BAB II

TEORI PENUNJANG

Sinyal adalah besaran vektor yang berubah-ubah menurut waktu, ruang, atau peubah-peubah lainnya. Dalam pembahasan ini sinyal berubah menurut waktu. Contoh :

$$s(t) = 5\,t, \qquad t = \text{waktu} \tag{2.1}$$

sinyal $s(t)$ berubah-ubah secara linear menurut waktu peubah bebas t. Suatu segmen sinyal suara dapat dinyatakan dengan derajat ketelitian yang tinggi sebagai jumlah dari beberapa sinusoida dengan amplitudo dan frekuensi, contoh :

$$s(t) = \sum_{i=1}^{N} A_i(t) \sin[\,2\pi F_i(t)t + \theta_i(t)] \;,\; t = \text{waktu} \tag{2.2}$$

dimana $A_i(t)$ = himpunan amplitudo, $F_i(t)$ = himpunan frekuensi, dan $\theta i(t)$ = himpunan fase sinusoida. Sinyal suara adalah sinyal analog atau sinyal waktu kontinu yang didefinisikan untuk setiap nilai waktu dan diambil pada nilai-nilai dalam selang kontinu [a,b], dengan a → -~ dan b → ~ .

2.1 Pencuplikan Sinyal Analog

Beberapa sinyal komunikasi sudah menjadi pembawaannya bersifat digital atau sinyal waktu diskret, misalnya data teletip, keluaran-keluaran komputer, sinyal-sinyal radar, dan sonar yang dipulsa. Sinyal analog tidak bisa diproses oleh komputer karena merupakan sinyal waktu kontinu. Untuk dapat diolah oleh komputer perlu mengkonversi sinyal analog ke sinyal digital. Hal ini dilakukan mula-mula dengan mencuplik (sampel) sinyal analog pada suatu laju periodik (f_s cuplikan per detik) dan kemudian dikonversi lebih lanjut ke dalam cuplikan-cuplikan beramplitudo diskret dengan proses kuantisasi. Menurut **kriteria *Nyquist*** :

> " Agar pencuplikan suatu sinyal dibatasi pita B hertz tidak menghancurkan sebarang kandungan informasi, maka laju pencuplikan $f_s \geq 2B$."

Jadi laju pencuplikan *Nyquist* adalah laju pencuplikan minimum yaitu 2B per detik, dengan menggunakan selang pencuplikan 1/2B.

2.2 Metode *Linear Predictive Coding* (LPC)

2.2.1 Dasar LPC

Metode *Linear Predictive Coding* adalah metode dalam memisahkan efek-efek dari sumber (*source*) dan tapis (*filter*) dari sinyal suara. Metode ini mengkodekan sinyal suara dengan cara menemukan atau menentukan himpunan bobot dari nilai sinyal-sinyal sebelumnya yang dapat memprediksikan nilai sinyal selanjutnya. Hal ini dapat digambarkan dengan, misal sinyal untuk cuplikan kata pada waktu n yaitu $s(n)$ dapat didekati sebagai kombinasi linear dari sejumlah p cuplikan terdahulu, sehingga :

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + \ldots + a_p s(n-p), \quad p < n \qquad (2.3)$$

dimana koefisien a_1, a_2, ... , a_p diasumsikan konstan untuk satu frame analisa kata. Persamaan (2.3) dapat diubah dengan menambahkan faktor eksitasi G $u(n)$. Input $u(n)$ dapat berupa urutan unit impulse untuk *voice sound* atau *random noise* untuk *unvoiced sound.* Sumber eksitasi dipilih berdasarkan *voiced* dan *unvoiced* dari kata yang akan disintesa, dan kemudian diberi penguatan gain G yang menunjukkan keras lemahnya suara.

