BAB 2
LANDASAN TEORI

2.1 Phonagnosia

Phonagnosia adalah gangguan neurologis di mana seseorang kehilangan kemampuan untuk mengenali suara-suara yang dikenal, khususnya suara manusia. Gangguan ini dapat bersifat sementara atau permanen dan seringkali terjadi pada orang yang sebelumnya tidak memiliki masalah pendengaran. Penderita phonagnosia dapat memiliki kemampuan pendengaran yang normal, namun sulit mengenali suara-suara orang-orang yang dikenalnya dan bahkan suara dirinya sendiri.

Phonagnosia disebabkan oleh kerusakan pada area otak yang terkait dengan pemrosesan suara dan pengenalan suara. Penyebab kerusakan tersebut dapat bervariasi, seperti cedera otak, stroke, tumor otak, atau penyakit neurologis seperti Alzheimer. Namun, terdapat juga kasus phonagnosia yang bersifat bawaan, yaitu terjadi sejak lahir.

Berdasarkan penelitian dalam jurnal *Brain and Language* memperkirakan bahwa lebih dari tiga persen orang dilahirkan dengan keadaan phonagnosia[13].

2.2 Speaker Recognition

Pengenalan pembicara dapat didefinisikan sebagai sebuah metode otomatis dalam mengenali siapa yang berbicara dengan memanfaatkan data-data pembeda pada gelombang suara. Metode pengenalan pembicara dapat diklasifikasikan menjadi 2 jenis, identifikasi pembicara dan verifikasi pembicara. Identifikasi pembicara adalah metode untuk mengenali identitas pembicara yang model cirinya terdapat pada basis data, ketika mengucapkan sesuatu. Sedangkan verifikasi pembicara adalah metode untuk menerima atau menolak identitas yang diakui oleh pembicara. Perbedaan mendasar antara kedua metode tersebut adalah jumlah alternatif keputusan. Pada metode identifikasi pembicara, jumlah alternatif keputusan sama dengan jumlah pembicara sedangkan jumlah alternatif keputusan pada metode verifikasi pembicara hanya ada dua alternatif keputusan, diterima atau ditolak. Sehingga performa identifikasi pembicara menurun ketika jumlah pembicara bertambah, sedangkan performa verifikasi pembicara konstan, tidak bergantung pada jumlah pembicara. Terdapat dua tipe dari Speaker Identification, yaitu text dependent (pembicara diberikan kata tertentu untuk diucapkan) dan text independent (pembicara dikenali berdasarkan kata-kata yang diucapkan) [3].

2.3 Uji Kecukupan Data

Langkah pertama dilakukan uji kecukupan data berfungsi untuk mengetahui apakah data yang diperoleh sudah mencukupi. Sebelum dilakukan uji kecukupan data terlebih dahulu menentukan derajat ketelitian adalah 5% (s = 0,05) yang menunjukkan penyimpangan maksimum hasil penelitian. Selain itu juga ditentukan tingkat kepercayaan 95% dengan k = 2 yang menunjukkan besarnya keyakinan pengukur akan ketelitian data. Jika banyaknya pengamatan yang sudah dilakukan ($N' \leq$ N), berarti pengamatan yang sudah dilakukan telah

memenuhi syarat, jika banyaknya pengamatan yang dilakukan ($N' \geq$ N) berarti banyaknya pengamatan yang sudah dilakukan belum memenuhi syarat, sehingga harus dilakukan pengamatan tambahan[14].

Pengujian kecukupan data dapat dihitung dengan persamaan berikut:

$$N' = \left[\frac{\frac{k}{s}\sqrt{N\sum xi^2 - (\sum xi)^2}}{\sum xi}\right]^2 \quad \textbf{(2.1)}$$

Dimana:

N' = Jumlah pengamatan yang seharusnya dilakukan
X = Data hasil pengukuran
S = Tingkat ketelitian yang dikehendaki (dalam decimal)
K = Koefisien indeks tingkat kepercayaan, yaitu:
Tingkat kepercayaan 0 % - 68 % harga k adalah 1
Tingkat kepercayaan 69 % - 95 % harga k adalah 2
Tingkat kepercayaan 96 % - 100 % harga k adalah 3

2.4 Uji Keseragaman Data

Uji keseragaman data adalah uji yang digunakan dalam suatu perancangan untuk melihat apakah data tersebut seragam atau tidak sehingga bisa dilanjutkan ke tahap berikutnya. Mean adalah sebuah rata-rata dari data yang diperoleh berupa angka dengan jumlah nilai-nilai dibagi dengan jumlah individu.

