terhadap sinyal suara tersebut. Persamaannya adalah sebagai berikut :

$$\omega[n] = \begin{cases} 0.54 - 0.46cos\left(\frac{2\pi n}{L-1}\right), & 0 \le n \le L-1 \\ 0, & n\ lainnya \end{cases} \tag{2.2}$$

Dimana *n* merupakan sample sinyal suara ke-*n*; *L* adalah panjangnya *frame* atau jumlah sampel dalam satu *frame*.

d. Discrete Fourier Transform (DFT) / Fast Fourier Transform (FFT)

Sinyal suara akan diubah ke dalam domain frekuensi dengan menggunakan algoritma *Fast Fourier Transform.* Persamaannya adalah sebagai berikut :

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j2\frac{n}{N}kn} \tag{2.3}$$

Dimana *x* merupakan sinyal suara pada *k frame* dan *N* jumlah sampel pada *frame*.

e. Mel-Filter Bank Processing dan Log

Sinyal suara akan diubah ke dalam skala *mel* dengan menggunakan persamaan berikut :

$$Mel(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.4}$$

Dimana f merupakan frekuensi yang akan diubah ke *Mel-spectrum*.

Setelah itu sinyal suara akan difilter dengan menggunakan *Mel-Filter bank*. *Mel-Filter bank* dibuat sebanyak 30 filter dimana filter berjarak secara linear pada frekuensi suara di bawah 1000 hz dan secara logaritmik pada frekuensi suara di atas 1000 hz [**16**].

f. Discrete Cosine Transform

Log-mel spectrum kemudian diubah kembali ke domain waktu.

g. Delta Energy dan Delta Spectrum

Proses penambahan fitur pada *cepstral*. Dengan menggunakan persamaan :

$$Energy = \sum_{t=t_1}^{t_2} x^2[\mathrm{t}] \tag{2.5}$$

Dimana *x* merupakan sinyal suara pada *frame* dengan rentang waktu t_1 dan t_2.

Proses perhitungan *delta* dihitung dengan menggunakan persamaan :

$$d(t) = \frac{c(t+1) - c(t-1)}{2} \tag{2.6}$$

Dimana *t* merupakan waktu; *c* merupakan fitur MFCC atau *cepstral* pada *t* waktu.

2.2 Principal Component Analysis (PCA)

PCA adalah suatu metode untuk mengekstraksi struktur dari *dataset* berdimensi tinggi, sehingga menjadi *dataset* dengan dimensi yang lebih rendah, dengan menghilangkan variabel yang tidak berkaitan, namun tetap memungkinkan mempertahankan sebanyak mungkin dari variasi yang ada dalam kumpulan *dataset* [**11**,**12**,**13**].

Caranya dengan merotasi data dengan mencari vektor *eigen* dan nilai *eigen* pada suatu *dataset*, lalu *dataset* diurutkan berdasarkan nilai *eigen*-nya dan diproyeksikan dengan vektor *eigen*-nya. Nilai *eigen* adalah suatu angka yang menunjukan besarnya persebaran data (*variance*) sedangkan vektor *eigen* adalah suatu vektor yang akan membagi *dataset* menjadi data yang paling tersebar, vektor *eigen* dengan nilai *eigen* terbesar akan menjadi komponen utamanya (*principal component*) [**11**,**12**,**13**]. Tahap dari proses algoritma PCA adalah sebagai berikut :

a. Mencari rata - rata

Menghitung nilai rata-rata dengan menggunakan persamaan :

$$\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij} \tag{2.7}$$

Dimana x_{ij} merupakan data pada baris *j* dan kolom *i* dengan jumlah kolom atau dimensi sebesar *n*.

b. Mencari co-variance

Menghitung nilai *co-variance* dengan menggunakan persamaan :

$$Cov(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \tag{2.8}$$

Dimana *x* dan *y* merupakan dimensi data pada *index x* dan *y; n* merupakan jumlah dimensi data; $\bar{x}$ dan $\bar{y}$ merupakan rata – rata fitur MFCC pada *x* dan *y*.

Setelah didapatkan *co-variance* maka perlu dicari vektor *eigen* (φi) dan nilai *eigen* (ψi) dengan menggunakan persamaan :

$$C\varphi_i = \varphi_i\, \psi_i \tag{2.9}$$

Dimana *C* merupakan matriks *co-variance*; φ_i merupakan vektor *eigen* pada *index i*; dan ψ_i merupakan nilai *eigen* pada *index i*.

$$\psi_i = \frac{1}{M}\sum_{j=1}^{M} {\varphi_i}^T \varphi_j \tag{2.10}$$

Dimana ψ_i merupakan nilai *eigen* pada *index i*; *M* merupakan dimensi matriks *co-variance;* dan φ_i merupakan vektor *eigen* pada *index i*.

c. Urutkan vektor

Setelah didapatkan vektor *eigen* dan vektor *eigen*-nya lalu vektor *eigen* diurutkan berdasarkan dari nilai *eigen* dari terbesar ke terkecil.

d. Ambil N-dimensi data yang akan dipertahankan

Vektor e*igen* akan dipilih berdasarkan jumlah dimensi yang diinginkan.

e. Proyeksikan data awal ke eigenspace.

Data awal dikalikan dengan vektor *eigen* dengan jumlah dimensi yang diinginkan. Hasilnya adalah data yang telah direduksi oleh PCA.

$$V_k = \mathrm{A}\varphi_k \tag{2.11}$$

Dimana *A* merupakan matriks yang akan direduksi; dan φ_k merupakan vektor *eigen* dengan *k element* (Dimensi yang akan dihasilkan setelah reduksi).

2.3 Kuantisasi Vektor

Kuantisasi Vektor adalah proses pemetaan vektor dari kumpulan vektor menjadi region dalam suatu ruang (data) [**16**,**17**]. Tujuan dari penggunaan kuantisasi vektor adalah mengurangi jumlah vektor yang ada dengan menghitung titik tengah (*centroid/codeword*) dari kumpulan vektor yang serumpun (*cluster*), sehingga akan mengurangi jumlah data yang ada. Kumpulan dari *centroid/codeword* disebut dengan *codebook* [**16**]. Algoritma kuantisasi vektor yang digunakan adalah algoritma Y. Linde, A. Buzo, dan R. Gray (LBG).

