BAB 2

TINJAUAN PUSTAKA

2.1 *Forecasting* dan metode *linear*

Forecasting atau peramalan mempunyai sejarah panjang, kepentingan penggunaan peramalan tergantung dari bidang yang menggunakannya, mulai dari bidang bisnis hingga bidang teknik. Kemampuan memprediksi masa depan dengan tepat, adalah merupakan dasar yang tepat dalam membuat keputusan untuk perencanaan, pembuatan jadwal, pembelian, penyusunan *strategy*, pembuatan kebijakan dan dalam operasi *supply chain*.

Forecasting atau peramalan pada awalnya di dominasi oleh metode *linear* (Zang, 2004). Karena metode *linear* mudah di kembangkan dan di implementasikan. Metode *linear* juga relatif mudah untuk di mengerti dan di interpretasikan. Namun, metode *linear* mempunyai keterbatasan yang sangat serius yaitu metode *linear* tidak dapat menangkap satupun hubungan *non linear* dari data. Pada tahun 1980 diadakan suatu kompetisi *forecasting*, dimana metode *linear* yang biasa di gunakan di uji dengan lebih dari 1000 *time series*. Dan hasil nya tidak ada satupun model *linear* yang terbaik. Yang berarti dapat di interpretasikan sebagai kegagalan dari model *linear* dalam melaporkan hubungan *non linear* yang biasa terjadi dalam dunia nyata.

2.2 ANN dalam berbagai bidang

Beberapa tahun terakhir, minat dan ketertarikan terhadap ANN sangat luar biasa, Beragam model ANN telah sukses di implementasikan dan berhasil memecahkan masalah yang kompleks dalam berbagai bidang ,(Zuhaimi 2008, Zang 2004).

Beberapa contoh implementasi ANN dalam berbagai bidang (Zang, 2004) yaitu :

- *Accounting earnings, earnings surprises*
- *Business cycles and recessions*
- *Business failure, bankruptcy, or financial health*

- *Consumer expenditures*
- *Commodity price, option price*
- *Consumer choice, marketing category, market development, market segments, market share, marketing trends*
- *Electricity demand*
- *Exchange rate*
- *Futures trading*
- *GDP growth, inflation, industrial production*
- *International airline passenger volume, tourist demand, travel demand*
- *Inventory control*
- *Joint venture performance*
- *Mutual fund net asset, mutual fund performance*
- *Ozone concentration and level, environmental prediction, air quality*
- *Product demand, product sales, retail sales*
- *Project escalation, project success*
- *Residential construction demand, housing value*
- *Stock return, stock indices, stock trend, stock risk*
- *Traffic*
- *US treasure bond*

2.3 Peramalan dengan ANN

Potensi ANN sebagai alat untuk memprediksi atau meramalkan suatu masalah telah diakui. Banyak peneliti yang telah mengaplikasikan ANN dalam berbagai bidang aplikasi (Zuhaimi, 2008; Zang, 2004), baik bidang ekonomi, keuangan, polusi, penanganan air dan banyak lagi. Salah satu aplikasi utama dari ANN adalah untuk peramalan atau untuk prediksi.

ANN memberikan suatu alternatif bagi para *forecaster*. Terutama dikarenakan sifat struktur *non linear* dari model ANN dalam menangkap hubungan yang sangat kompleks pada masalah di dunia nyata. Karena sifat struktur *non linear* tersebut. ANN mungkin menjadi metode yang sangat bagus untuk aplikasi *forecast* karena ANN dapat menangkap hubungan *non linear* (Zang, 2004; Chouldhary 2008) dan hubungan *linear*. Kemampuan ANN dalam memodelkan *linear time series* sendiri telah dipelajari dan diperkuat oleh banyak peneliti.

2.4 *Artificial Neural Network* (ANN)

ANN adalah model komputasi untuk memproses informasi dan mengidentifikasi pola. ANN berawal dari penelitian bagaimana memodelkan sistem *neural* biologi, khususnya otak manusia (N. Q Hung 2008 et al).