Rumus Mean:

$$\bar{X} = \frac{\sum_{i=1}^{n} Xi}{n} \quad \textbf{(2.2)}$$

Standar deviasi adalah suatu ukuran yang menggambarkan tingkat penyebaran data dari nilai rata – rata. Rumus standar deviasi yaitu:

$$SD = \sqrt{\frac{\sum(xi - \bar{X})^2}{n-1}} \quad \textbf{(2.3)}$$

Perhitungan BKA dan BKB:

$$BKA = \bar{X} + (2\ x\ SD) \quad \textbf{(2.4)}$$

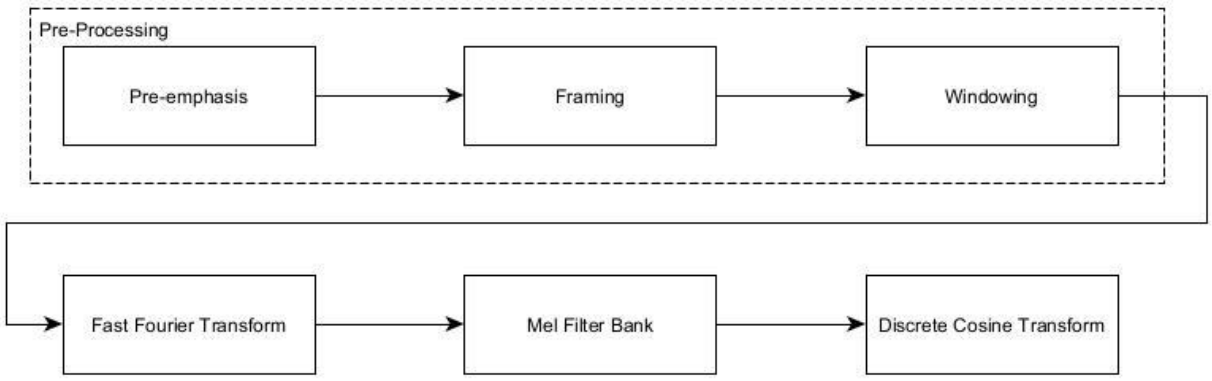
$$BKB = \bar{X} - (2\ x\ SD) \quad \textbf{(2.5)}$$

2.5 Mel Frequency Cepstrum Coefficient (MFCC)

MFCC yang menghitung koefisien cepstrum dengan mempertimbangkan persepsi sistem pendengaran manusia terhadap frekuensi suara. MFCC didasarkan pada variasi yang telah diketahui dari jangkauan kritis telinga manusia dengan frekuensi. MFCC memiliki 2 jenis filter dimana bersifat linear pada frekuensi dibawah 1000 Hz dan bersifat logaritmic pada frekuensi diatas 1000 Hz [7]. Beberapa keunggulan dari metode ini adalah [7]:

a. Mampu untuk menangkap karakteristik suara yang sangat penting bagi pengenalan suara, atau dengan kata lain dapat menangkap informasi-informasi penting yang terkandung dalam sinyal suara.
b. Menghasilkan data seminimal mungkin, tanpa menghilangkan informasi-informasi penting yang dikandungnya.
c. Mereplikasi organ pendengaran manusia dalam melakukan persepsi terhadap signal suara. (Darma, dkk, 2011).

Adapun gambar alur dari proses mel frequency cepstrum coefficients (MFCC) dapat dilihat pada gambar 2.1[12]:



Gambar 2. 1 Alur Proses MFCC

2.3.1 Pre-Emphasis

Pre-emphasis berfungsi untuk menstabilkan nilai magnitude dari sinyal suara [2]. Pada penelitian sebelumnya α = 0.95 [15]. Adapun persamaan dari pre-emphasis adalah sebagai berikut [15]:

$$\boldsymbol{Sa[n] = s[n] - \alpha . s[n-1]} \quad \textbf{(2.6)}$$

Dimana:

$Sa[n]$ = Hasil dari *pre-emphasis*

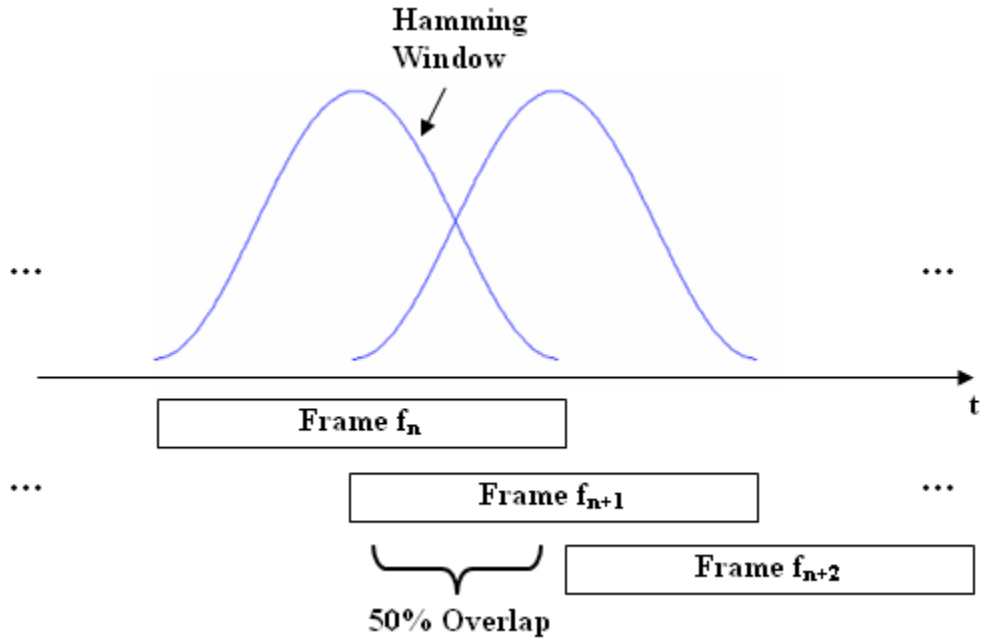
$s[n]$ = Sampel sebelum dilakukan pre-emphasis

α = Konstanta *pre-emphasis* = 0.95

2.3.2 Framing

Framing berfungsi membagi sinyal suara menjadi beberapa frame dengan panjang sampel tertentu. Panjang tiap frame umumnya sangat singkat sekitar 20 sampai 40 ms [2]. Tujuan dari

framing adalah untuk menghindari diskontinuitas sinyal, Karena diskontinuitas sinyal dapat menyebabkan ekstraksi parameter yang salah selama analisis. Untuk menghindari diskontinuitas sinyal pada dua frame berturut-turut, setiap dua frame saling tumpang tindih (overlapping) [15]. Pada penelitian sebelumnya telah dilakukan framing 20ms, overlaping 10ms [15]. Proses dari framing dapat dilihat pada gambar 2.2[15]:



Gambar 2. 2 Proses Framing

Adapun persamaan dari *framing* adalah sebagai berikut [15]:

Ns = *sample rate* x durasi *framing* **(2.7)**

Dimana:

Ns = Jumlah dari *sample point*

Durasi *framing* = 20ms (0.020)

No = *sample rate* x durasi *overlapping* **(2.8)**

Dimana:

No = Jumlah dari *overlapping*

Durasi *overlapping* = 10ms (0.010)

$$Nf = durasi\ x\ \frac{sample\ rate}{Ns} \quad \textbf{(2.9)}$$

Dimana:

Ns = Jumlah *sample point*

Nf = jumlah *frame rate*

2.3.3 Windowing

Konsep dari proses windowing adalah meruncingkan sinyal ke angka nol pada permulaan dan akhir setiap frame [8]. Adapun persamaan dari windowing adalah sebagai berikut [15]:

$$\boldsymbol{Wham[n] = 0,54 - 0,46cos\left(\frac{2\pi n}{Ns-1}\right), 0 \leq n \leq Ns - 1} \quad \textbf{(2.10)}$$

Dimana:

$Wham[n]$ = Hasil dari *Hamming Windowing*

Ns = jumlah *sampel point*

Selanjutnya menghitung windowing dengan persamaan berikut [15]:

$$\boldsymbol{Sb[n] = Sa[n]\ X\ Wham[n],\ 0 \leq n \leq Ns - 1} \quad \textbf{(2.11)}$$

Dimana:

$Wham[n]$ = Hasil dari *Hamming Windowing*

$Sa[n]$ = Sampel setelah dilakukan *pre-emphasis*

$Sb[n]$ = Hasil dari *Windowing*

2.3.4 Fast Fourier Transform (FFT)

Tujuan utama dari transformasi fourier ini adalah untuk mengubah sinyal dari domain waktu menjadi spektrum pada domain frekuensi. Fast Fourier Transform (FFT) merupakan algoritma perhitungan Discrete Fourier Transform (DFT) yang efisien sehingga akan mempercepat proses perhitungan DFT [4]. Adapun persamaan FFT sebagai berikut [15]:

$$\boldsymbol{F[k] = \sum_{n=0}^{Ns-1}\left(Sb[n]cos\frac{2\pi nk}{Ns}\right) - j\sum_{n=0}^{Ns-1}\left(Sb[n]sin\frac{2\pi nk}{Ns}\right), k = 0, 1, 2, \dots, Ns - 1} \quad \textbf{(2.12)}$$

