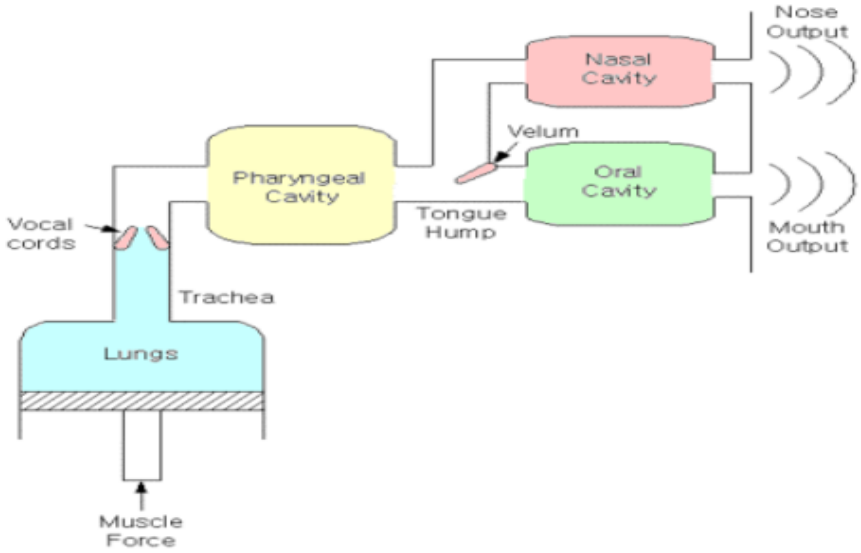
BAB II

LANDASAN TEORI

2.1. Suara Manusia

Menurut Inung Wijayanto (2013), produksi suara manusia memerlukan tiga elemen, yaitu sumber daya, sumber suara dan pemodifikasi suara. Ini adalah dasar dari teori *source-filter* pada produksi sinyal bicara. Sumber daya pada sinyal suara normal dihasilkan dari gerakan kompresi otot paru-paru. Sumber suara, selama sinyal *voiced* dan *unvoiced*, merupakan hasil dari getaran masing-masing pita suara. Pemodifikasi suara adalah artikulator, yang merubah bentuk *vocal tract* sehingga karakteristik frekuensi rongga akustik melewati apa yang dilalui suara.

Tiga kontrol utama pada produksi suara adalah paru-paru (sumber daya), posisi pita suara (sumber suara), dan bentuk *vocal tract* (pemodifikasi suara). *Vocal tract* terdiri dari *pharynx* (koneksi antara *esophagus* dengan mulut) dan mulut. *Nasal tract* mulai dari bagian belakang langit-langit dan berakhir pada *nostrils*.



Sumber: (Wijayanto, 2013)

Gambar 2.1. Diagram Blok Produksi Suara Manusia

Gambar 2.1. memperlihatkan model sistem produksi ucapan manusia yang disederhanakan. Pembentukan ucapan dimulai dengan adanya hembusan udara yang dihasilkan oleh paru-paru. Cara kerjanya serupa seperti piston atau pompa yang ditekan untuk menghasilkan tekanan udara. Pada saat *vocal cords* berada dalam keadaan tegang, aliran udara akan menyebabkan terjadinya vibrasi pada *vocal cords* dan menghasilkan bunyi ucapan yang disebut *voiced sound*. Pada saat *vocal cord* berada dalam keadaan lemas, aliran udara akan melalui daerah yang sempit pada *vocal tract* dan menyebabkan terjadinya turbulensi, sehingga menghasilkan suara yang dikenal dengan *unvoiced sound* (Inung Wijayanto dkk, 2013).

2.2. Tipe Suara Manusia

Tipe suara menurut Wijayanto (2013) adalah berbagai jenis suara yang diklasifikasikan menggunakan kriteria tertentu. Klasifikasi suara adalah proses dimana suara manusia dinilai, kemudian akan digolongkan menjadi tipe-tipe suara

tertentu. Ada banyak perbedaan tipe suara berdasarkan berbagai macam sistem klasifikasi. Tabel 2.1. menjelaskan tentang jangkauan vokal sesuai dengan tipe suara dan representasinya dalam frekuensi berdasarkan *scientific pitch notation*:

Tabel 2.1. Jangkauan Frekuensi Tiap Tipe Suara.