LBG adalah metode yang digunakan untuk mencari *centroid* pada proses kuantisasi vektor yang diciptakan oleh sekelompok orang bernama Y. Linde, A. Buzo, dan R. Gray pada tahun 1980 [**9**,**16**]. Pengelompokan data dilakukan dengan cara mencari jarak terdekat antara data dengan *centroid* dengan menggunakan persamaan *Euclidean Distance*. Tahap dari proses algoritma LBG adalah sebagai berikut [**16**,**18**]:

a. Inisialisasi Codebook

Inisialisasi *codebook* dengan satu *centroid.* Posisi *centroid* didasarkan pada semua dataset yang ada.

b. Gandakan Codebook

Gandakan jumlah *centroid* yang ada pada *codebook* dengan parameter yang telah ditetapkan.

c. Pasangkan vektor dengan centroid terdekat

Pasangkan semua dataset dengan *centroid* terdekatnya menggunakan persamaan *Euclidean Distance* [**17**,**19**] sebagai berikut :

$$d(p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} \quad (2.12)$$

Dimana *p* merupakan vektor data dan *q* merupakan vektor *centroid*.

d. Perbaharui centroid

Perbaharui posisi semua *centroid* pada *codebook* berdasarkan dari semua posisi data yang berpasangan dengan *centroid*-nya.

e. Hitung distorsi

Hitung perubahan posisi rata-rata *centroid*.

f. Ulangi

Jika distorsi belum melebihi dari *threshold*-nya maka proses berulang ke tahap *c*. Jika jumlah *centroid* belum sesuai dengan yang diharapkan maka proses akan berulang ke tahap *b*.

2.4 Hidden Markov Model

Hidden Markov Model merupakan varian dari *finite state machine* dimana *current state* tidak bisa diamati (*hidden*) namun setiap *state* menghasilkan *output* atau emisi berupa simbol yang bisa diamati [**20**]. HMM terbentuk dari beberapa variabel yaitu *N* adalah jumlah *state* dalam suatu model *Markov*; *M* adalah jumlah simbol yang ada pada suatu *observation sequence* yang dapat diinputkan ke dalam suatu model *Markov*; *A* adalah probabilitas *state* transisi; *B* adalah probabilitas distribusi simbol pada suatu *state*; dan π adalah probabilitas inisial *state* pada suatu model markov. HMM biasa ditulis seperti dibawah ini :

$$\lambda = (A, B, \pi) \quad (2.13)$$

Dimana A merupakan probabilitas transisi *state; B* merupakan probabilitas emisi simbol pada suatu *state*; dan π merupakan probabilitas inisial *state*.

Terdapat dua algoritma utama yang digunakan untuk proses latih dan uji yaitu algoritma *forward-backward* untuk proses latih dan algoritma *viterbi* untuk proses uji.

a. Algoritma forward

i. Inisialisasi

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \leq i \leq N \quad (2.14)$$

Dimana $\alpha_1(i)$ merupakan variabel inisial *alpha* pada *state i;* π_i merupakan nilai probabilitas inisial *state i*; $b_i(O_1)$ merupakan nilai probabilitas variabel emisi pada *state i* dengan inputan *obervation sequence* O_1; dan *N* merupakan jumlah *state.*

ii. Induksi

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(j) a_{ij}\right] b_j(O_{t+1}) \quad (2.15)$$

Dimana $\alpha_{t+1}(j)$ merupakan nilai variabel alpha pada waktu *t* + 1 dengan *next state j*; $\alpha_t(j)$ merupakan nilai variabel alpha pada waktu *t* dengan inputan *next state j*; dan $b_j(O_{t+1})$ merupakan nilai probabilitas emisi pada *next state j* dengan inputan *obervation sequence* O_{t+1}.

iii. Terminasi

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \quad (2.16)$$

Dimana $P(O|\lambda)$ merupakan nilai probabilitas model λ jika diberikan *observation sequence O*; dan $\alpha_T(i)$ merupakan nilai variabel *alpha* pada *state i.*

b. Algoritma backward

i. Inisialisasi

$$\beta_t(i) = 1 \qquad 1 \leq i \leq N \tag{2.17}$$

Dimana $\beta_t(i)$ merupakan nilai variabel *beta/forward* pada waktu *t*, *state i* dan *N* adalah jumlah *state*.

ii. Induksi

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(O_{t+1})\beta_{t+1}(j) \tag{2.18}$$

Dimana $\beta_t(i)$ merupakan variabel *beta* pada waktu *t* dan *state i;* a_{ij} merupakan nilai probabilitas transisi pada *current state i* dan *next state j*; $b_j(O_{t+1})$ merupakan nilai probabilitas emisi pada *next state j* jika diberikan inputan *observation sequence* O_{t+1}; dan $\beta_{t+1}(j)$ merupakan nilai variabel *beta* pada waktu $t+1$ dengan inputan *next state j*.

c. Algoritma Viterbi

Algoritma *Viterbi* ini akan mencari *state sequence* (*state* mana saja yang dilalui) yang menghasilkan probabilitas tertinggi pada suatu HMM jika diberikan suatu *obervation sequence* [**20**]. Nilai dari probabilias ini yang akan dibandingkan untuk setiap model yang ada pada proses *pengujian*. Algoritmanya adalah sebagai berikut :

i. Inisialiasi

$$\delta_1(i) = \pi_i b_i(O_1) \qquad 1 \leq i \leq N \tag{2.19}$$

$$\psi_i(i) = 0 \tag{2.20}$$

Dimana $\delta_1(i)$ merupakan nilai probabilitas awal pada *state i* (*state i* sebagai inisial *state*) dengan nilai probabilitas inisial *state* π_i , dan nilai probabilitas emisi $b_i(O_1)$ jika diberikan inputan *observation sequence* O_1 pada *state i.*

ii. Rekursi

$$\delta_t(j) = \max_{1 \leq i \leq N}\left[\delta_{t-1}(i)a_{ij}\right]b_j(O_t) \qquad 2 \leq t \leq T, \quad 1 \leq j \leq N \tag{2.21}$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \leq i \leq N}\left[\delta_{t-1}(i)a_{ij}\right] \qquad 2 \leq t \leq T, \quad 1 \leq j \leq N \tag{2.22}$$