Dimana:

$F[k]$ = Hasil dari FFT

$Sb[n]$ = sampel setelah dilakukan *windowing*

Persamaan (2.12) digunakan untuk mendapatkan hasil dari FFT.

$$Sc[n] = \left|[R^2 + I^2]^{\frac{1}{2}}\right| \quad \textbf{(2. 13)}$$

2.3.5 Mel Frequency Filter Bank

Skala mel-frekuensi adalah pemetaan frekuensi secara liniear untuk frekuensi di bawah 1 kHz dan logaritmik untuk frekuensi di atas 1 kHz. Sebagai titik referensi, pitch dari 1 kHz, 40 dB diatas perceptual hearing threshold, didefinisikan sebagai 1000 mels. Pada penelitian sebelumnya jumlah Filter Bank 30, Low mel frequency 130Hz, High mel frequency 6800Hz [15]. Adapun persamaan mel filter bank sebagai berikut [15]:

$$\boldsymbol{mel(f) = 2595 * log_{10}\left(1 + \frac{f}{700}\right)} \quad \textbf{(2.14)}$$

Dimana:

$mel(f)$ = Hasil *mel filter*

$$\boldsymbol{mel^{-1}(f) = 700\left(10^{\frac{f}{2595}} - 1\right)} \quad \textbf{(2.15)}$$

Dimana:

$mel^{-1}(f)$ = Hasil inverse *mel filter*

$$\boldsymbol{f_{b(i)} = \left(\frac{Ns}{Fs}\right) mel^{-1}\left(mel(Nlow) + i\,\frac{mel(Nhigh) - mel(Nlow)}{Nfbank - 1}\right)} \quad \textbf{(2.16)}$$

Dimana:

$f_{b(i)}$ = Hasil *boundary point filter*

$Nlow$ = Jumlah mel terendah 130Hz

$Nhigh$ = Jumlah mel tertinggi 6800Hz

$$\boldsymbol{h(i,k) = \begin{cases} 0, & untuk\ k < f_{b(i-1)} \\ \dfrac{k - f_{b(i-1)}}{f_{b(i)} - f_{b(i-1)}}, & untuk\ f_{b(i-1)} \le k < f_{b(i)} \\ \dfrac{f_{b(i+1)} - k}{f_{b(i+1)} - f_{b(i-1)}}, & untuk\ f_{(i)} \le k < f_{b(i+1)} \\ 0, & untuk\ k > f_{b(i+1)} \end{cases}} \quad \textbf{(2. 17)}$$

Dimana:

$f_{b(i)}$ = Hasil *boundary point filter*

$h(i,k)$ = Koefisien dari *mel filter bank*

$$\boldsymbol{Sd[i] = \sum_{k=0}^{Ns-1} |Sc[i]|^2 \; x \; h(i,k), i = 1,2 \ldots Nfbank} \qquad \textbf{(2.18)}$$

Dimana:

$Sc[i]$ = Sampel setelah dilakukan FFT

$Sd[i]$ = Hasil *Mel Filter Bank*

$h(i,k)$ = Koefisien dari *mel filter bank*

$Nfbank$ = Jumlah *Mel Filter Bank* 30

2.3.6 Discrete Cosine Transform (DCT)

Ini adalah proses untuk mengubah log Mel spectrum menjadi domain waktu menggunakan Discrete Cosine Transform (DCT). Hasil konversi tersebut disebut Mel Frequency Cepstrum Coefficient. Himpunan koefisien disebut vektor akustik. Oleh karena itu, setiap ucapan masukan ditransformasikan menjadi urutan vektor akustik [15]. Adapun persamaan DCT sebagai berikut[16]:

$$\boldsymbol{Se[n] = \sqrt{\frac{2}{Nbank}} \sum_{k=0}^{Nbank-1} \log(Sd[k+1]) \; x \cos\left[n\left(\frac{2k-1}{2}\right)\frac{\pi}{Nfbank}\right], n = 0,1,2, \ldots Nk-1} \qquad \textbf{(2.19)}$$

Dimana:

$Se[n]$ = Hasil dari DCT

$Sd[k]$ = Sampel setelah dilakukan *Mel Filter Bank*

Nk = Jumlah koefisien DCT = 12

$Nfbank$ = Jumlah filter bank

2.6 Learning Vector Quantization 3

LVQ adalah algoritma perhitungan saraf dengan kecerdasan komputasi, dengan menggunakan pembelajaran kompetitif untuk melakukan klasifikasi. LVQ mempelajari vektor fitur dari ekstraksi ciri yang dihasilkan oleh metode MFCC [15]. Algoritma LVQ3 merupakan pengembangan dari LVQ 2.1. Pada LVQ 2.1 dimana vektor pewakil kemungkinan mengalami divergensi selama proses pembelajaran dilakukan. Sedangkan pada LVQ 3 koreksi dilakukan terhadap LVQ 2.1, dimana untuk memastikan vektor pewakil agar selalu mendekati distribusi dari kelas [17]. Berikut adalah tahapan-tahapan pada algoritma pelatihan LVQ3 [17]:

1. Inisialisasi bobot w dan x.
2. Tentukan nilai learning rate α. Nilai α adalah $0 < \alpha(t) < 1$.
3. Nilai pengurangan learning rate α.

   $\alpha = \alpha - (\alpha \; x \; dec \; \alpha)$ **(2.20)**
4. Tentukan nilai minimum learning rate α (min α).
5. Tentukan nilai window (ε) dan epsilon (m).
6. Hitung jarak (D) antara vector masukan (X) dan vector bobot (W)

menggunakan persamaan:

$D = \sqrt{\sum(X - W)^2}$ **(2.21)**

7. Tentukan i, yaitu indeks jarak terdekat pertama (Di) dan j indeks jarak terdekat kedua (Dj).
8. Tentukan kondisi window Di dan Dj menggunakan persamaan:

   $\min\left[\frac{Di}{dj}, \frac{Dj}{Di}\right] > \frac{(1-\varepsilon)}{(1+\varepsilon)}$ **(2.22)**
9. Ubah bobot (w) dengan ketentuan:
   a. Jika kondisi bernilai benar, maka ubah bobot w menggunakan persamaan:

      $mi(t+1) = mi(t) - \alpha(t)\,[x(t) - mi(t)]$ **(2.23)**

      $mj(t+1) = mj(t) + \alpha(t)\,[x(t) - mj(t)]$ **(2.24)**
   b. Jika kondisi bernilai salah, maka ubah bobot w menggunakan persamaan:

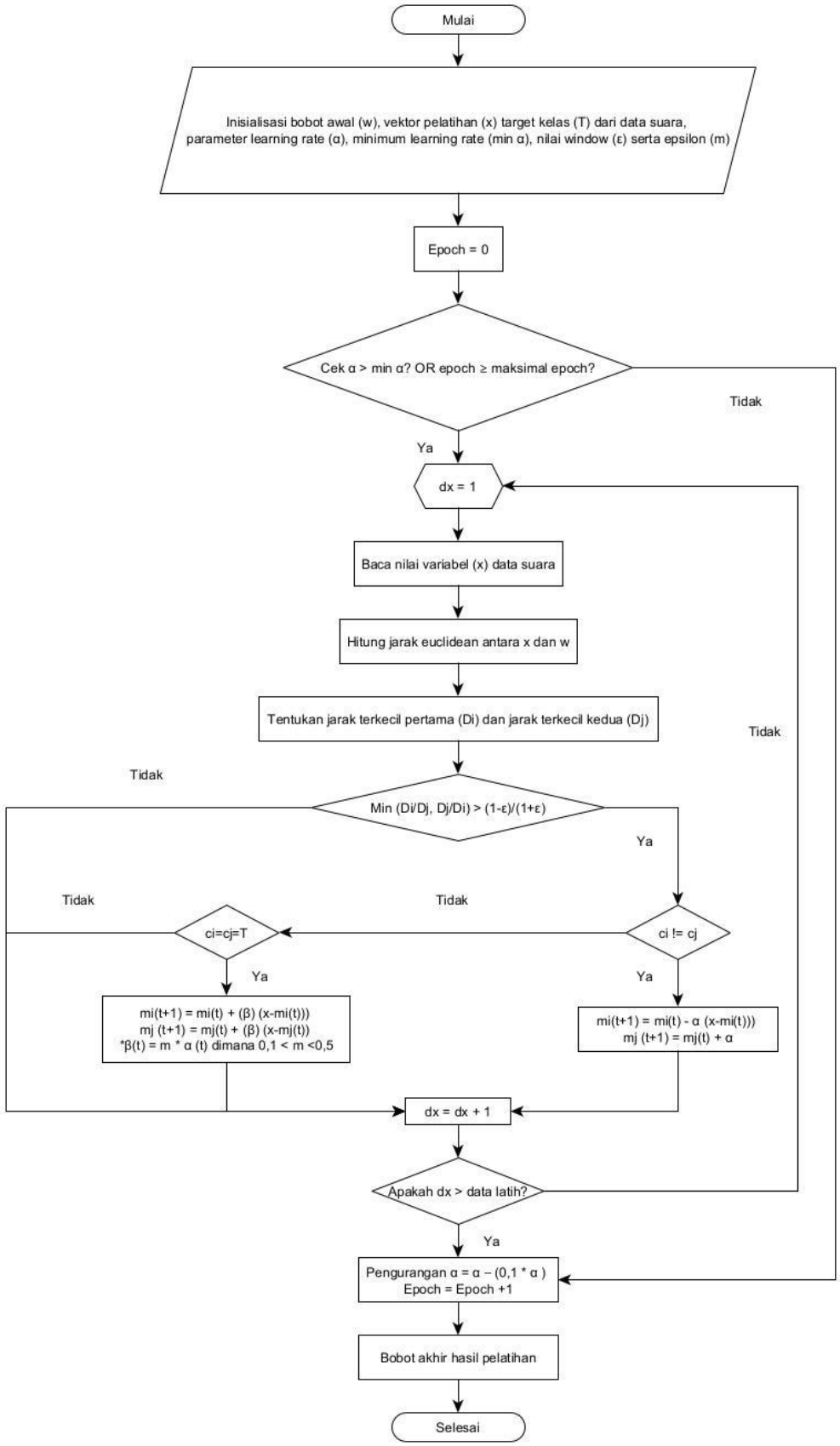
      $mi(t+1) = mi(t) + \beta(t)\,[x(t) - mi(t)]$ **(2.25)**

