BAB II
LANDASAN TEORI

Dalam pembangunan sistem ini menggunakan berbagai teori-teroi untuk menunjang proses penelitian, teori-teori tersebut antara lain :

2.1. Tinjauan Pustaka

Dalam kegiatan penelitian ini ada beberapa pustaka yang berkaitan dengan kegiatan penelitian yang akan dilakukan, antara lain:

(Indrawaty N, Andriana, & Permatasari, 2017) melakukan penelitian dengan judul ini dilakukan untuk menerapkan metode MFCC untuk menghitung koefisien cepstral dalam pengekstraian sinyal ucapan. Penelitian ini juga menggunakan metode pengklasifikasian VQ (*Vector Quantization)* yang mengubah hasil ekstraksi analisis MFCC dari masing-masing pengucap menjadi sekumpulan *codebook. Codebook* tersebut kemudian dibandingkan dengan hasil ekstraksi koefisien MFCC dari sinyal masukan yang akan di kenali.

(Doanda Khabi Putra, 2017)Melakukan penelitian dengan judul *Simulasi Dan Analisis Speaker Recognition Menggunakan Metode Mel Frequency Cepstrum Coefficient (MFCC) dan Gaussian Mixture Model (GMM)* untuk meningkatkan ke akurasian nilai ekstraksi ciri MFCC dan membandingkan distribusi Gaussian dengan memanfaatkan parameter *mean* dan *variance.*

(Ezhar Mega Risondang, Risondang, Andrean, & Arie, 2019) penelitian dengan judul *Aplikasi Identifikasi Suara Hewan Menggunakan Metode Mel-Frequency Cepstral Coeffisients(MFCC)* yang bertujuan menggunakan teknik pengenalan suara untuk mendeteksi, mengidentifikasi dan menerjemahkan suara binatang. Sistem ini terdiri dari dua tahap yaitu pelatihan dan pengujian. Pelatihan melibatkan pengajaran sistem dengan membangun kamus, model akustik untuk setiap kata yang perlu dikenali oleh sistem (analisis *offline*). Tahap pengujian menggunakan model akustik untuk

mengenali kata-kata terisolasi menggunakan algoritma klasifikasi. Aplikasi penyimpanan audio untuk mengidentifikasi berbagai suara binatang dapat dilakukan dengan lebih akurat di masa depan.

(Bhaskoro & W.D, 2012) melakukan penelitian yang berjudul *Aplikasi Pengenalan Gender Menggunakan Suara* penelitian ini akan fokus pada identifikasi suara manusia berdasarkan jenis kelamin yang akan dilakukan oleh komputer dengan melakukan beberapa fase, diantaranya: (i) *training phase* (fase Pelatihan), pada fase ini menentukan beberapa penutur yang memiliki tugas untuk memberikan sampel suaranya, sehingga sistem dapat menyimpan referensi data pelatihan suara untuk model pembicara. (ii) *recognition phase* (fase pengenalan), pada fase ini mencoba mencocokkan sebuah suara yang akan diuji dengan model dari penyimpanan referensi sebelumnya dan selanjutnya membuat keputusan pengenalan terhadap suara yang diujikan.

(Budiman, Nursyeha, & Rivai, 2016) Melakukan penelitian yang berjudul *Pengenalan Suara Burung Menggunakan Mel Frequency Cepstrum Coefficient Dan Jaringan Syaraf Tiruan Pada Sistem Pengusir Hama Burung*. Penelitian ini bertujuan merancang perangkat lunak untuk sistem pengusir hama burung yang mendeteksi jenis hama burung berdasarkan kicau burung. Diharapkan dengan adanya perangkat lunak ini, metode pengusiran hama burung pada ekosistem sawah dapat berjalan dengan lebih efisien.

(Chamidy, 2016) melakukan penelitian yang berjudul *Metode Mel Frequency Cepstral Coeffisients (MFCC) Pada klasifikasi Hidden Markov Model (HMM) Untuk Kata Arabic pada Penutur Indonesia.* Penelitian ini bertujuan untuk mendapatkan tingkat kesesuaian metode yang diterapkan pada masukan sinyal suara pengucapan kata *arabic* pada penutur Indonesia. Selanjutnya, fitur yang telah diekstraksi, diklasifikasi menggunakan *Hidden Markov Model* (HMM).