Sehingga menjadi :

$$s(n) = \sum_{i=1}^{p} a_i s(n-i) + Gu(n) \qquad (2.4)$$

Sinyal $s(n)$ dapat digeser menurut waktu dengan memindahkan variabel bebas n dengan $n-i$, dengan i adalah integer. Jika i positif, maka akan menghasilkan tunda (*delay*) pada sinyal dengan i satuan waktu. Jika i negatif, maka menghasilkan penambahan pada sinyal dengan i satuan waktu. Elemen tunda unit (z^{-1}) adalah suatu sistem khusus yang menunda secara sederhana sinyal yang melewatinya dengan satu cuplikan masukan $s(n)$ yang disimpan dalam memori pada waktu n untuk membentuk $s(n-1)$. Dengan mengubah persamaan (2.4) ke dalam domain z maka didapat :

$$S(z) = \sum_{i=1}^{p} a_i z^{-i} S(z) + GU(z) \qquad (2.5)$$

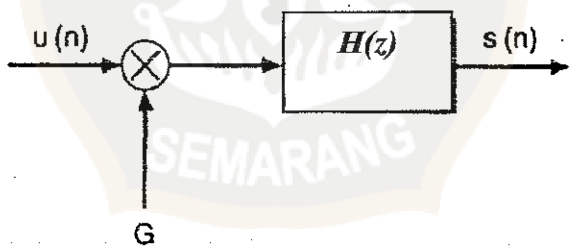
Spektrum suara $S(z)$ dihasilkan dengan melewatkan sumber eksitasi $U(z)$ melalui fungsi alih *all pole*, $H(z) = G/A(z)$. Sehingga di dapat :

$$S(z) = H(z)\ U(z) = \left(\frac{G}{A(z)}\right) U(z) \qquad (2.6\ a)$$

$$A(z) = \left(\frac{GU(z)}{S(z)}\right) \qquad (2.6\ b)$$

$$A(z) = 1 - \sum_{i=1}^{p} a_i z^{-i} \qquad (2.6\ c)$$

Interpretasi dari persamaan (2.6) dapat digambarkan seperti pada gambar 2.1, dimana tampak sumber eksitasi *u(n)* diskala dengan penguatan *G* bertindak sebagai masukan *filter all pole*, $H(z) = \dfrac{G}{A(z)}$ untuk menghasilkan sinyal kata *s(n)*.



Gambar 2.1. Model prediksi linier kata.

2.2.2 Analisa LPC

Berdasar pada gambar 2.1 di atas, hubungan antara *s(n)* dan *u(n)* adalah :

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) + Gu(n) \qquad (2.7)$$

Dengan menganggap kombinasi linier dari p cuplikan terdahulu sebagai perkiraan $\tilde{s}(n)$ sehingga :

$$\tilde{s}(n) = \sum_{k=1}^{p} a_k s(n-k) \tag{2.8}$$

Maka prediksi *error e(n)* yaitu :

$$e(n) = s(n) - \tilde{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k) \tag{2.9}$$

dengan fungsi alih *error* :

$$A(z) = \frac{E(z)}{S(z)} = 1 - \sum_{k=1}^{p} a_k z^{-k} \tag{2.10}$$

Terlihat bahwa *prediction error e(n)* sama dengan eksitasi terskala *Gu(n)* seperti terdapat pada gambar 2.1. Sehingga apabila *error* sinyal diminimalkan pada analisa sinyal berarti bahwa *error* sinyal akan seperti baik impuls sinyal maupun suara asli (*white noise*) dari sumber suara.

Permasalahan yang utama dalam pemodelan LPC ialah menentukan koefisien prediksi $\{a_k\}$ secara langsung dari sinyal kata sehingga sifat filter digital betul-betul sesuai dengan gelombang kata didalam satu jendela analisa. Karakteristik spektral sinyal kata bervariasi terhadap waktu, oleh karena itu untuk menentukan satu set koefisien prediksi $\{a_k\}$ pada suatu waktu *n*, haruslah diperkirakan dari suatu *segment* singkat sinyal kata yang terjadi di sekitar waktu *n*. Dengan kata lain permasalahannya adalah menemukan satu set koefisien prediksi $\{a_k\}$ yang dapat meminimkan kesalahan rata-rata kwadrat (*mean squared error*) selama satu *segment* singkat sinyal kata. Untuk itu didefinisikan *short term speech* dan kesalahan pada suatu waktu n yaitu :

$$s_n(m) = s(n+m) \tag{2.11a}$$

$$e_n(m) = e(n+m) \tag{2.11b}$$

dan akan dicari nilai kesalahan rata-rata kwadrat yang minimal dari sinyal kesalahan pada waktu n :

$$E_n = \sum_m e_n^2(m) \tag{2.12}$$

dimana dengan menggunakan definisi dari $e_n(m)$ dalam bentuk $s_n(m)$ (persamaan 2.9) dapat ditulis sebagai :