      $mj(t+1) = mj(t) + \beta(t)\,[x(t) - mj(t)]$ **(2.26)**

      dengan $\beta(t) = m\alpha(t)$, dimana $0{,}1 < m < 0{,}5$

Dimana:
D = jarak
$\sum$ = jumlah keseluruhan
x = nilai x (data)
w = nilai bobot
ε = *window*
$m_i(t+1)$ = bobot baru pertama (w baru)
$m_i(t)$ = bobot lama pertama (w lama)
$m_j(t+1)$ = bobot baru kedua (w baru)
$m_j(t)$ = bobot lama kedua (w lama)
α = *learning rate*
$\beta(t)$ = perkalian antara epsilon dan learning rate
$m\alpha(t) = \varepsilon\alpha(t)$ [18]

Adapun flowchart proses pelatihan dan pengujian metode LVQ 3 sebagai berikut:


Mulai
Inisialisasi bobot awal (w), vektor pelatihan (x) target kelas (T) dari data suara, parameter learning rate (α), minimum learning rate (min α), nilai window (ε) serta epsilon (m)
Epoch = 0
Cek α > min α? OR epoch ≥ maksimal epoch?
Tidak
Ya
dx = 1
Baca nilai variabel (x) data suara
Hitung jarak euclidean antara x dan w
Tentukan jarak terkecil pertama (Di) dan jarak terkecil kedua (Dj)
Tidak
Tidak
Min (Di/Dj, Dj/Di) > (1-ε)/(1+ε)
Ya
Tidak
Tidak
ci=cj=T
ci != cj
Ya
Ya
mi(t+1) = mi(t) + (β) (x-mi(t)))
mj (t+1) = mj(t) + (β) (x-mj(t))
*β(t) = m * α (t) dimana 0,1 < m <0,5
mi(t+1) = mi(t) - α (x-mi(t)))
mj (t+1) = mj(t) + α
dx = dx + 1
Apakah dx > data latih?
Ya
Pengurangan α = α – (0,1 * α)
Epoch = Epoch +1
Bobot akhir hasil pelatihan
Selesai

Gambar 2. 3 Flowchart Proses Pelatihan

Berikut penjelasan dari Gambar 2.3 flowchart proses pelatihan menggunakan metode LVQ3[17]:

a. Inisialisasi Nilai

   Pada tahapan ini dilakukan pemberian nilai parameter yang diperlukan diantara nya nilai bobot awal (w), vektor pelatihan (x), target kelas (T), learning rate (α), minimal learning rate (min α), nilai window (ε).

b. Masuk ke *epoch* = 0

c. Periksa Nilai α

   Pada tahap ini jika nilai learning rate (α) > nilai minimal learning rate (min α) atau epoch ≥ maksimal epoch maka dilanjutkan ke proses selanjutnya. Sedangkan jika nilai learning rate (α) < nilai minimal learning rate (min α) atau epoch ≤ maksimal epoch maka didapat bobot akhir dari hasil pelatihan.

d. Masuk ke dx = 1

e. Lalu baca nilai variabel (x) data suara yang digunakan

f. Kemudian hitung jarak euclidean antara variabel (x) dan bobot awal (w) menggunakan Persamaan (2.21).