Artificial Neural Network menawarkan pendekatan komputasi yang sedikit berbeda dengan komputasi digital konvensional. Komputer *digital* bekerja secara sekuensial dan dapat mengerjakan komputasi aritmatika dengan sangat cepat. *Neuron* dalam otak manusia sangat lambat tetapi dapat mengerjakan luar biasa banyaknya, pekerjaan komputasi dalam kegiatan sehari-hari, kemudian perasaan, dan membuat keputusan pada situasi yang sulit atau *fuzzy situation*. Tidak seperti komputer konvensional, Otak manusia berisi *neuron* yang luar biasa banyaknya, yang merupakan elemen untuk memproses *biological nervous system* yang bekerja secara paralel. ANN bekerja secara paralel, *distributed information processing structure* yang terdiri dari *processing* elemen yang *interconnected* melalui *uni directional channel* sinyal yang disebut *connection weights*. ANN di modelkan dari *neuron* biologis. Namun ANN lebih simpel karena hanya meniru sedikit saja dari *neuron* biologis.

Beberapa *attribut* utama dari ANN adalah :

- ANN dapat belajar dari contoh *example* dan men *generate* dengan baik data yang belum terlihat.
- ANN dapat digunakan pada situasi dimana input data belum bersih dari kesalahan, belum komplit atau *fuzzy*.
- ANN cepat dan akurat untuk digunakan dalam memprediksi (Iranmanesh, 2008).

Struktur dari neural network secara umum di jelaskan sebagai berikut :

1) *Neuron*

   Neuron adalah dasar dari model *neural network*. *Neuron* adalah saluran komunikasi yang dapat menerima *input* dan menghasilkan *output*. Neuron dapat menerima *input* yang berasal dari *neuron* lain atau pun dari *user*. Begitu juga dengan *output*, *neuron* dapat menghasilkan *output* kepada *neuron* lain atau pun ke *user*.

2) *Neuron connection weight*

   Pada dasarnya *Neuron* selalu terhubung bersama. Namun nilai hubungan/sambungan ini tidak selalu sama, dan pada hubungan/sambungan dapat di berikan nilai *weight* individu. *Weight* inilah yang memberikan kemampuan *neural network* untuk dapat mengenali pola tertentu. Dengan menyesuaikan *weight* nya maka *neural network* akan mengenali pola yang berbeda.

3) *Neuron Layers*

   Layer adalah kumpulan dari *neuron* yang melakukan fungsi yang sama. Terdapat tiga jenis *layer* :

   1) *Layer input* adalah layer dari *neuron* yang menerima *input* dari *user*.
   2) *Layer hidden* adalah layer dari *neuron* yang hanya terhubung ke *neuron* lainnya, dan tidak pernah terhubung langsung dengan *user*.

3) *Layer output* adalah layer dari *neuron* yang mengirimkan *output* kepada *user*.

4) Fungsi Aktivasi

Ketika *neuron* menghasilkan *output*, *neuron* sedang dalam keadaan akan di aktifkan, atau *fire*. *Neuron* akan diaktifkan ketika jumlah dari *input* memenuhi fungsi aktivasi. Fungsi aktivasi ini maksudnya membatasi *range* dari nilai *output*, misal nilai *output* hanya boleh antara 0 dan 1.

Ada beberapa fungsi aktivasi yang biasa di gunakan, antara lain :

- Fungsi Aktivasi *Tanh* (Ismail et al, 2008; Tsong-Wuu Lin, 2009). Jangkauan *output* berkisar antara -1 dan 1.