Dimana $\delta_t(j)$ merupakan nilai probabilitas pada waktu *t* dan *state j*, lalu dicari nilai tertinggi probabilitas pada waktu $t-1$ yang telah dikalikan dengan probabilitas transisi a_{ij} lalu dikalikan dengan probabilitas emisi pada *state j* jika diberikan inputan *obervation sequence* O_t dengan *T* adalah jumlah *observation sequence* dan *N* adalah jumlah *state*. $\psi_t(j)$ merupakan *state* yang menghasilkan probabilitas tertinggi pada waktu *t* dan *state j.*

iii. Terminasi

$$p^* = \max_{1 \leq j \leq N}[\delta_T(j)] \tag{2.23}$$

$$q_T^* = \operatorname*{argmax}_{1 \leq i \leq N}[\delta_T(j)] \tag{2.24}$$

Dimana p^* merupakan nilai probabilitas dengan *sequence state* yang menghasilkan probabilitas tertinggi dan q_T^* merupakan *state sequence*-nya.

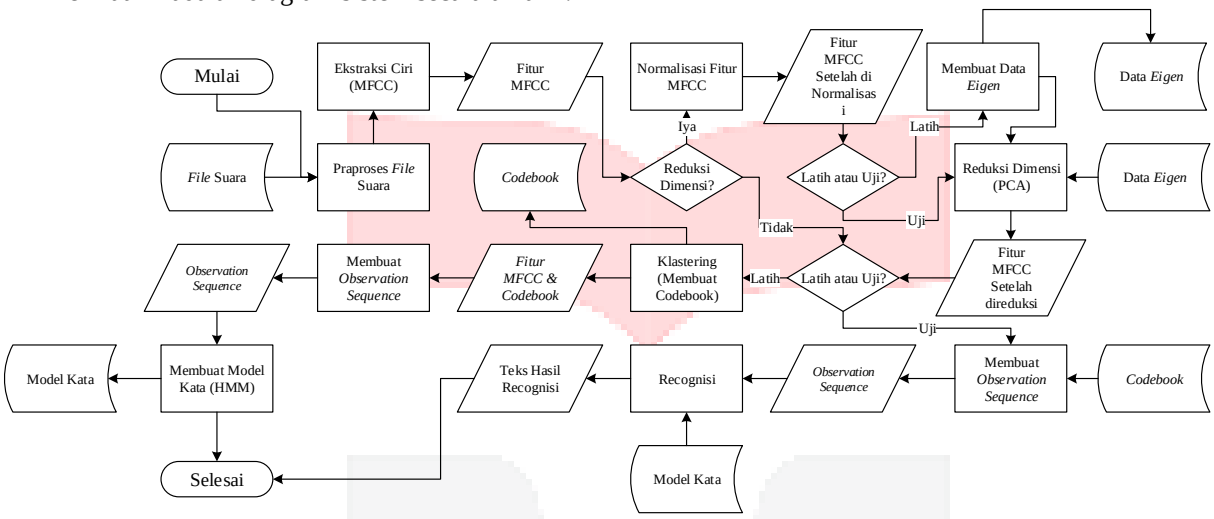
iv. Path

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad \text{t = T – 1, T – 2, ... , 1} \tag{2.25}$$

Dimana q_t^* merupakan hasil akhir *state sequence* pada waktu *t*; ψ_{t+1} merupakan *state* yang telah dicari simpan sebelumnya pada waktu t + 1 jika diberikan inputan *next state*-nya q_{t+1}^* .

3. Sistem yang Dibangun

Berikut ini adalah diagram sistem secara umum :



Gambar 3.1 Diagram Alur Keseluruhan Sistem

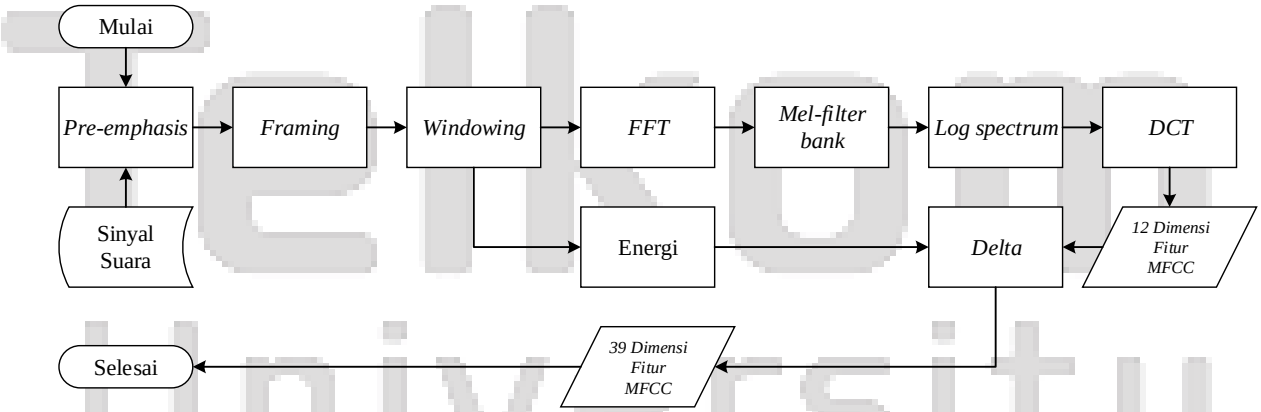
Sistem yang dibangun berupa aplikasi komputer *desktop* yang berjalan pada *Java Virtual Machine*. Sistem terdiri dari dua proses yaitu *pelatihan* dan *pengujian*. Pada proses *pelatihan* sistem akan membaca data latih lalu menyimpan data – data hasil pelatihan ke dalam media penyimpanan yang nantinya akan dimuat ke dalam sistem untuk proses pengujian. Sistem pengenalan ucapan ini terdiri dari lima bagian yaitu *praproses*, *ekstraksi ciri*, *reduksi dimensi*, *vektor kuantisasi/klasterisasi* dan *klasifikasi*.