$$E(n) = \sum_m \left[s_n(m) - \sum_{k=1}^{p} a_k s_n(m-k) \right]^2 \tag{2.13}$$

agar didapatkan koefisien prediksi, persamaan (2.13) di atas harus dideferensialkan terhadap masing-masing a_k dan disama-dengankan nol :

$$\frac{\partial E_n}{\partial a_k} = 0, \quad k = 1,2,3, \ldots ,p \tag{2.14}$$

sehingga didapatkan :

$$\sum_m s_n(m-i)s_n(m) = \sum_{k=1}^{p} \hat{a}_k \sum_m s_n(m-i)s_n(m-k) \qquad , i = 1 \ldots \ldots p \tag{2.15}$$

dimana $\sum_m s_n(m-i)s_n(m-k)$ adalah *covariance* dari s_n(m) yaitu :

$$\phi_n(i,0) = \sum_{k=1}^{p} a_k \, \phi_n(i,k) \tag{2.16}$$

yang menunjukkan adanya p persamaan dengan p variabel tak diketahui, secara umum bahwa *minimum mean squared error* $\hat{E}_n$ dapat dinyatakan sebagai :

$$\hat{E}_n = \sum_m s_n^2(m) - \sum_{k=1}^{p} \hat{a}_k \sum_m s_n(m) s_n(m-k) \tag{2.17a}$$

$$= \phi_n(0,0) - \sum_{k=1}^{p} \hat{a}_k \, \phi_n(0,k) \tag{2.17b}$$

Untuk memecahkan persamaan (2.16) diperlukan perhitungan $\phi_n(i,k)$ untuk $1 \le i \le p$ dan $0 \le k \le p$ dan kemudian untuk menyelesaikan p persamaan digunakan metode autokorelasi.

2.2.3 Metode Autokorelasi

Cara yang cukup sederhana dan langsung adalah dengan mendefinisikan batas pada m di dalam penjumlahan yaitu dengan mengasumsikan bahwa *segment* kata, $s_n(m)$, sama dengan nol di luar interval $0 \le m \le N-1$. Dengan dibatasinya durasi $s_n(m)$ sebanyak N sample, dalam interval $0 \le m \le N-1$ yang berarti sama dengan mengalikan $s_n(m)$ dengan sebuah *window* dengan panjang N. Sehingga spektral dari sinyal asli yang terkonsentrasi hanya pada satu frekuensi, menjadi rata pada seluruh jarak frekuensi pada *window*. Maka cuplikan kata untuk diminimalkan dapat dinyatakan sebagai berikut :

$$s_n(m) = \begin{cases} s_n(m+n).w(m), & 0 \le m \le N-1 \\ 0 & m < 0 \text{ atau } m > N-1 \end{cases} \qquad (2.18)$$

Pada tugas akhir ini *window* yang digunakan adalah *Hamming window* yang memiliki bentuk persamaan sebagai berikut :

$$w(m) = 0{,}54 - 0{,}46\cos\left(\frac{2\pi m}{N-1}\right), \qquad 0 \le m \le N-1 \qquad (2.19)$$

Dari persamaan (2.18), untuk $m<0$, sinyal kesalahan $e(m)$ sama dengan nol sebab $s_n(m)$=0 untuk $m<0$, selanjutnya untuk $m>N-1$ sekali lagi sinyal kesalahan $e_n(m)$ sama dengan nol sebab $s_n(m)=0$ untuk $m>N-1$, sedangkan di daerah antaranya sinyal kata, $s_n(m)$ diprediksikan dari cuplikan-cuplikan sebelumnya dan harganya tidak sama dengan nol walaupun tidak menutup kemungkinan hasilnya sama dengan nol. Dengan demikian kesalahan terbesar terjadi di ujung awal dan ujung akhir dari prediksi sinyal kata, sebab pada ujung awal LPC berusaha memprediksikan sinyal kata yang tidak sama dengan nol dari cuplikan-cuplikan sebelumnya yang berharga sama dengan nol dan pada ujung akhir LPC berusaha memprediksikan sinyal kata yang berharga sama dengan nol dari cuplikan-cuplikan sebelumnya yang berharga tidak sama dengan nol.

Tujuan *windowing* dengan menggunakan fungsi *weighting* adalah untuk mentapis sinyal kata di dekat ujung-ujung awal dan ujung-ujung akhir sinyal kata, sehingga memperkecil terjadinya kesalahan pada kedua ujung-ujung tersebut.