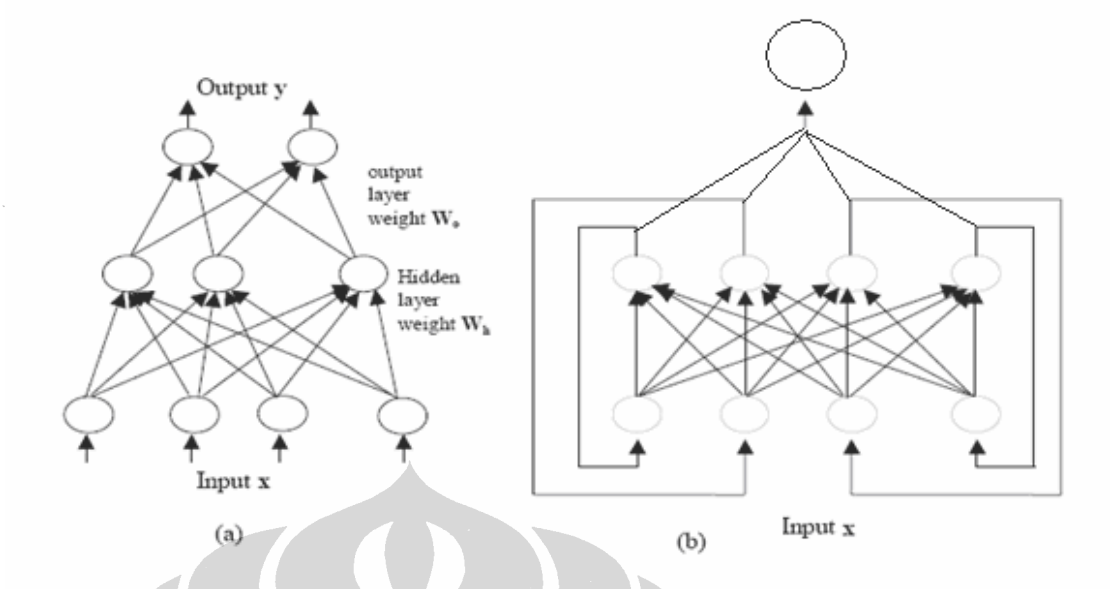
$$f(n) = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}} \quad (2, 1)$$

- Fungsi Aktivasi *Sigmoid*(Zuhaemi et al, 2008; Tsong-Wuu Lin, 2009). Jangkauan *output* berkisar antara 0 dan 1.

$$f(n) = \frac{1}{1 + \exp^{-n}} \quad (2,2)$$

5) Arsitektur *Neural Network*

Neural Network arsitektur mewakili konfigurasi mengindikasi bagaimana unit di kelompokan bersama dan interkoneksi antara unit tersebut. Konfigurasi ini dapat di kelompokan menjadi dua kategori umum yaitu *feed forward* dan *Recurrent*. Arsitektur ini dapat di lihat pada gambar 2.1 di bawah ini.



(Kamruzzaman, 2006)

Gambar 2.1 a) *Feedforward* arsitektur (b) *Recurrent* arsitektur

2.5 GMDH (*Group Method of Data Handling*)

GMDH pertama kali di kembangkan oleh G.Ivakhnenko pada tahun 1967. GMDH pada awal kehadirannya dikembangkan sebagai *rival method* dari *stockhastic approximation*, sejatinya GMDH di kembangkan untuk mengestimasi *regresi polynomial* yang kompleks. Metode GMDH membangun model *polynomial regresi* hirarki untuk memodelkan hubungan yang sangat kompleks dari *input* dan *output*. Pada GMDH model yang di bentuk di *generate* dari data (Miroslav Šnorek et al, 2007). *Domain expert* tidak ambil bagian dalam menentukan hubungan (Alexandr Kiryanov, 2008). GMDH dapat digunakan untuk klasifikasi, pembuatan *cluster*, dan *forecasting* (Gregory Ivahnenko, 2008; Onwubolu, 2007). GMDH dapat dikatakan gabungan antara metode statistik dan metode *neural network* (Onwubolu et al, 2007).

GMDH telah banyak di gunakan dalam berbagai bidang seperti dalam bidang kedokteran, biologi, manufaktur, ekologi dan sistem lingkungan, psikologi, ekonomi dan lain sebagainya. Untuk peramalan *multivariate* GDP menggunakan Algoritma COMBI, *performance* GMDH lebih buruk dibandingkan model ANN yang lain (Bulgakova, 2007) akan tetapi hasil peramalan GMDH lebih baik

daripada menggunakan model ARCH (Yue He et al, 2008) . Untuk peramalan *univariate / time series* GMDH menunjukan *performance* yang baik, dari hasil perbandingan *moving average* (Howland et al, 2003). GMDH juga telah di uji untuk peramalan rata-rata air (Zubov D.A, 2008).