3.1 Praproses

Pada praproses semua data suara baik untuk proses *pelatihan* maupun *pengujian* akan dihilangkan bagian heningnya (*silence*) dengan menggunakan algoritma *End Point Detection*. Proses ini dilakukan untuk menghilangkan sinyal suara yang kosong sehingga data sinyal suara yang tidak penting akan dihilangkan. Proses berulang untuk semua *file* yang akan dilatih atau diuji ke dalam sistem.

3.2 Ekstraksi ciri

Pada tahap ini sinyal suara akan diekstrak cirinya dengan menggunakan MFCC. Prosesnya digambarkan dengan diagram berikut ini :



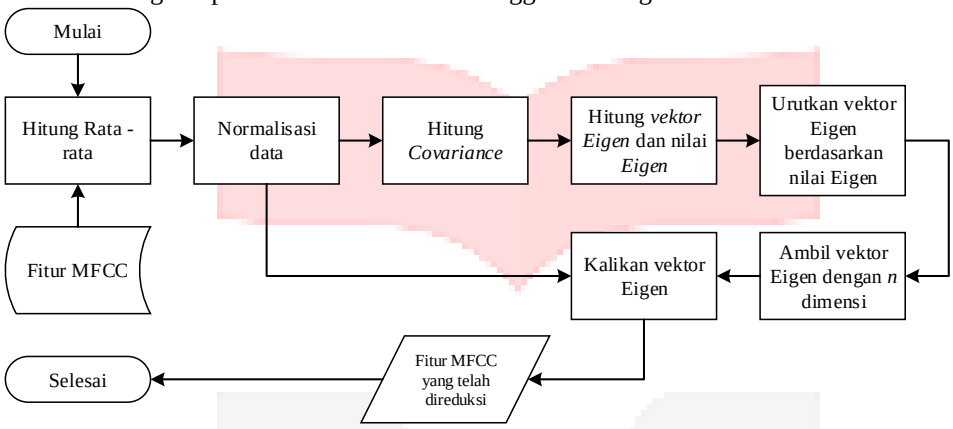
Gambar 3.2 Diagram Alur Proses Ekstraksi Ciri Dengan Algortima MFCC

Sinyal suara akan dipotong – potong menjadi *frame* dengan rentang waktu 21 millisekon dan ditumpuk dengan jarak 10 milisekon. Masing – masing *frame* akan dikalikan dengan *Hamming Window* dengan Persamaan 2.2. Sinyal suara kemudian diubah ke dalam domain frekuensi dengan algoritma *Fast Fourier Transform* dengan Persamaan 2.3 lalu dikonversi ke dalam skala *mel* dan difilter dengan *Mel-filter bank* dengan jumlah 30, dimana frekuensi terendahnya 80 hz dan tertingginya 48000 hz. Kemudian sinyal suara akan dikembalikan ke dalam domain waktu dengan DCT. Setelah diambil 12 koeffisien *cepstral* lalu diekstrak energinya sebanyak 1 dimensi. Lalu total jumlah 13 dimensi yang sudah diekstrak akan dicari *delta* dan *delta-delta*-nya dengan Persamaan 2.6

sehingga total dimensi fitur MFCC yang diekstrak untuk setiap *frame* sinyal suara adalah sebanyak 39 dimensi. Proses berulang untuk semua *frame* pada semua sinyal suara yang diinputkan.

3.3 Reduksi dimensi

Berikut ini adalah diagram proses reduksi dimensi menggunakan algoritma PCA :



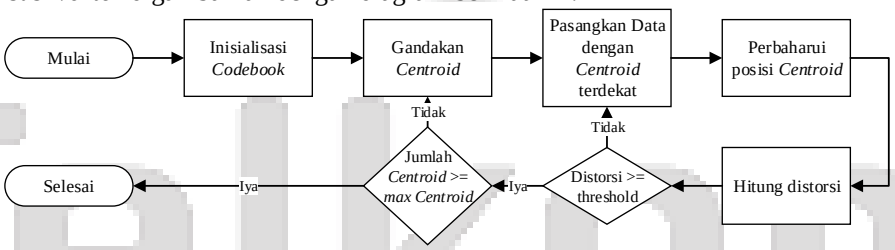
Gambar 3.3 Diagram Alur Proses Reduksi Dimensi Menggunakan Algoritma PCA

Data input untuk proses ini adalah semua fitur MFCC untuk semua *file* suara pada data latih. Data akan dinormalisasi dengan cara mengurangi semua data dengan rata-ratanya. Kemudian akan dicari nilai *covariance*-nya untuk semua dimensi data. Setelah itu dicari vektor *eigen* dan nilai *eigen*-nya dengan menggunakan *Eigenvalue Decomposition*. Nilai *eigen* akan menentukan kandungan informasi yang ada pada suatu dimensi, dikarenakan semakin tinggi nilai *eigen* maka data pada dimensi tersebut sangat bervariasi [**11**,**13**]. Vektor *eigen* akan diurutkan berdasarkan nilai *eigen*-nya. Pada tahap ini proses reduksi dimensi dilakukan dengan cara mengambil vektor *eigen* sesuai dengan dimensi data yang ingin dihasilkan, lalu vektor *eigen* akan dikalikan dengan data awal yang telah dinormalisasi. Data vektor *eigen* ini akan disimpan ke dalam *database* sistem untuk proses *pengujian*.

Data akustik vektor yang telah direduksi dimensinya pada proses reduksi dimensi ini yang akan digunakan untuk proses *vektor kuantisasi*.

3.4 Klasterisasi (Kuantisasi Vektor)

Proses *kuantisasi vektor* digambarkan dengan diagram berikut ini :



Gambar 3.4 Diagram Alur Proses Klasterisasi Dengan Algoritma LBG

Jenis HMM yang digunakan untuk proses *klasifikasi* adalah diskret HMM dimana pada jenis ini HMM hanya bisa menerima inputan berupa simbol – simbol diskret sehingga vektor akustik berupa kumpulan fitur MFCC yang telah dihasilkan proses *ekstraksi ciri* harus dikonversikan ke dalam simbol diskret dengan kuantisasi vektor menggunakan algoritma LBG.