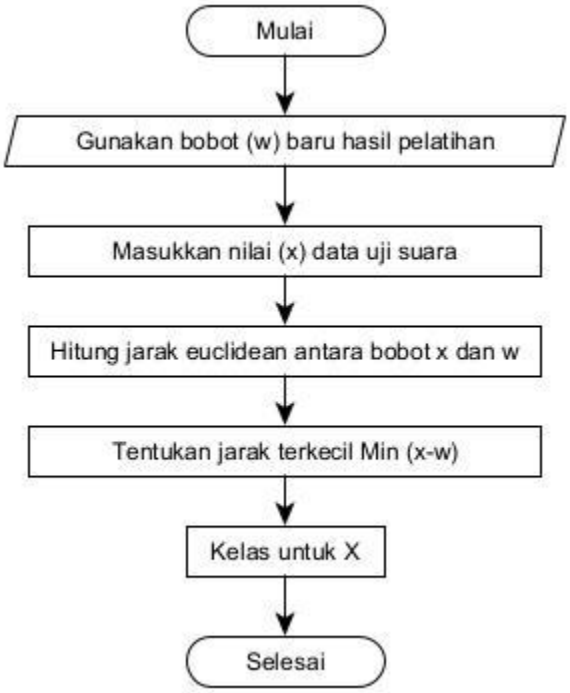
g. Tentukan jarak terkecil pertama (Di) dan jarak terkecil kedua (Dj) dilihat dari hasil perhitungan jarak euclidean.

h. Lalu cek $\min\left[\frac{Di}{dj}, \frac{Dj}{Di}\right] > \frac{(1-\varepsilon)}{(1+\varepsilon)}$, jika bernilai benar maka cek kondisi ci != cj jika bernilai benar maka terjadi perubahan bobot D_i dan D_j menggunakan Persamaan (2.23) dan Persamaan (2.24). Apabila kondisi ci !=cj bernilai salah maka dilakukan pengecekkan nilai ci dan cj = T jika bernilai benar maka lakukan perubahan bobot ci dan cj menggunakan Persamaan (2.25) dan Persamaan (2.26). Dan apabila pada pengecekkan nilai ci dan cj = T bernilai salah maka lakukan dx = dx + 1. Akan tetapi jika nilai $\min\left[\frac{Di}{dj}, \frac{Dj}{Di}\right] > \frac{(1-\varepsilon)}{(1+\varepsilon)}$ bernilai salah maka lakukan lakukan dx = dx + 1.

i. Setelah dilakukan dx = dx + 1 selanjutnya cek kondisi apakah dx > data latih?. Jika bernilai benar maka dilajutkan ke tahap selanjutnya. Jika bernilai salah maka balik ke proses dx = 1.

j. Lakukan pengurangan learning rate menggunakan Persamaan (2.20) dan penambahan epoch.
k. Bobot Akhir Pada tahap ini didapatkan hasil bobot akhir vektor w baru yang akan digunakan pada proses pengujian.

Setelah didapatkan bobot akhir maka dilanjutkan ke tahap pengujian metode LVQ3 seperti pada Gambar 3.4[17]:



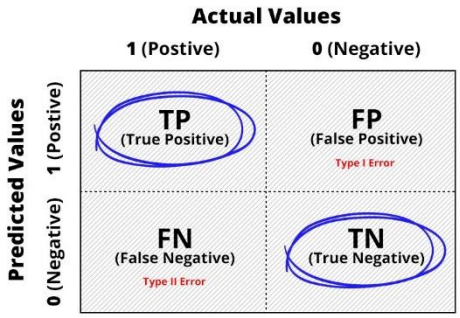
Gambar 2. 4 Flowchart Proses Pengujian

Berikut penjelasan dari Gambar 2.4 *flowchart* proses pengujian menggunakan metode LVQ3[17]:

a. Gunakan bobot (w) baru hasil pelatihan dari proses pembelajaran metode LVQ3.
b. Masukkan nilai (x) data uji suara yang akan dilakukan pengujian.
c. Lalu hitung jarak euclidean antara bobot x dan w menggunakan Persamaan (2.16).
d. Tentukan jarak terkecil.
e. Hasil akhir yaitu kelas untuk X.