Gender	Tipe Suara	Range Vokal	Frekuensi Range Vokal (Hz)	Frequensi Fundamental (Hz)
Pria	Tenor	C3 –C5	130.813 - 523.251	16.35
	Bariton	F2 – F4	87.3071 - 349.228	21.80
	Bass	E2– E4	82.4069 - 329.628	20.60
Wanita	Soprano	C4–A5	261.626 - 1046.50	16.35
	Mezzo-Soprano	A3–A5	220.000 - 880.000	27.50
	Alto	F3 – F5	174.614 - 698.456	21.80

Sumber: (Wijayanto, 2013).

Tabel 2.1. menunjukkan jangkauan frekuensi tiap tipe suara manusia menurut Wijayanto (2013). Suara pria terdiri dari tipe Tenor, Bariton, dan Bass, sedangkan pada wanita terdiri dari Soprano, Mezzo-Soprano, dan Alto. Adapun *range* vokal antara suara pria dan wanita berbeda satu oktaf, sehingga wanita dapat menjangkau suara yang tinggi.

Frekuensi fundamental yang terdapat pada tabel 2.1. adalah frekuensi dasar manusia, sedangkan frekuensi range vokal adalah frekuensi saat manusia berbicara.

2.3. *Short Time Fourier Transform* (STFT)

Menurut Tulus Hayadi (2013), STFT (Short Time Fourier Transform) merupakan metode transformasi yang mengembangkan metode *Fourier Transform* dengan kelebihan pada kemampuan untuk mentransformasi *non-stationary signal*. Adapun ide dibalik metode ini adalah membuat *non-stationary*

signal menjadi suatu representasi *stationary* sinyal dengan memasukkan suatu *window function*. Dalam hal ini, sinyal yang ada dibagi menjadi beberapa segmen dimana segmen yang didapatkan, diasumsikan terdiri dari *stationary* signal.

Adapun rumus yang digunakan dapat dilihat pada persamaan:

(2.1)

$$STFT\{x[n]\}(m,\omega) \equiv X(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

Keterangan:

x[n] = sinyal masukan selama *n* waktu

n = waktu (sekon).

$w[n]$ = fungsi *windows*

ω = kecepatan sudut ($2\pi f$)

m = panjang *windows*

Perlu diperhatikan di sini bahwa x[n] adalah sinyal dengan domain waktu dan STFT{x[n]} adalah sinyal dengan domain frekuensi dan waktu. Karena itu, berbeda dengan *Fourier Transform*, STFT merupakan metode transformasi menghasilkan *Time-Frequency Representation* (TFR) dari sinyal. Di sini, w[n] adalah *window function* yang dapat mengambil bentuk distribusi normal dengan rumus berikut ini:

(2.2)

$$w[n] = e^{-a(n^2)/2}$$

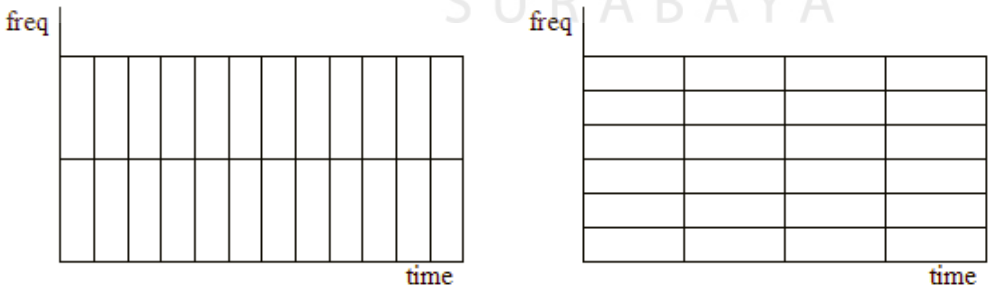
Dimana a adalah parameter untuk modulasi. Untuk menggambar spectrogram nya digunakan rumus