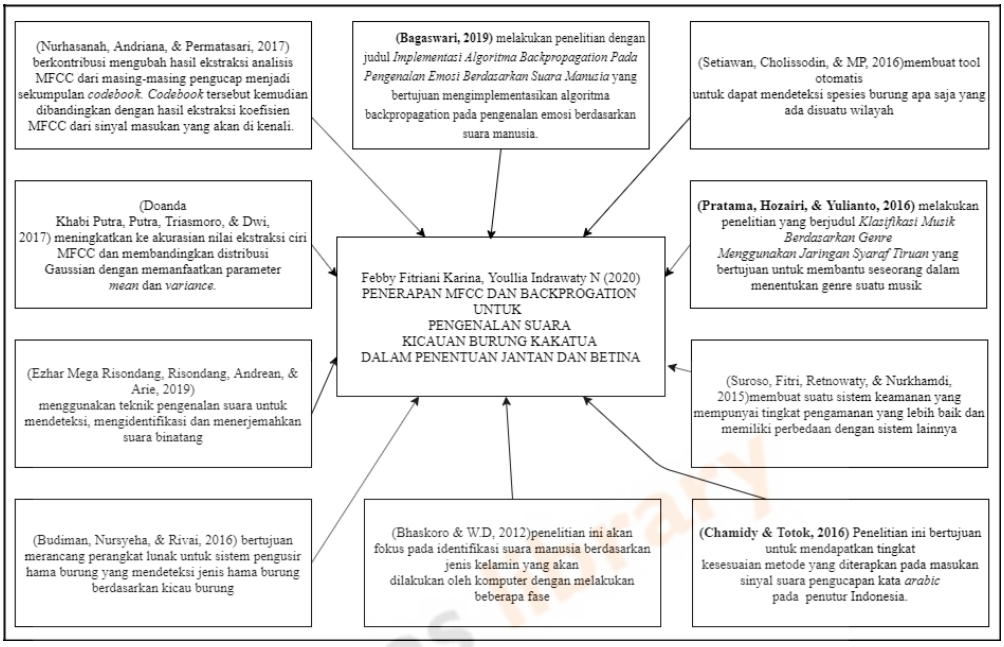
(Suroso, Fitri, Retnowaty, & Nurkhamdi, 2015) Pada penelitian ini berjudul *Aplikasi Pengenalan Ucapan Dengan Ekstraksi Ciri Mel-Frequency Cepstrum Coefficients (MFCC) dan Jaringan Syaraf Tiruan (JST) Propagasi Balik Untuk Buka dan Tutup Pintu*, penelitian ini bertujuan untuk membuat suatu sistem keamanan yang mempunyai tingkat pengamanan yang lebih baik dan memiliki perbedaan dengan sistem lainnya. Hal tersebut disebabkan karena sistem ini menggunakan pengenalan ucapan dengan karakteristik tersendiri, yaitu dengan ucapan "Buka"dan "Tutup" untuk pintu. Maka, dalam hal ini DSP digunakan sebagai aplikasi pengenalan ucapannya. Dimana ucapan "Buka" dan "Tutup" akan diekstrasi ciri menggunakan MFCC dan dikenali dengan JST Propagasi Balik.

(Pratama, Hozairi, & Yulianto, 2016) melakukan penelitian yang berjudul *Klasifikasi Musik Berdasarkan Genre Menggunakan Jaringan Syaraf Tiruan* yang bertujuan untuk membantu seseorang dalam menentukan genre suatu musik yang dibuat melalui metode Jaringan Syaraf Tiruan (JST) *Backpropagation*. Melalui sistem ini, pengguna dapat menentukan genre musik dengan mudah dan cepat karena didalam proses pengklasifikasian tidak memerlukan waktu yang lama.dari metode tersebut dihasilkan 0,084 untuk genre jazz, 114 untuk genre reggae, 221 untuk genre dangdut, 128 untuk genre rock dan 0,42 untuk genre pop.

(Setiawan, Cholissodin, & MP, 2016) melakukan penelitian dengan judul *Mendeteksi Jenis Burung Berdasarkan Pola Suaranya* yang bertujuan untuk membuat tool otomatis untuk dapat mendeteksi spesies burung apa saja yang ada disuatu wilayah.

(Bagaswari, 2019) melakukan penelitian dengan judul *Implementasi Algoritma Backpropagation Pada Pengenalan Emosi Berdasarkan Suara Manusia* yang bertujuan mengimplementasikan algoritma *backpropagation* pada pengenalan emosi berdasarkan suara manusia.

Penelitian-penelitian tersebut digambarkan keterhubungannya melalui pemetaan pustaka pada Gambar 2.1 berikut ini.

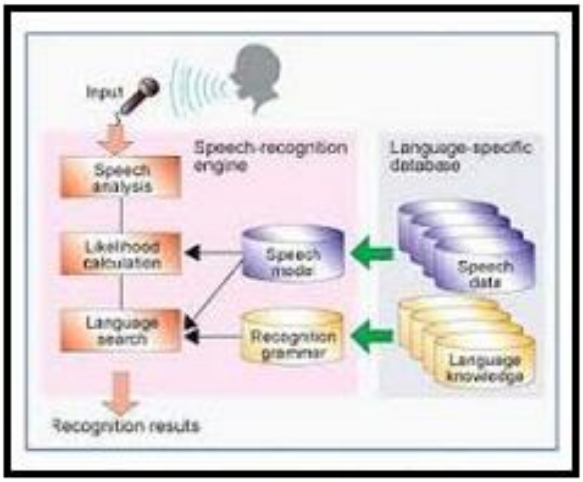


Gambar 2.1. Pemetaan Tinjauan Pustaka

2.2. Pengenalan Suara

Pengenalan suara merupakan salah satu upaya agar suara dapat dikenali atau diidentifikasi sehingga sinyal suara dapat dimanfaatkan. Pengenalan suara dapat dibedakan ke dalam beberapa bentuk pendekatan, yaitu pendekatan akustik-fonetik (*the acoustic-phonetic approach*), pendekatan kecerdasan buatan (*the artificial intelligence approach*), dan pendekatan pengenalan-pola (*the pattern recognition approach*).

Pendekatan pengenalan pola terdiri dari dua langkah yaitu pembelajaran pola suara dan pengenalan pola melalui perbandingan pola. Tahap perbandingan pola adalah tahap bagi ucapan akan dikenali, dibandingkan polanya dengan setiap kemungkinan pola yang telah dipelajari dalam *fase* pembelajaran, untuk kemudian diklasifikasi dengan pola terbaik yang sesuai. (Hapsari, 2011)

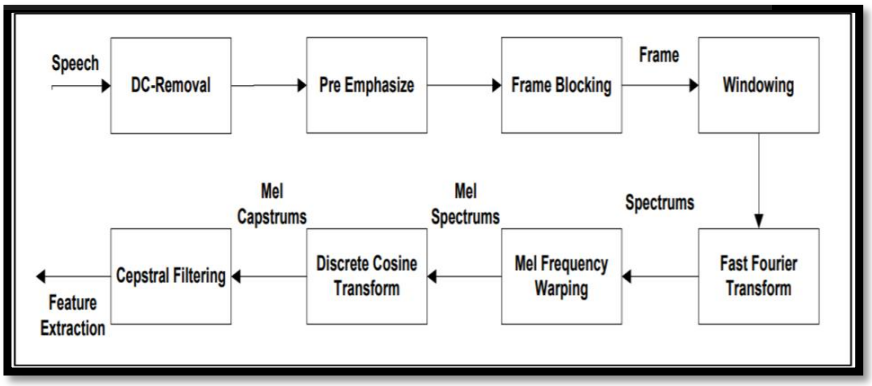


Gambar 2.2. Skema Pengenalan Suara

Sumber (Melissa, 2008)