Berdasarkan pada sinyal *weighted* pada persamaan (2.18) maka *mean squared error* menjadi :

$$E_n = \sum_{m=0}^{N-1+p} e_n^2(m) \tag{2.20}$$

dan $\phi_n(i,k)$ dapat dinyatakan sebagai :

$$\phi_n(i,k) = \sum_{m=0}^{N-1+p} s_n(m-i)s_n(m-k) \ , \quad \begin{matrix} 1 \le i \le p \\ 0 \le k \le p \end{matrix} \tag{2.21}$$

atau

$$\phi_n(i,k) = \sum_{m=0}^{N-1-(i-k)} s_n(m)s_n(m+i-k), \quad \begin{matrix} 1 \le i \le p \\ 0 \le k \le p \end{matrix} \tag{2.22}$$

karena persamaan (2.22) hanya fungsi dari *i-k* (dari dua variabel *i* dan *k*), maka fungsi *covariance* $\phi_n(i,k)$ tereduksi menjadi fungsi autokorelasi yang sederhana yaitu :

$$\phi_n(i,k) = r_n(i-k) = \sum_{m=0}^{N-1-(i-k)} s_n(m)s_n(m+i-k) \tag{2.23}$$

Karena fungsi autokorelasi bersifat simetris, yaitu $r_n(-k) = r_n(k)$, maka persamaan LPC dapat dinyatakan sebagai :

$$\sum_{k=1}^{p} r_n(|i-k|)\hat{a}_k = r_n(i), \qquad 1 \le i \le p \tag{2.24}$$

dan dapat dinyatakan dalam bentuk matriks sebagai berikut :

$$\begin{bmatrix} r_n(0) & r_n(1) & r_n(2) & . & . & . & r_n(p-1) \\ r_n(1) & r_n(0) & r_n(1) & . & . & . & r_n(p-2) \\ r_n(2) & r_n(1) & r_n(0) & . & . & . & r_n(p-3) \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ r_n(p-1) & r_n(p-2) & r_n(p-3) & . & . & . & r_n(0) \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \\ . \\ . \\ . \\ \hat{a}_p \end{bmatrix} = \begin{bmatrix} r_n(1) \\ r_n(2) \\ r_n(3) \\ . \\ . \\ . \\ r_n(p) \end{bmatrix} \quad (2.25)$$

Matriks autokorelasi $p \; x \; p$ ini merupakan matriks *Toeplitz* (simetris dengan seluruh elemen diagonalnya identik) dan oleh karenanya dapat diselesaikan secara efisien dengan menggunakan berbagai prosedur, antara lain yang terkenal adalah algoritma Levinson Durbin.

2.3 Transformasi Fourier

Transformasi Fourier merupakan representasi frekuensi domain dari suatu sinyal. Representasi ini mengandung informasi yang tepat sama dengan kandungan informasi dari sinyal semula. Domain frekuensi dapat diproses secara digital lebih cepat daripada domain waktu.

2.3.1 Discrete Fourier Transform (DFT)

DFT merupakan perluasan dari transformasi Fourier yang berlaku untuk sinyal-sinyal diskrit dengan panjang yang terhingga. Untuk sinyal waktu diskret *x(n)* dapat disajikan dengan :

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi nk/N}, k = 0,1,...,N-1 \quad (2.26)$$

Faktor eksponensial dalam persamaan tersebut dinamakan *twiddle factor* yang bersifat periodik dengan periode N dan dilambangkan dengan W_N^{nk}, sehingga DFT dari sinyal waktu diskrit $x(n)$ dapat dituliskan sebagai :

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n) W_N^{nk}, k = 0,1,\ldots, N-1 \qquad (2.27)$$

N menyatakan jumlah titik data, misalnya $N = 3$, maka persamaan tersebut dapat dikalkulasi sebagai berikut :

$$X(0) = (1/3)\,[x(0)W0 + x(1)W0 + x(2)W0] \qquad (2.28a)$$

$$X(1) = (1/3)\,[x(0)W0 + x(1)W1 + x(2)W2] \qquad (2.28b)$$

$$X(2) = (1/3)\,[x(0)W0 + x(1)W2 + x(2)W4] \qquad (2.28c)$$

Terlihat bahwa persamaan di atas memerlukan N^2 perkalian kompleks dan $N(N-1)$ penjumlahan kompleks. Perhitungan DFT secara langsung secara umum tidak efisien, karena tidak memanfaatkan sifat Simetri dan Periodisitas dari *twiddle factor*, yaitu

Sifat Simetri : $W_N^{k+N/2} = - W_N^k$, (2.29a)

Sifat Periodisitas : $W_N^{k+N} = W_N^k$, (2.29b)

Kedua sifat DFT di atas bisa digunakan untuk algoritma perhitungan yang lebih cepat dan efisien yaitu *Fast Fourier Transform* (FFT).