(2.3)

$$spectogram\{x(t)\}(\tau,\omega) = |X(\tau,\omega)|^2$$

Pada rumus 2.3., τ menunjukkan *range* dalam waktu ω adalah *range* dalam frekuensi. Sehingga spektrogram adalah sebuah grafik yang merepresentasikan sinyal ke dalam domain waktu dan frekuensi. Permasalahan yang muncul di sini adalah bahwa STFT menggunakan kernel window pada suatu interval waktu tertentu. Berbeda dengan *Fourier Transform* yang menggunakan kernel $e^{-2j\pi ft}$ sepanjang waktu, sehingga tidak ada permasalahan dalam hal resolusi frekuensi. Kalau STFT memilih *window* dengan lebar *infinity*, maka metode ini tidak akan ada bedanya dengan *Fourier Transform*. Dari ulasan yang singkat ini dapat diambil kesimpulan seperti pada gambar 2.2.:

- *Window* sempit: mempunyai resolusi waktu yang bagus, tetapi resolusi frekuensi yang tidak bagus
- *Window* lebar: mempunyai resolusi frekuensi yang bagus, tetapi resolusi waktu yang tidak bagus



Sumber: (Hayadi, 2013)

Gambar 2.2. *Window* sempit (kiri) dan *Window* lebar (kanan)

2.4. Jaringan Saraf Tiruan *Backpropagation*

2.4.1. Pengertian

Perambatan galat mundur (*Backpropagation*) menurut Kiki (2004) adalah sebuah metode sistematik untuk pelatihan *multilayer* jaringan saraf tiruan. Metode ini memiliki dasar matematis yang kuat, obyektif dan algoritma ini mendapatkan bentuk persamaan dan nilai koefisien dalam formula dengan meminimalkan jumlah kuadrat galat *error* melalui model yang dikembangkan (*training set*).

1. Dimulai dengan lapisan masukan, hitung keluaran dari setiap elemen pemroses melalui lapisan luar.
2. Hitung kesalahan pada lapisan luar yang merupakan selisih antara data aktual dan target.
3. Transformasikan kesalahan tersebut pada kesalahan yang sesuai di sisi masukan elemen pemroses.
4. Propagasi balik kesalahan-kesalahan ini pada keluaran setiap elemen pemroses ke kesalahan yang terdapat pada masukan. Ulangi proses ini sampai masukan tercapai.
5. Ubah seluruh bobot dengan menggunakan kesalahan pada sisi masukan elemen dan luaran elemen pemroses yang terhubung.

2.4.2. Arsitektur Model *Backpropagation*

Fungsi Aktivasi menurut Jong J.S:

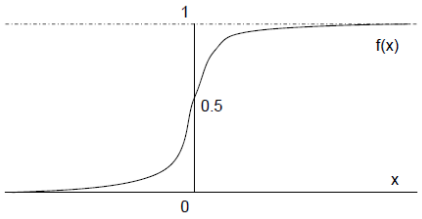
Syarat fungsi aktivasi yang dapat dipakai adalah kontinu, terdeferensial dengan mudah dan merupakan fungsi yang tidak turun

Fungsi yang sering dipakai adalah:

- *sigmoid* biner yang memiliki range (0,1)

  Grafik fungsinya:

  $f(x) = 1/(1 + e^{-x})$ dengan turunan $f'(x) = f(x)(1 – f(x))$ ....(2.4)



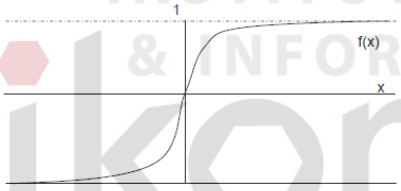
Sumber: (Jong, 2005)

Gambar 2.3. Grafik fungsi aktivasi *sigmoid* biner

- Fungsi *sigmoid* bipolar dengan *range* (1, -1)

  Grafik fungsinya:

  $f(x) = 2/(1 + e^{-x})-1$, $f'(x) = (1+f(x))(1-f(x))/2$ ...(2.5)



Sumber: (Jong, 2005)