2.3. *Mel Frequency Cepstrum Coefficient (MFCC)*

Mel-Frequency Cepstral Coefficient (MFCC) merupakan metode dari eksrasi ciri yang menghitung koefisien *cepstral* dengan mempertimbangkan pendengaran manusia. MFCC *feature extraction* merupakan penyesuaian dari sistem pendengaran manusia, dimana sinyal suara akan disaring dengan cara linear untuk frekuensi rendah (dibawah 1000 *Hz*) (Triansyah & N, 2017) dan secara logaritmik untuk frekuensi tinggi (diatas 1000 *Hz*).



Gambar 2.3. Tahapan Metode MFCC

(Sumber : Jurnal Ilmiah Teknologi Informasi Terapan)

Gambar 2.3 merupakan gambaran dari tahapan proses ekstraksi ciri menggunakan metode MFCC, tahapan untuk melakukan esktraksi ciri yaitu dimulai dari input suara, selanjutnya dilakukan proses *Pre-emphasize* , *framing*, *windowing*, FFT, *mel filter bank*, *discrete cosine transform* dan *cepstral filtering* lalu menghasilkan ekstraksi dari suara. Berikut ini merupakan penjelasan tahapan dari proses MFCC yang telah dilakukan berdasarkan dari jurnal penelitian (Nurhasanah, Dewi, & Saputro, 2018)

2.3.1. *DC – Removal*

Remove DC Components digunakan untuk menghilangkan sebagian komponen sinyal DC(arus searah) untuk mendapatkan hasil normalisasi dari data suara yang masuk. Untuk menghitung nilai rata-rata dari data sampel suara, dan mengurangkan nilai dari setiap sampel suara dengan nilai rata-rata tersebut (Putra & Resmawan, 2011). Persamaan 2.1 adalah untuk menghitung *DC-Removal*.

$$y[n] = x[n] - \bar{x}\,, 0 \le n \le N - 1 \qquad (2.1)$$

Keterangan : y[n] = Sample sinyal hasil proses *DC-Removal*,

x[n] = Sample sinyal uji,

$\bar{x}$ = Nilai rata-rata sample sinyal uji,

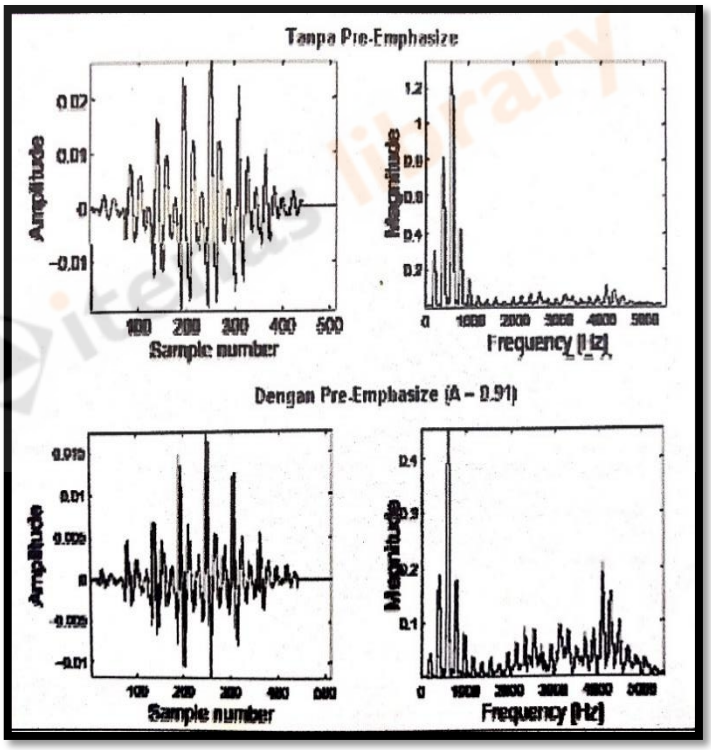
N = Panjang sinyal ($\frac{Fs}{Ts}$).

2.3.2. *Pre-Emphasize*

Pre-emphasize filtering adalah salah satu proses *filter* yang sering digunakan sebelum sinyal diproses lebih lanjut. *Filter* ini mempertahankan beberapa bagian frekuensi tinggi yang terdapat pada sebuah spektrum, yang umumnya tereliminasi pada saat proses produksi suara. Tujuan dari *Pre-emphasize filtering* ini adalah :

- Mengurangi *noise ratio* yang ada pada sinyal, sehingga dapat meningkatkan kualitas dari sinyal.
- Menyeimbangkan nilai spektrum *voiced sound*. Pada saat memproduksi *voice sound*, *glottis* manusia menghasilkan sekitar -12 dB *octave slope.*

Namun saat energi akustik tersebut dikeluarkan melalui bibir, terjadi peningkatan sebesar +6 dB. Sehingga sinyal yang terekam melalui *microphone* adalah sekitar -6 dB *octave slope* (Putra & Resmawan, 2011). Dampak dari efek ini dapat dilihat pada Gambar 2.4.



Gambar 2.4. Contoh dari Pre-Emphasize Pada Frame

(Sumber : Manunggal, 2005)

Pada Gambar 2.4 terlihat bahwa distribusi energi pada setiap frekuensi terlihat lebih seiras setelah diimplementasikan kedalam proses *pre-emphasize filter*.