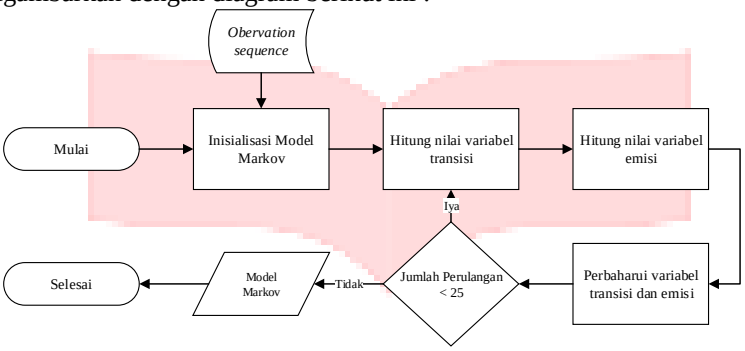
Pada proses *kuantisasi vektor* sistem akan mengambil inputan berupa kumpulan akustik vektor untuk semua data latih dan memprosesnya menjadi *codebook*. Pertama akan dicari *centroid* awal sebanyak satu yang didasarkan pada semua data latih yang ada. Lalu, *centroid* akan digandakan dengan parameter *split* sebesar 0,005. Setelah itu semua data akan dipasangkan dengan *centroid* terdekatnya menggunakan Persamaan 2.12. Posisi *centroid* akan diperbaharui dan dihitung distorsinya. Jika distorsi belum melebihi *threshold*-nya maka proses akan berulang lagi ke proses *Pasangkan Data dengan Centroid Terdekat*, jika jumlah *centroid* belum sesuai dengan yang diinginkan yaitu 256 maka proses akan berulang ke proses *Gandakan Centroid*. Data kumpulan *centroid* yang disebut dengan *codebook* ini akan disimpan ke dalam *database* untuk digunakan pada proses *pengujian*.

Setelah *codebook* dihasilkan maka setiap *frame* dalam akustik vektor atau fitur MFCC akan dikonversi menjadi simbol – simbol dengan cara memasangkannya dengan *centroid* terdekat, lalu diambil nomor urut *centroid* tersebut dalam kumpulan *centorid* pada *codebook*. Proses ini akan menghasilkan kumpulan *observation*

sequence untuk setiap *file* suaranya yang akan digabungkan untuk semua kelas yang sama (kata yang sama) untuk melatih model kata pada proses *klasifikasi.*

3.5 Klasifikasi

Proses *klasifikasi* digambarkan dengan diagram berikut ini :



Gambar 3.5 Diagram Alur Proses Klasifikasi

Pada proses *pelatihan* model kata akan dilatih dengan cara menginputkan *observation sequence* atau kumpulan simbol – simbol dari proses kuantisasi vektor untuk kata yang sama, kemudian dilatih dengan menggunakan algoritma *forward-backward.* Tiap iterasinya akan dihitung *alpha* dan *beta*-nya yang akan menjadi salah satu komponen untuk melatih parameter emisi dan transisi untuk model kata tersebut. Proses diulangi sebanyak 25 kali dan diulangi untuk masing – masing kata sampai semua model kata yang akan dilatih terbentuk.

Pada proses pengujian akurasi dihitung dengan cara menginputkan *obervasion sequence* dengan semua model kata yang telah dilatih ke dalam sistem dan dihitung berapa probabilitas *observation sequence* tersebut dihasilkan oleh model kata tersebut menggunakan algoritma *Viterbi*. Hasil probabilitas tersebut akan dibandingkan dengan semua probabilitas model kata yang ada pada sistem. Probabilitas tertinggi yang akan menjadi hasil akhirnya.

3.6 Skenario Pengujian

Pengujian dilakukan dengan cara membagi data menjadi beberapa kelas antara dua sampai sepuluh kelas. Masing – masing kelas merepresentasikan jumlah kata yang dilatih pada sistem. Sistem dua kelas akan dilatih dengan kata “Aceh” dan “Bandung” sedangkan sistem tiga kelas akan dilatih dengan kata “Aceh”, “Bandung”, dan “Bengkulu” begitu juga seterusnya.

Pengurangan dimensi dilakukan dengan cara mengurangi dimensi fitur MFCC dari *default*-nya sebesar 39 dimensi ke 34, 29, 24, 19, 14, 9, 5, 4, 3, 2 dan 1 dimensi.

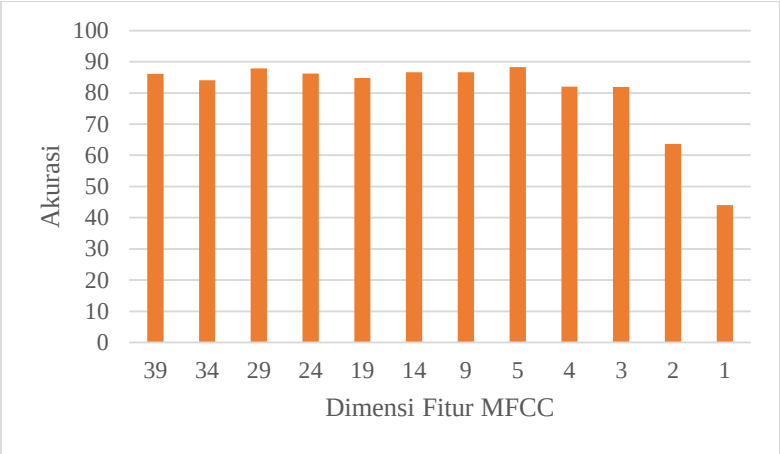
Perhitungan akurasi dilakukan dengan cara menjumlahkan kata yang berhasil dikenali dengan benar dibagi dengan jumlah kata yang diujikan.