2.3.2 *Fast Fourier Transform* (FFT) *radix* 2 *Decimation in Frequency* (DIF)

Dengan FFT, maka jumlah titik data N merupakan bilangan yang dapat difaktorkan sebagai

$$N = r_1\ r_2\ r_3 \ldots\ r_j, \tag{2.30}$$

Dimana $r_1 = r_2 = r_3 = \ldots = r_j \equiv r$, sehingga $N = r^j$. Bilangan r disebut *radix* algoritma FFT. Pada tugas akhir ini digunakan FFT *radix* 2, yang banyak digunakan untuk analisa sinyal. Kemudian dengan memanfaatkan sifat simetri dan periodisitas dari *twiddle factor*, jumlah operasi aritmatika yang diperlukan dapat dikurangi.

Pada algoritma FFT *radix* 2 DIF, data sebanyak N dibagi menjadi dua bagian, yaitu data dengan indeks 0 sampai N/2 – 1 dan data dengan indeks N/2 sampai N-1.

$$X(k) = \sum_{n=0}^{(N/2)-1} x(n)\, W_N^{kn} + \sum_{n=N/2}^{N-1} x(n)\, W_N^{kn},\ k = 0, 1, 2, \ldots, N-1 \tag{2.31a}$$

$$= \sum_{n=0}^{(N/2)-1} x(n)\, W_N^{kn} + W_N^{kN/2} \sum_{n=0}^{(N/2)-1} x(n+\frac{N}{2})\, W_N^{kn} \tag{2.31b}$$

sesuai sifat simetri DFT $W_N^{kN/2} = (-1)^k$, maka persamaan (2.31b) dapat ditulis :

$$X(k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + (-1)^k x\left(n + \frac{N}{2}\right)\right] W_N^{kn},\ k = 0, 1, 2, \ldots, N-1 \tag{2.32}$$

Pada proses perhitungan FFT *radix* 2 DIF memerlukan $(N/2)\log_2 N$ perkalian kompleks dan $N\log_2 N$ penjumlahan kompleks. Terlihat bahwa algoritma FFT lebih cepat dan efisien dibandingkan dengan algoritma DFT.

2.4 Jaringan Syaraf Tiruan (JST)

JST adalah suatu sistem pengolah informasi yang mempunyai karakteristik menyerupai jaringan syaraf biologis manusia. Jaringan saraf tiruan telah dikembangkan dengan menggunakan model matematis untuk menirukan cara kerja jaringan saraf biologis, yaitu dengan berdasarkan asumsi bahwa :

1. Neuron sebagai elemen-elemen sederhana pengolah informasi.
2. Sinyal dilewatkan dari satu neuron ke neuron yang lain melalui hubungan koneksi.
3. Tiap hubungan koneksi mempunyai nilai bobot tersendiri.
4. Tiap neuron mempergunakan fungsi aktifasi terhadap masukan yang diterimanya untuk menentukan sinyal keluaran.

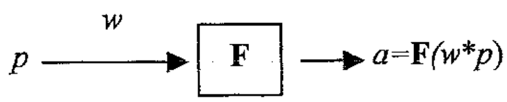
Karakteristik JST ditentukan oleh pola hubungan antar neuron (disebut *architecture*), metode untuk menentukan nilai bobot tiap hubungan (disebut *training*), dan ditentukan oleh fungsi aktifasi.

JST terdiri dari sejumlah besar elemen pengolah sederhana yang disebut neuron, unit atau sel. Tiap neuron terhubung dengan neuron yang lain melalui sambungan komunikasi dimana tiap sambungan mempunyai nilai bobot tersendiri. Nilai bobot ini menyediakan informasi yang akan digunakan oleh jaringan untuk memecahkan masalah.