Bentuk yang paling umum digunakan dalam *pre-emphasize filter* adalah Persamaan 2.2 sebagai berikut :

$$y[n] = s[n] - \alpha\, s[n-1] \quad (2.2)$$

Keterangan : y[n] = Sinyal hasil *pre-emphasize filter,*

s[n] = Sinyal sebelum *pre-emphasize filter*.

Setelah mengetahui hasil *pre-emphasize* selanjutnya data sinyal sebelum proses *pre-emphasize* ditambahkan dengan data sinyal setelah proses *pre-emphasize.*

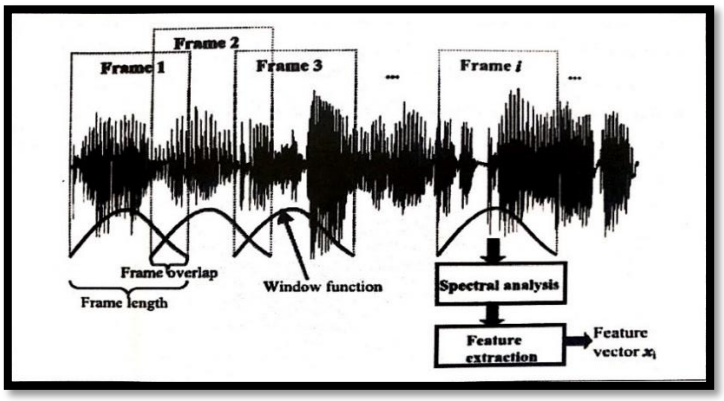
Persamaan 2.3 proses setelah *pre-emphasize*

$$Xn = Nn + y[n] \quad (2.3)$$

Pada umumnya nilai α yang sering digunakan adalah antara 0.9 sampai 1.0 , sedangkan *default* nilai α adalah 0.97 (Fawwaz Muhammad S, 2020).

2.3.3. *Frame Blocking*

Karena sinyal suara selalu berubah akibat adanya pergeseran artikulasi dari organ produksi vokal, sinyal harus diproses secara *short segments* (*short frame*). Panjang *frame* biasanya digunakan untuk pemrosesan sinyal adalah antara 10-30 milidetik. Panjang *frame* yang digunakan sangat mempengaruhi keberhasilan dalam analisa spektral. Di satu sisi, ukuran *frame* harus sepanjang mungkin untuk dapat menunjukkan resolusi *frekuensi* yang baik. Tetapi di sisi lain, ukuran *frame* juga harus cukup pendek untuk dapat menunjukkan resolusi waktu yang baik.



Gambar 2.5.Short Term Spectral Analisys
(Sumber : Resmawan, 2011)

Proses *frame blocking* Gambar 2.5 ini dilakukan sampai seluruh sinyal dapat diproses. Selain itu, proses ini dilakukan secara *overlapping* untuk setiap *frame*- nya. Panjang daerah *overlap* yang umum digunakan adalah kurang lebih 30% sampai 50% dari panjang *frame*. *Overlapping* dilakukan untuk menghindari hilangnya ciri atau karakteristik suara pada perbatasan perpotongan setiap *frame* (Putra & Resmawan, 2011).

Untuk menghitung jumlah *frame* digunakan Persamaan 2.4 sebagai berikut :

$$Jumlah\ Frame = ((I - N)/(M + 1) \quad (2.4)$$

Keterangan : I = *Sample rate,*

N = *Sample point* (*sample rate* * waktu *framing*),

M = N/2 .

2.3.4. *Windowing*

Proses *framing* bisa saja menyebabkan terjadinya kebocoran spektral (*spectral leakage*) atau aliasing. Aliasing merupakan sinyal baru dimana *frekuensi* memiliki perbedaan dengan sinyal aslinya. Efek tersebut bisa terjadi karena rendahnya jumlah *sampling rate*, atau bisa juga karena proses *frame*

blocking dimana proses ini menyebabkan sinyal menjadi *discontinue*. Untuk mengurangi beberapa kemungkinan terjadinya kebocoran spektral, maka hasil dari proses *framing* harus melewati proses *window*.

Sebuah fungsi *window* yang benar harus menyempit pada bagian *main lobe* dan melebar pada bagian *side lobe*-nya. Representasi dari fungsi *window* terhadap sinyal suara sebagai masukan dihitung menggunkan Persamaan 2.5 sebagai berikut:

$$x(n) = x_i(n)w(n)n = 0,1,\dots,n-1 \quad (2.5)$$

Keterangan : $x(n)$ = Nilai sampel sinyal hasil *windowing*,

$x_i(n)$ = Nilai sampel sinyal dari *frame* sinyal ke-i,

$w(n)$ = fungsi *window*,

n = *frame size*, merupakan kelipatan 2,

Ada beberapa fungsi *window*,namun yang sangat sering digunakan dalam aplikasi *speaker recognition* adalah *hamming window*. Fungsi *window* ini menghasilkan *noise* yang tidak terlalu besar, selain itu *sidelobe level* yang dihasilkan juga tidak terlalu tinggi (kurang lebih -43 dB). (Putra & Resmawan, 2011)

Fungsi *Hamming window* dihitung menggunakan Persamaan 2.6 sebagai berikut :

$$W = 0.54 - 0.46\cos\frac{2\pi n}{M-1} \quad (2.6)$$

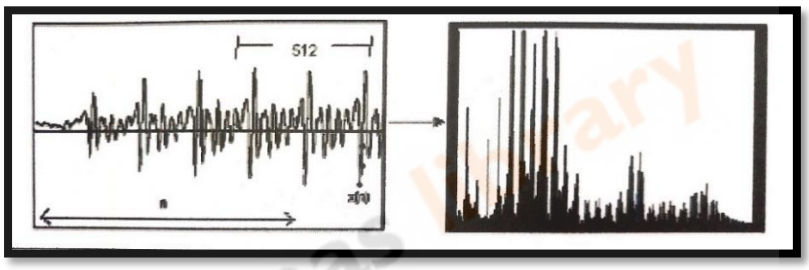
Keterangan : W = *Hamming window*,

n =0,1,2,...,M-1,

M = Panjang *frame*.