Pengujian performansi dilakukan dengan cara menghitung selisih waktu dalam milisekon ketika program mulai, setelah proses *ekstraksi ciri* selesai, setelah proses *reduksi dimensi* selesai, setelah proses *kuantisasi vektor* selesai dan setelah proses *klasifikasi* selesai. Perhitungan waktu diulang sebanyak lima kali untuk proses yang sama dan dihitung rata – ratanya.

4. Evaluasi

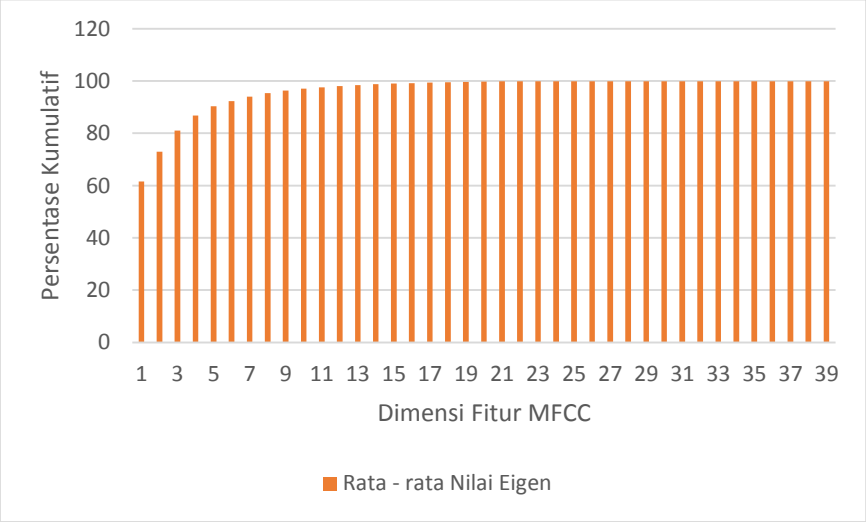
4.1 Hasil Pengujian

a. Akurasi sistem dengan berbagai jumlah dimensi fitur MFCC



Gambar 4.1 Rata – Rata Akurasi Sistem Terhadap Dimensi Fitur MFCC

b. Persentase kumulatif jumlah pesebaran data berdasarkan rata-rata nilai eigen pada semua kelas dataset



Gambar 4.2 Persentase Kumulatif Jumlah Nilai Eigen

c. Tabel performansi sistem untuk proses pelatihan dan pengujian

4-1 Tabel Waktu Rata-rata Proses Pelatihan dan Pengujian Untuk Sistem Dengan 6 Kelas

Total Waktu	Proses		Persentase Peningkatan Performansi (Terhadap 39 Dimensi)	
Jumlah Dimensi	Pelatihan (s)	Pengujian (s)	Pelatihan (%)	Pengujian (%)
39	103,19	1,33	0	0
34	101,21	1,25	1,92	6,02
29	100,16	1,28	2,94	3,76
24	100,1	1,25	2,99	6,02
19	99,79	1,32	3,29	0,75
14	100,39	1,25	2,71	6,02
9	99,94	1,3	3,15	2,26
4	99,85	1,24	3,24	6,77
1	99,84	1,25	3,25	6,02

4.2 Analisis Hasil Pengujian

Dari pengujian di atas, terlihat bahwa sistem sudah mampu mengenali ucapan dengan baik dengan rata-rata akurasi sistem sebesar 80,19%. Pada Tabel Akurasi Sistem Lengkap di Lampiran 2 terlihat bahwa persentase tertinggi pada sistem tanpa pengurangan dimensi fitur MFCC ada pada sistem dengan 2 dan 3 kelas dengan akurasi sebesar 100% dan akurasi terendahnya ada pada sistem dengan 7 kelas dengan 71,42%. Rata – rata untuk sistem tanpa pengurangan dimensi fitur MFCC adalah sebesar 86,1%. Berdasarkan hasil akurasi sistem tersebut dapat diketahui bahwa algoritma MFCC untuk ekstraksi ciri; LBG untuk kuantisasi vektor; dan HMM untuk klasifikasi sudah sangat baik untuk digunakan sebagai algoritma dalam pembangunan sistem pengenalan ucapan.

Berdasarkan grafik pada Gambar 4.1 terlihat bahwa akurasi sistem cenderung konstan pada akurasi lebih dari 80%, hingga jumlah fitur MFCC berada pada 2 dimensi ke bawah terjadi penurunan akurasi masing - masing sekitar 20%. Ini dikarenakan berdasarkan rata – rata nilai *eigen* untuk semua *dataset* pada grafik di Gambar 4.2 terlihat bahwa 81% data berkumpul pada tiga dimensi pertama fitur MFCC. Maka dari itu pengurangan dimensi hingga dimensi berjumlah di bawah tiga akan berakibat data tersisa sebanyak 72% untuk 2 dimensi dan 61% untuk 1 dimensi. Kehilangan banyak data berakibat pada penurunan akurasi sistem secara drastis. Sehingga penggunaan fitur MFCC dengan jumlah dimensi 2 dan 1 tidak direkomendasikan.

Performansi sistem secara keseluruhan tidaklah berubah secara drastis, peningkatan performansi rata - rata adalah sebesar 2,9% untuk proses pelatihan dan 4,7% untuk proses pengujian. Ini dikarenakan hanya proses *kuantisasi vektor* yang mengalami jumlah penurunan jumlah data untuk dikalkulasi, pada proses *klasifikasi* akustik vektor sudah diubah ke dalam bentuk *obervation sequence* sehingga tidak terjadi perubahan data baik sebelum pengurangan dimensi maupun sesudahnya. Selain itu proses pengurangan dimensi menambah beban komputasi sistem rata-rata sebesar 293,5 milisekon pada proses *pelatihan*.