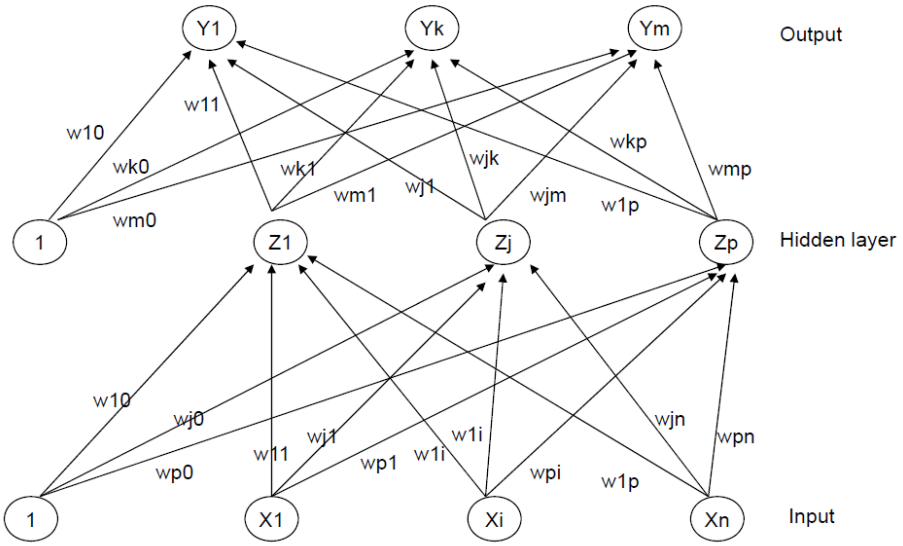
Gambar 2.4. Grafik fungsi aktivasi *sigmoid* bipolar

Adapun pelatihan standar *backpropagation* menurut Jong, J.S. yang terdapat pada gambar 2.5. adalah:

- Meliputi 3 fase, maju, mundur, dan modifikasi bobot
- Fase I Propagasi maju, sinyal masukan(x_i) dikalikan dengan bobot garis(w), kemudian dipropagasikan ke *hidden layer* menggunakan fungsi aktivasi(f(x)) yang ditentukan. Keluaran dari setiap unit *hidden*(z_j) selanjutnya dipropagasikan maju lagi ke layar *hidden* diatasnya

menggunakan fungsi aktivasi yang ditentukan, demikian seterusnya hingga menghasilkan keluaran jaringan (y_k). Berikutnya, keluaran jaringan dibandingkan dengan target yang harus dicapai (t_k). Selisih (t_k - y_k) adalah kesalahan yang terjadi. Jika kesalahan ini lebih kecil dari batas toleransi maka iterasi dihentikan, tetapi bila kesalahan masih lebih besar maka bobot setiap garis(w) dalam jaringan akan dimodifikasi untuk mengurangi kesalahan yang terjadi