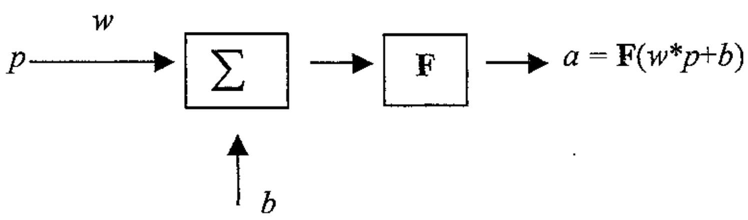
Tiap neuron mempunyai keadaan internal yang disebut level aktifasi (*activation level*) yang terdiri dari fungsi dari masukan yang diterima. Biasanya, suatu neuron mengirimkan nilai aktifasinya ke beberapa neuron yang lain. Sebuah neuron hanya bisa mengirim satu sinyal pada satu saat dan sinyal itu disebarkan ke beberapa neuron yang lain.

2.4.1 Model Neuron

Neuron dengan satu masukan (*single* input) ditunjukkan pada gambar 2.2a dan 2.2b. Neuron pada gambar 2.2a, masukan x dikalikan dengan bobot w, membentuk $w*p$ yang menjadi argumen masukan fungsi aktifasi **F** yang menghasilkan keluaran a. Sedang neuron pada gambar 2.2b mempunyai nilai bias yang berfungsi untuk menyajikan hubungan *input* dan *ouput* lebih mudah daripada tanpa bias. Nilai bias ditambahkan pada perkalian $w*p$ membentuk $w*p+b$ yang menjadi argumen masukan fungsi aktifasi **F** yang menghasilkan keluaran a.



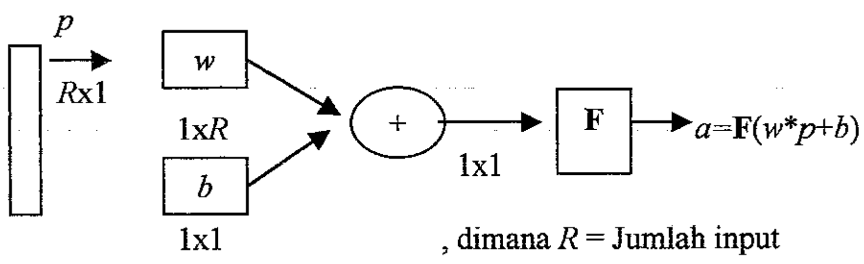
Gambar 2.2a Neuron single input tanpa bias.



Gambar 2.2b Neuron *single input* dengan bias.

Bobot w dan nilai bias b adalah parameter yang bersifat *adjustable* (dapat disesuaikan). Ide dasar dari JST adalah parameter-parameter tersebut *adjusted* (disesuaikan) sehingga jaringan menghasilkan keluaran atau sifat sesuai yang diinginkan. Jadi pada saat pelatihan jaringan parameter bobot dan nilai bias disesuaikan berdasar pekerjaan pengenalan yang dilakukannya.

Neuron dengan banyak masukan (*multiple input*) ditunjukkan pada gambar 2.3. Masukan $p(1)$, $p(2)$, ... , $p(R)$ mempunyai bobot masing-masing $w(1,1)$, $w(1,2)$, ... , $w(1,R)$. Bobot w membentuk suatu baris berdimensi 1xR yang kemudian dikalikan dengan masukan p yang membentuk suatu kolom berdimensi Rx1. Hasil perkalian berdimensi 1x1 ditambahkan nilai bias b, yang kemudian menjadi masukan untuk fungsi aktifasi **F** dan menghasilkan keluaran a.