5. Kesimpulan

Berdasarkan sejumlah pengujian yang telah dilakukan, metode MFCC, LBG dan HMM sudah sangat baik untuk digunakan dalam pembangunan sistem pengenalan ucapan untuk inputan data *isolated-word* atau satu kata per waktu. Pengurangan jumlah dimensi pada fitur MFCC berpengaruh terhadap akurasi sistem. Ini dikarenakan jumlah dimensi fitur MFCC berpengaruh terhadap jumlah data yang merepresentasikan suatu kata pada sinyal suara. Jumlah dimensi yang optimal untuk menghasilkan sistem berakurasi paling tinggi namun tetap memiliki performansi yang tinggi adalah fitur MFCC dengan jumlah 5 dimensi. Dimana pada 5 dimensi pertama ini fitur MFCC sudah mencakup 90% dari total data yang ada pada fitur MFCC berjumlah 39 dimensi. Jumlah dimensi paling tinggi tidak menjamin bahwa akurasi akan paling tinggi. Ini dikarenakan tidak semua variabel yang dapat diekstrak pada proses *ekstraksi ciri* akan berguna untuk proses pengenalan kata. Penggunaan fitur MFCC dengan jumlah dimensi 2 dan 1 tidak direkomendasikan, dikarenakan pada jumlah dimensi ini fitur MFCC sudah kehilangan terlalu banyak data sehingga berakibat pada penurunan akurasi sistem secara drastis.

Pengurangan dimensi fitur MFCC tidak meningkatkan performansi secara signifikan, ini dikarenakan hanya proses *kuantisasi vektor* yang mengalami penurunan jumlah data untuk dikalkulasi, sedangkan pada proses *klasifikasi*, data sudah diubah ke dalam bentuk *obervation sequence* sehingga tidak terjadi perubahan data sebelum pengurangan maupun sesudah pengurangan dimensi pada fitur MFCC. Selain itu, proses *reduksi dimensi* menambah beban sistem yang sebelumnya tidak ada jika tidak dilakukan reduksi dimensi.

Daftar Pustaka

[1] Santosh K. Gaikwad, Bharti W. Gawali, dan Yannwar Pravin. 2010. "A review on speech recognition technique". In *International Journal of Computer Applications (0975 - 8887)* on pp. 16-18.

[2] Wisesty U N, Adiwijaya , dan Astuti W. 2015. "Feature extraction analysis on Indonesian speech recognition system". In *Information and Communication Technology (ICoICT), 2015 3rd International Conference* on pp. 54-58. IEEE.

[3] Wisesty U N, Adiwijaya , dan Thee H L. 2012. "Indonesian speech recognition system using Discriminant Feature Extraction – Neural Predicitive Coding (DFE-NPC) and Probabilistic Neural Network". In *Computational Intelligence and Cybernetics (CyberneticsCom), 2012 IEEE International Conference* on pp. 158-162. IEEE.

[4] Wisesty U N, Mubarok M S, dan Adiwijaya. 2017. "A classification of marked hijaiyah letters' pronunciation using hidden Markov model". In *AIP Conference Proceedings 1867*

[5] Yulita I N, Houw Liong The, dan Adiwijaya. 2012. Fuzzy Hidden Markov Models for Indonesian speech classification. JACIII. 3:13 381–387.

[6] Adiwijaya , Aulia M. N., Mubarok M. S., Wisesty N. N., dan Nhita F.. 2017. "A Comparative Study of MFCC-KNN and LPC-KNN for Hijaiyyah Letters Pronounciation Classification System". In *IEEE 2017 5th International Conference* IEEE.

[7] Aziz R. A., Mubarok M. S., dan Adiwijaya. 2016, September. "Klasifikasi topik pada lirik lagu dengan metode multinomial naive bayes". In *Indonesia Symposium on Computing (IndoSC) 2016*

[8] Chadawan Ittichaichareon , Siwat Suksri , dan Thaweesak Yingthawornsuk. 2012. "Speech Recognition using MFCC". In *International Conference on Computer Graphics, Simulation and Modeling (ICGSM 2012).* on pp. 135-138.

[9] Huang Feng-Long. 2011. "A novel method for speaker independent recognition based on Hidden Markov

Model". In *ACEEE Int. J. on Signal & Image Processing* on pp. 27-30.

[10] Saleh Mohd Azha Mohd, Ibrahim Noor Salwani, dan Ramli Dzati Athiar. 2014. "Data reduction on MFCC features based on Kernel PCA for speaker verification system". In *Walia Journal* on pp. 56-62.

[11] Viszlay Peter , Pleva Matus , dan Juhar Jozef. 2011. "Dimension reduction with Principle Component Analysis applied to Speech Supervectors". In *Journal of Electrical and Electronics Engineering* on pp. 245-250.

[12] Suri Babu Korada, Anitha Y , dan Anjana.K.K.V.S. 2013. "Dimensionality Reduction in Feature Vector using Principle Component Analysis (PCA) for Effective Speaker Recognition". In *International Journal of Applied Information System (IJAIS)* on p. 15.

[13] Xuechuan Wang dan Douglas O'Shaughnessy. 2003. "Improving the Efficiency of Automaic Speech Recognition by Feature Transformation and Dimensional Reduction". In *EUROSPEECH 2003* on pp. 1025-1028.

[14] Roisin Loughran , Jacqueline Walker , Michael O'Neill , dan Marion O'Farrell. "The Use of Mel-frequency Cepstral Coefficients in Musical Instrument Identification"

[15] Pratiwi A. I. dan Adiwijaya. 2018. "On the feature selection and classification based on information gain for document sentiment analysis". In *Applied Computational Intelligence and Soft Computing 2018*

[16] Dipmoy Gupta , Radha Mounima C., Navya Manjunath , dan Manoj PB. 2012. "Isolated Word Speech Recognition Using Vector Quantization (VQ)". In *International Journal of Advanced Research in Computer Science and Software Engineering* on pp. 164-168.

[17] Adiwijaya. 2014. Aplikasi Matriks dan Ruang Vektor. Yogyakarta: Graha Ilmu.