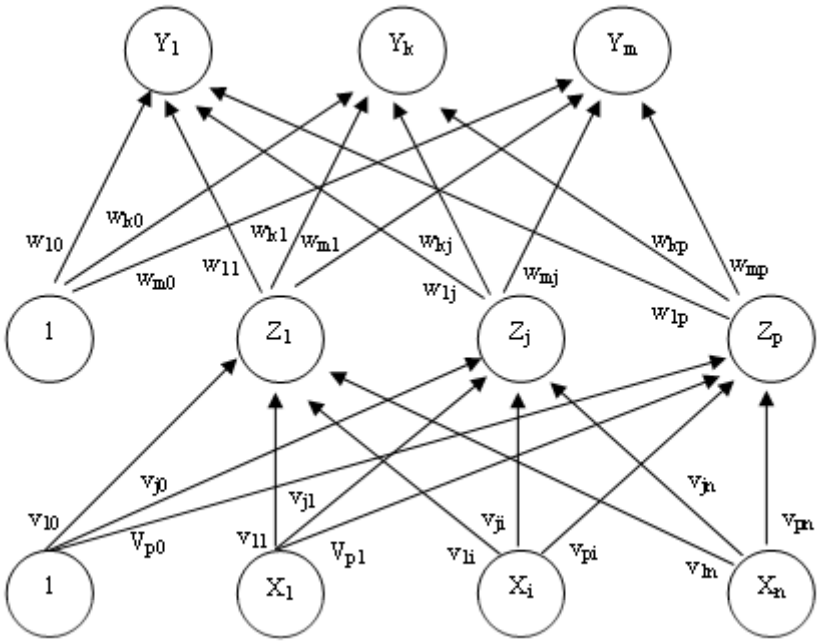
- Fase II Propagasi mundur, Berdasarkan kesalahan (t_k - y_k), dihitung faktor δ_k(k=1,2,3,..,m) yang dipakai untuk mendistribusikan kesalahan di unit (y_k) ke semua unit *hidden* yang terhubung langsung dengan y_k. δ_k juga dipakai untuk mengubah bobot garis(w) yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor δ_j di setiap unit di *hidden layer* sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di layar di bawahnya. Demikian seterusnya hingga semua faktor δ di unit *hidden* yang berhubungan langsung dengan unit masukan dihitung
- Fase III Perubahan bobot, bobot semua garis dimodifikasi bersamaan. Perubahan bobot suatu garis didasarkan atas faktor δ neuron di layar atasnya. Sebagai contoh, perubahan bobot garis yang menuju ke layar keluaran didasarkan atas δ_k yang ada di unit keluaran. Fase tersebut diulang hingga penghentian terpenuhi. Umumnya kondisi penghentian yang dipakai adalah jumlah iterasi atau kesalahan.



Sumber: (Jong, 2005)

Gambar 2.5. Algoritma *backpropagation* menurut Jong, J.S.

Backpropagation menurut Wirda Ayu Utari (2010), memiliki beberapa unit yang ada dalam satu atau lebih layar tersembunyi. Gambar 2.6. adalah arsitektur *backpropagation* dengan n buah masukan (ditambah sebuah bias), sebuah layar tersembunyi yang terdiri dari p unit (ditambah sebuah bias), serta m buah keluaran.



Sumber: (Utari, 2010)

Gambar 2.6. Arsitektur Model *Backpropagation* menurut W.A. Utari

V_{ji} merupakan bobot garis dari unit masukan X_i ke unit layar tersembunyi Z_j (V_{j0} merupakan bobot garis yang menghubungkan bias di unit masukan ke unit layar tersembunyi zj). W_{kj} merupakan bobot dari unit layar tersembunyi Z_j ke unit keluaran V_k (w_{k0} merupakan bobot dari bias di layar tersembunyi ke unit keluaran Z_k).

Algoritma *backpropagation* menggunakan *error* keluaran untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error* ini, tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi yang dapat dideferensiasikan, seperti sigmoid, tansig atau purelin.

Adapun algoritma *backpropagation* sesuai gambar 2.6. menurut Wirda Ayu Utari (2010), adalah sebagai berikut:

a. Inisialisasi bobot (ambil bobot awal dengan nilai *random* yang cukup kecil).

b. Tetapkan maksimum *epoch* (banyaknya iterasi), target *error*, dan *learning rate* (α).

c. Inisialisasi *Epoch* = 0; MSE = 1; dimana MSE adalah Mean Square Error (rata-rata kuadrat *error* yang didapatkan tiap iterasi)

d. Kerjakan langkah-langkah berikut selama *epoch* < maksimum *epoch* dan (MSE > target *error*):

1. Epoch = Epoch + 12
2. Untuk tiap-tiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan:

Feedforward :

a. Tiap-tiap unit masukan (x_i, i=1,2,3,...,n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada di atasnya (lapisan tersembunyi).

b. Tiap-tiap unit pada suatu lapisan tersembunyi (Z_j, j=1,2,3,...,p) menjumlahkan unit masukan (x_i) yang dikalikan dengan nilai bobot garis dari input menuju *hidden layer* (v_{ij}, i=1,2,3,...n; j=1,2,3,...,p) dengan masukan bias ($b1_j$, j=1,2,3,...n) :

(2.6)

$$z\_in_j = b1_j + \sum_{i=1}^{n} x_i v_{ij}$$

Gunakan fungsi aktivasi untuk menghitung sinyal keluarannya:

(2.7)

$$z_j = f(z\_in_j)$$

dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit keluaran).