2.3.5. *Fast Fourier Transform (FFT)*

Fast Fourier Transform (FFT) merupakan langkah dari perhitungan DFT yang mudah sehingga dapat mempercepat proses perhitungan *Discrete Fourier Transform* (DFT). Perhitungan DFT secara langsung dalam komputerisasi akan memakan waktu yang sangat lama. Hal ini disebabkan karena dengan adanya DFT, dibutuhkan N^2 perkalian bilangan kompleks. Oleh sebab itu dibutuhkan cara lain untuk menghitung DFT dengan cepat. FFT juga dapat menghilangkan proses perhitungan ganda yang ada dalam DFT. (Putra & Resmawan, 2011)



Gambar 2.6.Domain Waktu Menjadi Frekuensi
(Sumber : Resmawan, 2011)

Untuk melakukan perhitungan FFT, dihitung menggunkan Persamaan 2.7 sebagai berikut:

$$F(K) = \sum_{n=1}^{N} f(n) \cos\left(\frac{2\pi nkT}{N}\right) - j \sum_{n=1}^{N} f(n) \sin\left(\frac{2\pi nkT}{N}\right) \qquad (2.7)$$

Keterangan : F(k) = *Forurier Form Transform,*

F(n) = Sampel data,

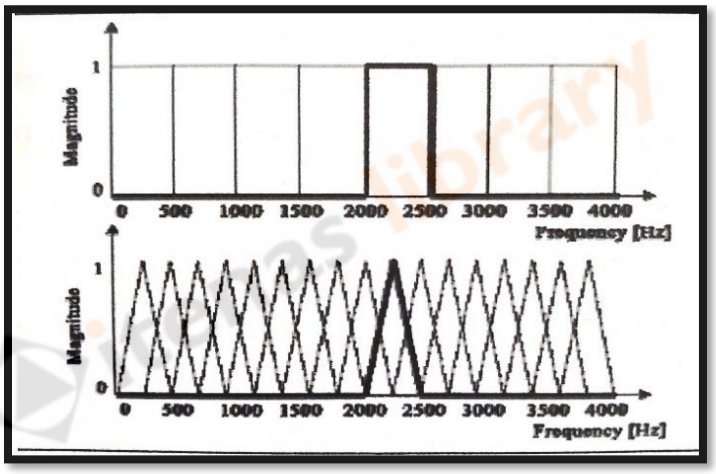
K = Sampel ke-n,

N = Titik *transform*

T = hasil *windowing*

2.3.6. *Mel Frequency Warping*

Mel Frequency Wrapping biasanya dilakukan dengan menggunakan *Filterbank*. *Filterbank* merupakan salah satu bentuk dari *filter* yang bertujuan untuk mengetahui ukuran energi dari *frequency band* tertentu yang ada dalam sinyal suara. *Filterbank* digunakan baik pada domain waktu maupun pada domain frekuensi, tetapi didalam proses MFCC, *filterbank* harus diterapkan dalam domain frekuensi. Gambar 2.7 menunjukan dua jenis *filterbank magnitude* dengan *filter* secara linier terhadap frekuensi 0-4 kHz. (Putra & Resmawan, 2011)



Gambar 2.7.Contoh Magnitude dari Rectangular dan Triangular Filterbank

(Sumber : Setiawan,dkk. 2011)

Filterbank menggunakan representasi *konvolusi* dalam melakukan *filter* terhadap sinyal. *Konvolusi* dilakukan dengan melakukan proses multiplikasi antara spektrum sinyal dengan koefisien *filterbank*. Untuk dapat melakukan tahap *filterbank,* maka sebaiknya harus mencari nilai-nilai koefisien dari *filterbank.* Nilai koefisien dari *filterbank* dapat dicari meggunakan Persamaan 2.8 sebagai berikut :

$$H_i = 2595 * \log(1 + 1000/700)/S_i/2 \quad (2.8)$$

Keterangan : H_i = Koefisien *filterbank*,

S_i= Nilai hasil dari proses FFT,

2595= Variabel hasil *mel.*

Setelah mendapatkan nilai koefisien *filterbank*, selanjutnya *filterbank* dapat dihitung menggunakan Persamaan 2.9 sebagai berikut :

$$Y(t) = \sum_{f=1}^{n} S(j)\, H(j) \qquad (2.\ 9)$$

Keterangan : Y(t) = *Filterbank*,

n = jumlah *magnitude spectrum* (n ϵ n)

S(j) = *Magnitude spectrum* pada frekuensi j

H(j) = koefisien *filterbank* pada frekuensi j

M = jumlah *channel* dalam *filterbank*

Pandangan manusia terhadap frekuensi dari sinyal suara tidak mengikuti *linear scale*. Frekuensi yang sebenarnya (dalam bentuk Hz) merupakan sebuah sinyal akan diukur manusia secara subyektif dengan menggunakan *mel scale*. *Mel frequency scale* merupakan *linear* frekuensi *scale* pada frekuensi dibawah 1000 Hz, dan merupakan *logarithmic scale* pada frekuensi diatas 1000 Hz.