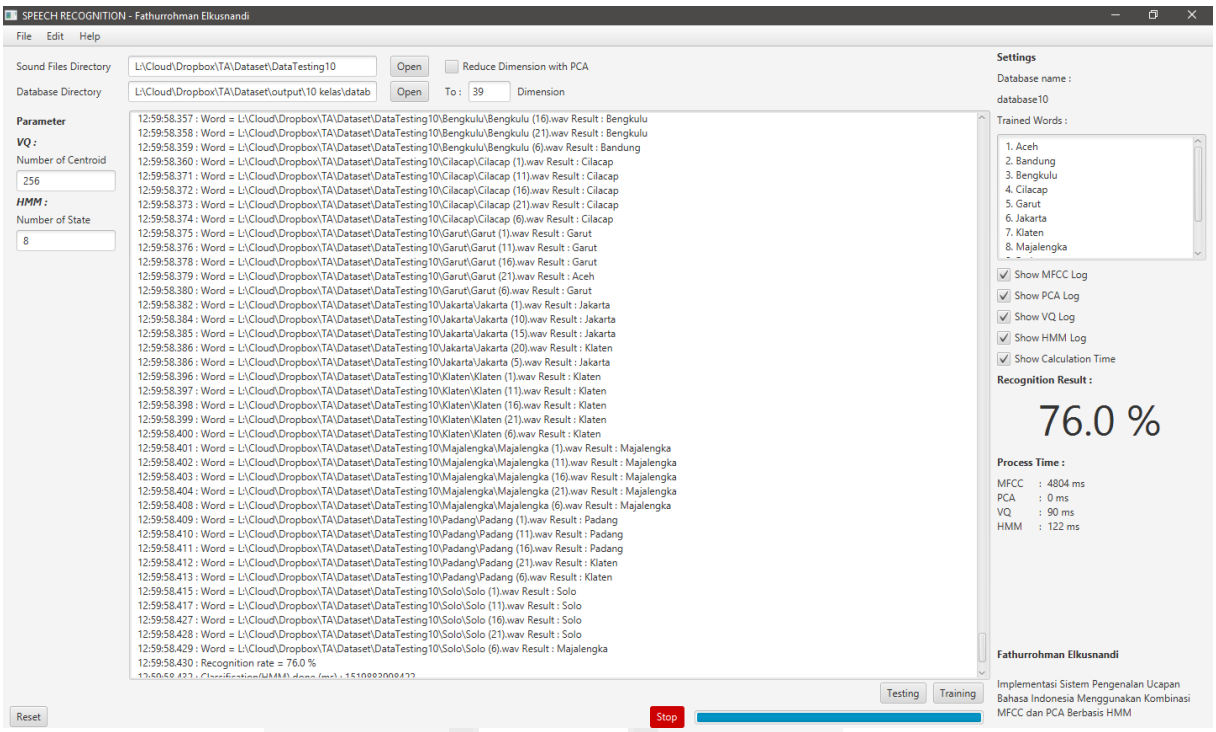
[18] Manjushre B. Aithal, Pooja R. Gaikwad, dan Shashikant L. Sahare. 2015. "Speech enhancement using PCA for speech and emotion recognition". In *Global Journal of Engineering, Design & Technology* on pp. 6-12.

[19] Adiwijaya. 2016. Matematika Diskrit dan Aplikasinya. Bandung: Alfabeta.

[20] Rabiner Lawrence R. 1989. "A tutorial on Hidden Markov Models and selected application in speech recognition". In *Procendings of the IEEE* on pp. 257-286.

Lampiran

1. Tampilan sistem



2. Tabel akurasi sistem lengkap

Akurasi	Kelas									
Dimensi MFCC	2	3	4	5	6	7	8	9	10	Rata-rata Akurasi Sistem
39	100	100	95	92	73,3	71,42	85	82,2	76	86,1
34	100	100	90	88	76,6	71,42	75	75,5	80	84,06
29	100	100	95	96	76,6	80	77,5	80	86	87,9
24	100	93,3	95	92	76,6	68,57	80	82,2	88	86,19
19	100	80	95	88	83,3	80	75	75,55	86	84,76
14	100	100	95	88	80	77,14	77,5	84,4	78	86,67
9	100	93,3	100	92	73,3	80	85	84,4	72	86,67
5	100	93,3	95	100	83,3	71,42	90	75,5	86	88,28
4	90	93,3	90	92	73,3	71,42	72,5	77,7	78	82,02
3	100	80	90	84	76,6	74,28	77,5	71,1	84	81,94
2	100	80	75	60	60	51,42	47,5	46,6	52	63,61
1	80	60	45	44	40	40	35	26,6	26	44,07
									Rata-rata	80,19%

3. Tabel performansi sistem pada proses *pelatihan* menggunakan data latih dengan 6 kelas

Waktu Proses (ms)	Proses				Total Waktu (s)	
No	Ekstraksi Ciri	Reduksi Dimensi	Kuantiasai Vektor	Klasifikasi	Pelatihan	Tanpa Ekstraksi Ciri
39	11659	0	856,2	90672	103,19	91,53
34	10128	301,2	650,8	90125,4	101,21	91,08
29	9651,2	301	571,2	89637,4	100,16	90,51
24	9585,4	247,4	467,8	89794,8	100,1	90,51
19	9378,2	226,8	363,4	89819	99,79	90,41
14	9810,4	336	297	89943,2	100,39	90,58
9	9368,2	290,6	250,2	90030,4	99,94	90,57
4	9560,4	238,6	187	89863,6	99,85	90,29
1	9575	385,6	113,8	89765,6	99,84	90,27

4. Tabel performansi sistem pada proses *pengujian* menggunakan data uji dengan 6 kelas

Waktu Proses (ms)	Proses				Total Waktu (s)	
No	Ekstraksi Ciri	Reduksi Dimensi	Kuantiasai Vektor	Klasifikasi	Pengujian	Tanpa Ekstraksi Ciri
39	1278,2	0	10,8	41,6	1,33	0,05
34	1054,4	162,2	7,8	27,6	1,25	0,2
29	1080	163,8	8,2	25	1,28	0,2
24	1056,4	158	7,6	23,8	1,25	0,19
19	1122,6	167,6	7,8	22,8	1,32	0,2
14	1056,8	165,8	7,2	21,6	1,25	0,19
9	1107,2	160,4	7,6	22	1,3	0,19
4	1051,4	157,2	8	20	1,24	0,19
1	1053,4	169	8,2	20,2	1,25	0,2