Tiap-tiap unit keluaran (y_k, k=1,2,3,...,m) menjumlahkan unit masukan dari *hidden layer* (z_i) yang dikalikan dengan nilai bobot garis dari *hidden layer* menuju *output* (w_{jk}, j=1,2,3,...n; k=1,2,3,...,p) dengan masukan bias pada *hidden layer* ($b2_j$, j=1,2,3,...n):

(2.8)

$$y\_in_k = b2_k + \sum_{i=1}^{p} z_i w_{jk}$$

gunakan fungsi aktivasi untuk menghitung sinyal keluarannya:

(2.9)

$$y_k = f(y\_in_k)$$

dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit keluaran).

Catatan: Langkah (b) dilakukan sebanyak jumlah lapisan tersembunyi.

c. Tiap-tiap unit keluaran (y_k, k=1,2,3,...,m) menerima target pola yang berhubungan dengan pola masukan pembelajaran, hitung informasi errornya ($\delta 2_k$, k=1,2,3,...n):

(2.10)

$$\delta 2_k = (t_k - y_k) f'(y\_in_k)$$

(2.11)

$$\varphi 2_{jk} = \delta_k z_j$$

(2.12)

$$\beta 2_k = \delta_k$$

kemudian hitung koreksi bobot (Δw_{jk}) (yang nantinya akan digunakan untuk memperbaiki nilai w_{jk}) dengan mengalikan dengan nilai *learning rate* (α):

(2.13)

$$\Delta w_{jk} = \alpha \varphi 2_{jk}$$

hitung juga koreksi bias ($\Delta b2_k$) (yang nantinya akan digunakan untuk memperbaiki nilai $b2_k$):

(2.14)

$$\Delta b2_k = \alpha \beta 2_k$$

langkah (d) ini juga dilakukan sebanyak jumlah lapisan tersembunyi, yaitu menghitung informasi *error* dari suatu lapisan tersembunyi ke lapisan tersembunyi sebelumnya.

d. Tiap-tiap unit tersembunyi (z_j, j=1,2,3,...,p) menjumlahkan delta masukannya (dari unit-unit yang berada pada lapisan di atasnya):

(2.15)

$$\delta\_in_j = \sum_{k=1}^{m} \delta 2_k w_{jk}$$

kalikan nilai ini dengan turunan dari fungsi aktivasinya untuk menghitung informasi *error* ($\delta 1_j$, j=1,2,3,....n):

(2.16)

$$\delta 1_j = \delta\_in_j f'(z\_in_j)$$

(2.17)

$$\varphi 1_{ij} = \delta 1_j x_j$$

(2.18)

$$\beta 1_j = \delta 1_j$$

kemudian hitung koreksi bobot (Δv_{ij}) (yang nantinya akan digunakan untuk memperbaiki nilai (v_{ij}):

(2.19)

$$\Delta v_{ij} = \alpha \varphi 1_{ij}$$

hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai ($\Delta b1_j$):

(2.20)

$$\Delta b1_j = \alpha \beta 1_j$$

e. Tiap-tiap unit keluaran (Y_k, k=1,2,3,...,m) memperbaiki bias dan bobotnya (j=0,1,2,...,p):

(2.21)

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

(2.22)

$$b2_k(baru) = b2_k(lama) + \Delta b2_k$$

Tiap-tiap unit tersembunyi (Z_j, j=1,2,3,...,p) memperbaiki bias ($b1_j$) dan bobotnya (v_{ij}) (i=0,1,2,...,n) :

(2.23)

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

(2.24)

$$b1_j(baru) = b1_k(lama) + \Delta b1_j$$

3. Hitung MSE (Mean Square Error)

Menghitung nilai rata-rata kuadrat *error* (E = selisih target nilai dengan keluaran; n = banyak data).

(2.25)

$$MSE = \frac{\sum E^2}{n}$$