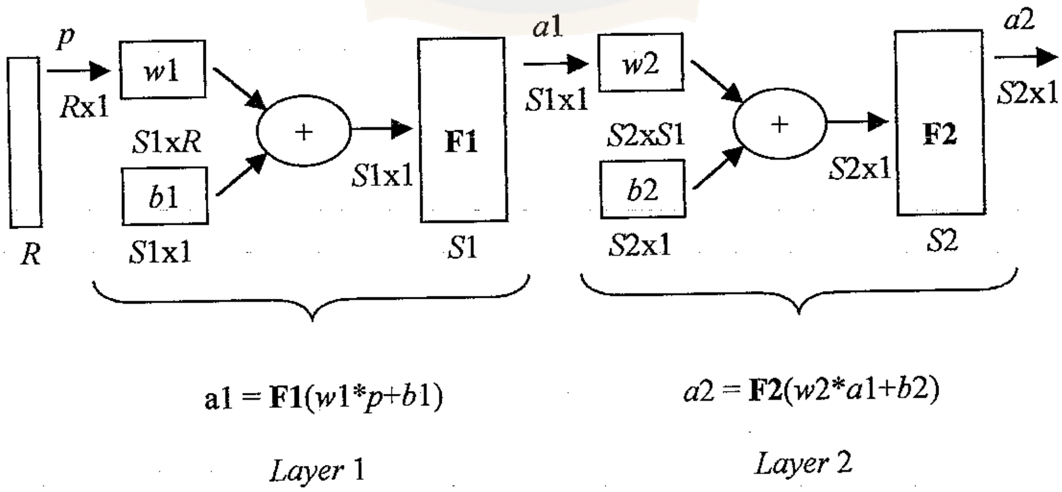


Gambar 2.3 Neuron *R multiple input* dengan bias.

2.4.2 Pola Hubungan Antar Neuron (*architecthure*)

Neuron-neuron dalam JST tersusun ke dalam bentuk lapisan-lapisan (*layers*) dan memiliki pola keterhubungan baik dalam satu atau lebih lapisan. Faktor kunci yang menentukan karakteristik suatu jaringan adalah fungsi aktivasi dan pola keterhubungan antar neuron tersebut. Neuron-neuron yang berada dalam satu lapisan tertentu memiliki fungsi aktivasi dan pola keterhubungan antar neuron yang sama. Dalam menentukan jumlah lapisan dalam suatu jaringan, lapis masukan tidak dihitung, karena lapis masukan tidak melakukan proses perhitungan, hanya mendistribusikan sinyal ke semua neuron di lapis selanjutnya.

Sama halnya dengan model neuron yang mempunyai satu atau lebih masukan, suatu jaringan juga dapat terdiri atas satu atau lebih neuron *layer*. Gambar 2.4 di bawah ini memperlihatkan jaringan dengan dua *layer* dengan *S*1 neuron pada *layer* 1 dan *S*2 neuron pada *layer* 2.



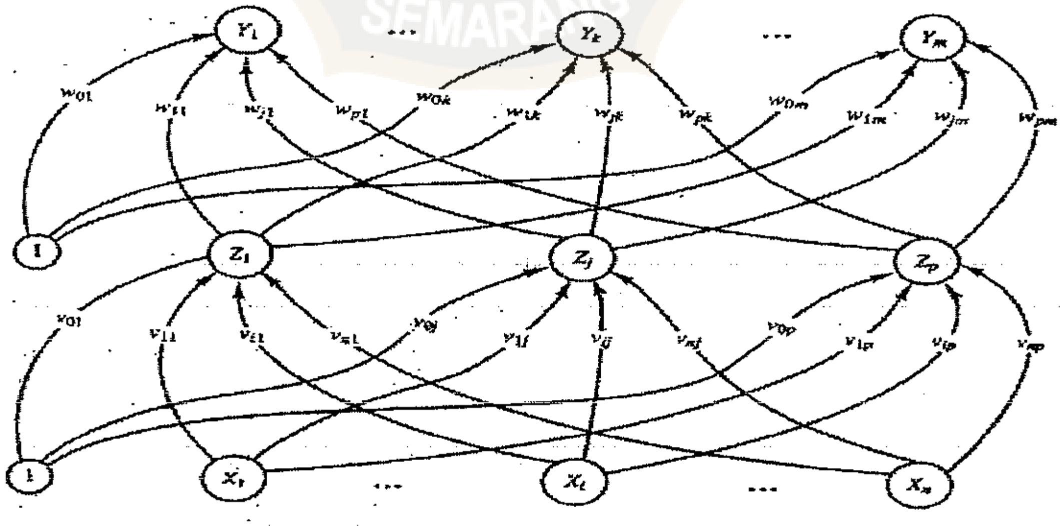
Gambar 2.4 Neuron dengan 2 *layer* dan *R multiple input*.