2.3.7. *Discrete Cosine Transform (DCT)*

DCT merupakan langkah terakhir dari proses MFCC *feature extractio*n. Konsep dasar dari DCT adalah mendekorelasikan *mel spectrum* sehingga dapat menghasilkan representasi nilai terbaik dari properti spektral lokal. Konsep dasar dari DCT sama dengan *inverse fourier transform*. Tetapi hasil dari DCT biasanya mendekati PCA (*principle component analysis*). PCA merupakan metode *static* klasik yang dapat digunakan secara luas dalam analisa data dan kompresi. Hal ini mengakibatkan DCT menggantikan *inverse fourier transform* dalam proses MFCC *feature extraction*. (Putra & Resmawan, 2011)

Berikut adalah formula yang digunakan untuk menghitung DCT seperti yang ditunjukan pada Persamaan 2.10.

$$Cn = \sum_{k=1}^{k}(logS_k)\cos\left[n\left(K-\frac{1}{2}\right)\frac{\pi}{k}\right]; n = 1,2,\dots K \qquad (2.10)$$

Keterangan : S_k = keluaran dari proses *filterbank* pada *index* k

K = jumlah koefisien yang diharapkan

Koefisien ke nol dari DCT pada umumya akan dihilangkan, walaupun sebenarnya mengindikasikan energi dari *frame signal* tersebut. Hal ini diimplementasikan karena, berdasarkan penelitian-penelitian yang pernah dilakukan, koefisien ke nol ini tidak *reliable* terhadap *speech recognition*.

2.3.8. *Cepstral Filtering*

Hasil dari proses MFCC *feature extraction* biasanya memiliki beberapa kelemahan. *Low order* dari *cepstral coefficients* sangat sensitif terhadap *spectral slope*, sedangkan pada bagian *high order*nya sangat sensitif terhadap *noise*. Oleh sebab itu, *cepstral liftering* menjadi salah satu standar teknik yang diterapkan untuk meminimalisasi sensitifitas tersebut. (Putra & Resmawan, 2011)

Cepstral liftering dapat dilakukan dengan menggunakan fungsi *window* terhadap *cepstral features* seperti yang ditunjukan pada Persamaan 2.11 sebagai berikut:

$$W(n) = C_n + \frac{L}{2}\sin\left(\frac{n\pi}{L}\right) n = 1,2,\dots,L \qquad (2.11)$$

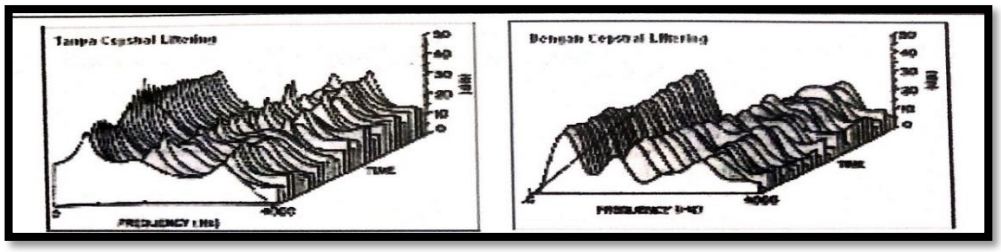
Keterangan : L = jumlah *cepstral coefficients*

n = *index* dari *cepstral coefficients*

Cn = Nilai hasil DCT

Cepstral liftering berfungsi menghaluskan spektrum hasil dari *main processor* sehingga dapat digunakan dengan lebih baik untuk *pattern matching*.

Gambar 2.8 merupakan perbandingan dari spektrum dengan dan tanpa *cepstral liftering*.



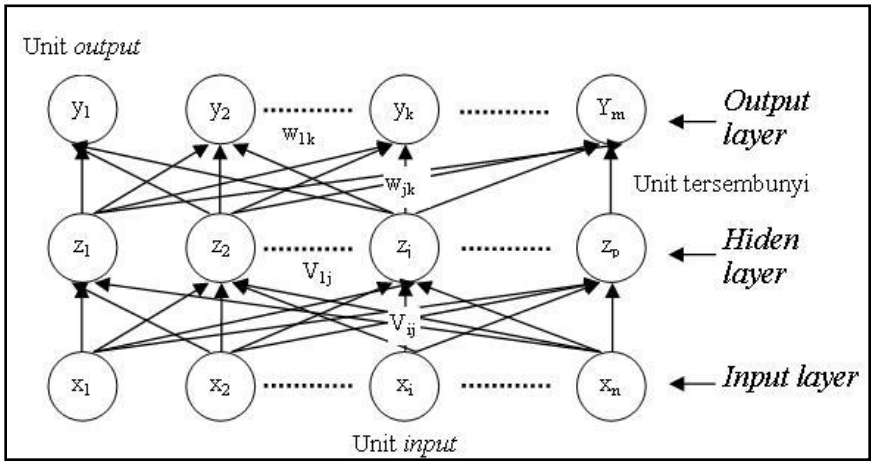
Gambar 2.8.Perbandingan Spektrum Dengan dan Tanpa Cepstral Liftering

(Sumber : Resmawan, 2011)

Berdasarkan Gambar 2.8 dapat dilihat perbandingan spektrum setelah melalui tahap *cepstral liftering* (kanan) lebih halus dari pada spektrum yang tidak melalui tahap *cepstral liftering* (kiri).