2.7 Confusion Matrix

Confusion matrix atau error matrix digunakan untuk mengevaluasi kinerja dari algoritma biasanya digunakan pada supervised learning. Confusion matrix memberikan informasi perbandingan hasil klasifikasi yang telah dilakukan oleh sistem atau model dengan hasil sebenarnya. Confusion matrix berbentuk tabel matrik yang menampilkan hasil dari kinerja model dalam mengklasifikasikan data uji [19]. Gambar berikut ini merupakan contoh confusion matrix dengan empat kelas prediksi dan nilai aktual yang berbeda.



Gambar 2. 5 Confusion Matrix

Pada gambar 2.5 hasil dari proses klasifikasi pada confusion matrix. Terdapat empat pada tabel confusion matrik yaitu True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN). Berikut beberapa rumus yang diterapkan dalam mengevaluasi performa dari model:

2.7.1 Accuracy

Accuracy merupakan rasio antara prediksi benar (positif dan negatif) dengan keseluruhan data. Dengan kata lain, accuracy merupakan tingkat ketepatan nilai prediksi dengan nilai aktual (sebenarnya).

$$\boldsymbol{Akurasi} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{P+N}} \quad \textbf{(2.27)}$$

2.7.2 Precision

Precision merupakan rasio prediksi benar positif dengan keseluruhan hasil yang diprediksi positf. Dari semua kelas positif yang telah di prediksi dengan benar, berapa banyak data yang benar-benar positif. Nilai precision dapat diperoleh dengan persamaan.

$$\boldsymbol{Precission} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} \quad \textbf{(2.28)}$$

2.7.3 Recall

Recall merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Nilai recall dapat diperoleh dengan persamaan.

$$\boldsymbol{Recall = \frac{TP}{TP+FN}} \quad \textbf{(2.29)}$$

2.8 Perangkat Lunak Pembangun

2.8.1 Tensorflow

TensorFlow adalah perpustakaan perangkat lunak, yang dikembangkan oleh Tim Google Brain dalam organisasi penelitian Mesin Cerdas Google, untuk tujuan melakukan pembelajaran mesin dan penelitian jaringan syaraf dalam. TensorFlow kemudian menggabungkan aljabar komputasi teknik pengoptimalan kompilasi, mempermudah penghitungan banyak ekspresi matematis dimana masalahnya adalah waktu yang dibutuhkan untuk melakukan perhitungan[20].

2.8.2 Google Colab

Google colab merupakan sebuah tools yang dikeluarkan oleh google. Tools ini memberikan fasilitas kepada para peneliti atau orang yang ingin mempelajari dan mengolah data menggunakan machine learning maupun deep learning, namun memiliki keterbatasan perangkat untuk melakukan komputas. Google colab menyediakan layanan GPU gratis sebagai Backend komputasi yang dapat digunakan selama 12 jam[20].

Berikut adalah beberapa kelebihan dalam menggunakan google colab[21]:

- Free Access

  Penggunaan google colab ditunjukan bagi para peneliti yang sedang mengembangkan penelitian dan membutuhkan spesifikasi computer yang tinggi. Hayanya perlu diingat bahwa google colab membutuhkan koneksi internet

- Good Spesification

  Ketika kita pertama kali menginstall google colab maka akan diberikan akses cloud computer dengan spesifikasi:

  1. GPU Tesla
  2. RAM 12 GB
  3. HDD 300GB

- Easy Sharing

  Kita dapat melakukan integrasi dengan google drive milik kita dan kemudian menyimpan scrypt kedalam project github. Ataupun berbagi link dengan orang lain.

  1. Colaborate

google colab juga memudahkan kita berkolaborasi dengan orang bisa lebih mudah bereksperimen secara bersamaan, atau sekadar menggunakan fitur ini untuk mempelajari codingan orang lain yang telah rapi (karena format notebook).

2. Mudah berintegrasi

   google colab terbilang sangat fleksibel dalam hal integrasi. Kita Dapat dengan mudah menghubungkan google colab dengan notebook jupyter di computer kita (local runtime), menghubungkan dengan google drive, atau dengan github.

3. Fleksibel

Google Colab hanya perlu dijalankan di browser, sehingga dapat dengan mudah menjalankan program deep learning dari ponsel. Selama ponsel cerdas terhubung ke Google Drive yang sama, dapat memantaunya dari browser ponsel cerdas.