Layer yang menghasilkan keluaran disebut *output layer*, sedangkan *layer* lainnya disebut *hidden layer* (lapis tersembunyi) . Pada gambar 2.4 diatas layer 2 disebut *output layer*, sedang *layer* 1 disebut *hidden layer.*

2.4.3 JST *Back Propagation* (BP)

Gambar JST BP dengan satu lapis tersembunyi (*hidden layer*) ditunjukkan pada gambar 2.4. Pelatihan JST BP dengan satu lapis tersembunyi ini terdiri dari tiga langkah yaitu :

1. Penghitungan nilai keluaran dari masukan yang diterima (*feedforward*).
2. Penghitungan kesalahan yang disebabkan perbedaan output terhadap suatu referensi.
3. Pembaharuan nilai bobot berdasarkan kesalahan yang terjadi agar kesalahan yang terjadi kemudian dapat semakin kecil.



Gambar 2.5 JST Propagasi Balik dengan satu lapis tersembunyi

Pada saat penghitungan nilai masukan, tiap unit masukan (X_i) menerima sinyal masukan dan meneruskan sinyal ini ke tiap-tiap unit tersembunyi $Z_1, \ldots, Z_p$.

Kemudian tiap unit tersembunyi menghitung nilai aktifasinya dan mengirim sinyal hasil perhitungan tersebut (z_j) ke masing-masing unit keluaran (Y_k). Dimana unit keluaran ini menghitung nilai aktifasinya (y_k) untuk membentuk nilai keluaran yang merupakan respon jaringan tersebut terhadap nilai masukan yang diterima. Pada saat pelatihan, tiap unit keluaran membandingkan nilai aktifasinya (y_k) dengan nilai target (t_k) untuk menghitung nilai kesalahan. Dan berdasar nilai kesalahan tersebut dapat dihitung nilai faktor δ_k (k=1,…,m), δ_k tersebut digunakan untuk mendistribusikan nilai kesalahan dari unit keluaran kembali ke semua unit di lapis sebelumnya (unit tersembunyi yang terhubung dengan unit keluaran). Kemudian dengan cara yang sama dihitung faktor δ_j (j=1,…,p) untuk tiap-tiap unit tersembunyi Z_j.

Setelah semua nilai δ dihitung, nilai bobot untuk semua lapis disesuaikan. Penyesuaian nilai bobot w_{jk} (dari unit tersembunyi Z_j ke unit keluaran Y_k) didasarkan pada faktor δ_k dan nilai aktifasi z_j dari unit tersembunyi Z_j. Penyesuaian nilai bobot υ_{ij} (dari unit masukan X_i ke unit tersembunyi Z_j) didasarkan pada faktor δ_j dan nilai aktifasi x_i dari unit masukan.

2.4.4 Fungsi Aktifasi

Fungsi aktivasi merupakan karakteristik internal neuron yang berfungsi untuk mengubah sinyal masukan yang diterima neuron menjadi sinyal keluaran yang akan dikirim ke neuron lainnya. Pada lapisan yang sama setiap neuronnya akan memiliki fungsi aktifasi yang sama pula.

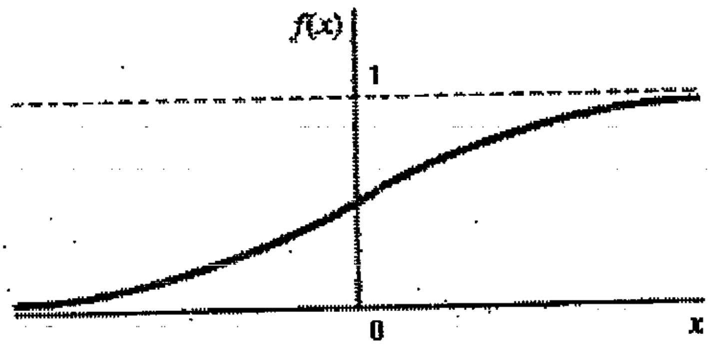
Fungsi aktifasi yang digunakan pada tugas akhir ini ialah fungsi sigmoid biner (*binary sigmoid function*). Fungsi ini merupakan fungsi dengan bentuk kurva S, dan mempunyai jarak antara 0 dan 1 serta didefinisikan sebagai berikut :

$$f_1(x) = \frac{1}{1+\exp(-x)} \tag{2.33}$$

dengan fungsi turunannya :

$$f_1'(x) = f_1(x)[1 - f_1(x)] \tag{2.34}$$

Grafik fungsi sigmoid ini terdapat pada gambar 2.6, terlihat bahwa keluaran dari fungsi tersebut terletak antara 0 dan 1.



Gambar 2.6 Fungsi Sigmoid Biner