2.4. Jaringan Syaraf Tiruan (*Backpropagation*)

Algoritma yang paling populer pada algoritma *neural network* adalah algoritma *backpropagation*. Algoritma pelatihan *backpropagation* atau yang diterjemahkan menjadi propagasi balik pertama kali dirumuskan oleh Paul Werbos pada tahun 1974 dan dipopulerkan oleh Rumelhart bersama McClelland untuk dipakai pada *neural network*. Metode *backpropagation* pada awalnya dirancang untuk *neural network feedforward*, tetapi pada perkembangannya, metode ini diadaptasi untuk pembelajaran pada model *neural network* lainnya (Astuti, 2009). Adapun arsitektur jaringan syaraf *backpropagation* seperti terlihat pada gambar dibawah ini:



Gambar 2.9.Arsitektur Backpropagation

Dengan keterangan :

x_1 s/d x_n : *input layer*

z_1 s/d z_p : *hidden layer*

y_1 s/d y_m : *output layer*

Jaringan saraf tiruan *backpropagation* memiliki 2 tahap utama yang harus dilakukan yaitu tahap perambatan maju, dan tahap perambatan mundur.Perambatan maju merupakan proses yang berfungsi untuk mengetahui output dari JST, sedangkan perambatan mundur merupakan proses yang berfungsi untuk memperbaharui nilai bobot W dan bobot V (Wadi, 2020)

JST *backpropagation* merupakan algoritma yang bekerja dengan cara mengenali pola data berdasarkan riwayat data terdahulu. Algoritma JST *backpropagation* melakukan proses pembelajaran terhadap riwayat data terdahulu sehingga mampu mengenali pola data untuk melakukan prediksi atau klasifikasi terhadap data-data yang baru.

Untuk dapat melakukan proses klasifikasi atau prediksi data dengan menggunakan JST *backpropagation* maka harus dilakukan 2 proses yaitu proses pelatihan dan proses pengujian. Dengan penjelasan sebagai berikut:

a. Proses pelatihan

Proses ini merupakan proses *training* bagi JST *backpropagation* yang bertujuan untuk mengenali pola data. Proses pelatihan ini melatih bobot-bobot sehingga dapat mengenali pola data. Hasil dari proses pelatihan adalah nilai bobot V, bobot W , dan bias yang terlatih. Bobot-bobot tertatih tersebut merupakan representasi dari pengetahuan yang dimiliki oleh JST.

Pada proses pelatihan JST akan melakukan dua tahap yaitu tahap perambatan maju dan tahap perambatan mundur. Proses pelatihan diartikan juga sebagai perulangan terhadap proses perambatan maju dan perambatan mundur yang memliki tujuan utama adalah untuk mendapatkan bobot V, bobot W, dan bias yang terlatih.

b. Proses pengujian

Proses ini merupaka bagian dari proses klasifikasi atau proses prediksi yang dilakukan terhadap data uji atau data yang baru.

Algoritma JST *backpropagation* memiliki beberapa parameter yang harus ditentukan saat proses prediksi atau klasifikasi data. Berikut adalah parameter-parameter yang harus ditentukan:

a. Jumlah *neuron input layer* (n_{input})

Parameter ini digunakan untuk menentukan jumlah *neuron* yang terdapat pada *input layer*.

b. Jumlah *neuron hidden layer*(n_{hidden})

Parameter ini digunakan untuk menentukan jumlah *neuron* yang terdapat pada *hidden layer*.

c. Jumlah *neuron output layer*(n_{output})

Parameter ini digunakan untuk menentukan jumlah *neuron* yang terdapat pada *output layer*.

d. Laju pembelajaran (α)

Parameter ini akan menentukan kecepatan laju pembelajaran JST *backpropagation*. Dengan kata lain, parameter ini akan menentukan seberapa cepat proses pelatihan dari JST *backpropagation.*

e. Jumlah iterasi

Parameter ini akan menentukan jumlah perulangan dari proses pelatihan. Dengan kata lain, parameter ini akan menentukan berapa banyak proses perambatan maju dan proses perambatan mundur akan dilakukan pada proses pelatihan.

f. Toleransi *error*

Parameter ini digunakan sebagai acuan *error* antara output JST dan target output JST. Parameter ini digunakan untuk *trigger* yang akan menghentikan proses pelatihan. Jadi, pada saat nilai *error* antara *output* JST dan target *output* JST lebih kecil atau sama dengan toleransi *error* maka proses pelatihan biasanya dihentikan

Setelah penentuan parameter-parameter sebelumnya, maka penentuan nilai bobot V awal dan nilai bobot W awal sudah bisa dilakukan. Maka selanjutnya proses perambatan maju dan proses perambatan mundur pada JST *backpropagation* bisa dilakukan. Berikut adalah langkah-langkah yang dilakukan pada proses perambatan maju dan proses perambatan mundur pada JST *backpropagation*.

1. Perambatan maju

Pada proses perambatan maju dilakukan perhitungan nilai sinyal yang dikirim dari *input layer* ke *hidden layer* (perhitungan nilai *neuron* pada *hidden layer*). Selain itu dilakukan perhitungan nilai sinyal keluaran yang diteruskan dari *hidden layer* (perhitungan nilai *neuron* pada *output layer*). Berikut adalah tahapan yang dilakukan pada proses perambatan maju.

a. Perhitungan nilai *neuron* pada *hidden layer* (nilai Z)

Perhitungan nilai $Z_i(Z_1, Z_2, Z_3, Z_4, \ldots., Z_{n)}$ dilakukan dengan menggunakan persamaan berikut ini

$$Z_{inj} = v_{oj} + \sum_{i=0}^{n} x_i v_{ij} \quad (2.12)$$

$$z_j = f(z_{inj}) \quad (2.13)$$

Dimana:

v_{oj} =Bias pada *neuron hidden layer* ke-j

x_i = *Neuron input* ke-i

v_{ij}= Bobot yang menghubungkan *neuron input* ke-i dan neuron *hidden layer* ke-j

Z_{inj}=Sinyal dari *input layer* ke *neuron hidden layer* ke-j

Z_j=*Neuron hidden* layer ke-j

$f(z_{inj})$=Fungsi aktivasi terhadap nilai z_{inj}

Perhitungan nilai *neuron* pada *output layer* (nilai Y)

Perhitungan nilai $Y_i(Y_1, Y_2, Y_3, Y_4, \ldots, Y_{n)}$ dilakukan dengan menggunakan persamaan berikut ini

$$y_{ink} = w_{0k} + \sum_{j=0}^{n} z_j \, w_{jk} \quad (2.14)$$

$$y_k = f(y_{ink}) \quad (2.15)$$

Dimana

$w_{0k} =$ Bias pada *neuron* ke-k

$z_j =$ *Neuron hidden layer* ke-j

$w_{jk} =$ Bobot yang menghubungkan *neuron hidden layer* ke J dan *neuron output layer* ke-k

$y_{ink} =$ Sinyal dari *hidden layer* ke *neuron output layer* ke-k

$y_k =$ *Neuron output layer* ke k

$f(y_{ink}) =$Fungsi aktivasi terhadap nilai y

2. Perambatan mundur

Proses perambatan mundur dilakukan untuk memperbarui nilai bobot yang menghubungkan *output layer* dan *hidden layer* (bobot W). Juga untuk memperbarui bobot yang menghubungkan *hidden layer* dan *input layer* (bobot V). Berikut berikut adalah tahapan yang dilakukan pada proses perambatan mundur:

Perhitungan perambatan mundur dari *output* layer ke *hidden layer*

Tahapan ini bertujuan untuk memperbarui bobot yang menghubungkan *output layer* dan *hidden layer* (bobot W). Berikut adalah persamaan yang digunakan dalam tahapan ini:

$$\delta_k = (t_k - y_k) f'(y_{ink}) \quad (2.16)$$

$$= (t_k - y_k) y_k (1 - y_k)$$

$$\Delta w_{jk} = \alpha \delta_k z_j \quad (2.17)$$

$$\Delta w_{ok} = \alpha \delta_k \quad (2.18)$$

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (2.19)$$

$$w_{ok}(baru) = w_{ok}(lama) + \Delta w_{ok} \quad (2.20)$$

Dimana:

δ_k = Faktor koreksi dari *neuron output* ke-k

t_k = Target output *neuron output* ke-k

y_k = *Neuron output* ke-k

α = Laju pembelajaran

Δw_{jk} = Koreksi bobot yang menghubungkan *neuron output* ke-k dan *neuron hidden* layer ke-j

Δw_{ok} =Koreksi bias pada *neuron output* ke-k

$w_{jk}(baru)$ =Bobot terbaru yang menghubungkan *neuron output* ke-k dan *neuron hidden layer* ke-j.

$w_{ok}(baru)$ =Bobot lama yang menghubungkan *neuron output* ke-k dan *neuron layer* ke-j

Perhitungan perambatan mundur dari *hidden layer* ke *input layer*

Tahapan ini bertujuan untuk memperbarui bobot yang menghubungkan *hidden input layer* (bobot V). Berikut adalah persamaan yang digunakan dalam tahapan ini

$$\delta_{inj} = \sum_{k=1}^{n} \delta_k w_{jk} \quad (2.21)$$

$$\delta_j = \delta_{inj} f'(z_{inj}) \quad (2.22)$$

$$\delta_j = \delta_{inj} z_j (1 - z_j) \quad (2.23)$$

$$\Delta V_{ij} = \alpha \delta_j x_i \quad (2.24)$$

$$\Delta V_{oj} = \alpha \delta_j \quad (2.25)$$

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad (2.26)$$

$$v_{oj}(baru) = v_{oj}(lama) + \Delta v_{oj} \quad (2.27)$$

Dimana:

δ_{inj} = Sinyal faktor koreksi dari *layer output* ke *neuron hidden layer* ke-j

δ_j = Faktor koreksi dari *neuron hidden layer*

α = Laju pembelajaran

ΔV_{ij} = Koreksi bobot yang menghubungkan *neuron hidden layer* ke-j dan *input layer* ke-i

ΔV_{oj} = Koreksi bias pada *neuron hidden layer*

$v_{ij}(baru)$=Bobot terbaru yang menghubungkan *neuron hidden layer* ke-j dan *input layer* ke-i

$v_{ij}(lama)$=Bobot lama yang menghubungkan *neuron hidden layer* ke-j dan *input layer* ke-i

2.5. Kakatua

Gambar 2.10.Burung Kakatua
(Sumber: id.wikipedia.org)

Kakaktua merupakan burung yang mudah dikenali dari bentuk anatomi kepala besar, leher pendek, dan paruh kokoh melengkung. Berbulu mengkilap yang khas, biasanya berwarna dominan hijau atau putih untuk berkamuflase diantara tumbuhan hutan. Kaki terdiri atas 2 jari kearah depan dan 2 jari kearah belakang yang digunakan untuk memanjat pohon. Paruh kerap digunakan sebagai kaki ketiga untuk memanjat atau berpegangan. Sayap umumnya ramping dan meruncing, membuat kakaktua dapat terbang cepat dan bermanuver (Burnie, 2008)

Menurut Kindersley (2010) burung kelompok paruh bengkok biasanya beraneka warna mencolok dan mempunyai banyak populasi di kawasan tropis, terutama di hutan tropis. Selain kakaktua sejati kelompok ini antara lain mencakup parkit dan betet. Berisik, bersifat sosial di alam liar dan juga burung kelompok ini disukai karena tampak indah dan cerdas. Merupakan kelompok peniru yang yang hebat, terbukti mereka dapat menirukan suara manusia, dan sangat sedikit kakaktua yang melakukan migrasi yang sesungguhnya.

Ciri fisik dari burung kakatua yang akan membedakan antara jantan dan betina adalah sebagai berikut:

- Jantan: Mata hitam pekat, paruh lebih besar, bulu ekor lebih lebar dan bulu jenggot lebih banyak
- Betina: Mata hitam bagian lingkaran berwarna kemerah merahan hati, paruh lebih kecil, bulu ekor tidak lebar dan terkadang tidak memiliki jenggot.