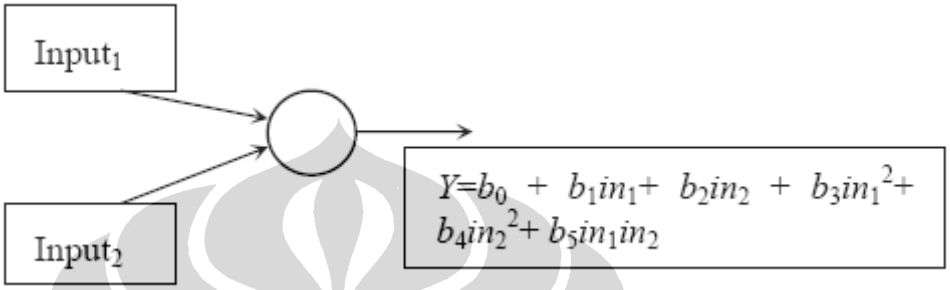
Algoritma GMDH yang dipakai pada karya akhir ini mengikuti (Morrison S. Obeng et al, 2008) menghasilkan dan menguji semua kombinasi *input-ouput* ke sistem. Koefisien dari elemen ditetapkan menggunakan metode regresi. *Threshold* di tentukan pada tiap level untuk menentukan apakah elemen pada layer menghasilkan hasil yang dapat di terima. Jika hasil dari elemen berada dalam *threshold*, maka akan di lanjutkan ke layer berikutnya. Elemen dan variabel yang kurang berguna dalam memprediksi akan terfilter dengan sendirinya. Pada metode ini algoritma memilih optimal set dari variabel *input*, lalu memilih *degree* dari *nonlinearity* pada model akhir, dan memilih struktur dan *degree* dari interaksi pada model akhir. Ada beberapa keuntungan yang dapat diperoleh apabila menggunakan metode ini :

1) Hanya membutuhkan sedikit *set training.*
2) Menghindari atau mengurangi perhitungan komputasi yang sulit.
3) *Inputs/function* dari input yang hanya memberikan sedikit pengaruh kepada *output*, secara otomatis akan tersisih.

Algoritma GMDH sering mengkombinasikan elemen *linear* dan *non linear* dengan tujuan untuk mencari hubungan antara *input-output* yang paling baik dari sistem. Pada algoritma (Morrison S. Obeng et al, 2008), *linear* dan parabola (kuadrat dari *input*) dipilih. Setiap neuron memiliki dua *input*. Ini akan menghasilkan 6 *term* Persamaan kuadrat *polynomial* (Morrison S. Obeng et al, 2008) berikut ini digunakan sebagai persamaan untuk menghasilkan *output* dari *hidden neuron*.

$$Y=b_0 + b_1in_1+ b_2in_2 + b_3in_1^2+ b_4in_2^2+ b_5in_1in_2 \quad (2.1)$$

Pada (2.1), Y adalah *output* dari *neuron*, b_n adalah koefisien dari tiap *term*, dan in_1 dan in_2 adalah dua buah *input* yang terpilih untuk *neuron* tersebut, b_0 koefisien di tambahkan sebagai konstanta penyeimbang dari fungsi yang dapat digunakan untuk penyeimbang *noise* atau *error* dari sistem.



Gambar 2.2 Diagram dari *Neuron* dasar.

Diagram dari gambar 2.2 menjelaskan dari persamaan 2.1

Ada beberapa hal yang perlu di perhatikan dalam pembentukan model GMDH, antara lain :

1) Melatih *network.*

   Aspek yang paling penting dari *network* adalah bagaimana *network* memilih *neuron* dan *layer* yang nantinya akan ditempatkan dalam *final network*. Ukuran *network,* kecepatan, dan keakuratan dari *network* dipengaruhi oleh hal tersebut. Berikut di jelaskan bagaimana menghitung koefisien dari sebuah *neuron,* memilih *set neuron* untuk sebuah *layer*, dan memilih *layer* untuk *final network.*

2) Menghitung koefisien

   Menghitung koefisien dari *neuron* barangkali merupakan hal yang paling sulit dari keseluruhan urutan *design*. Metode berikut digunakan oleh (Morrison S. Obeng et al, 2008) . *Training set* dari *user* (komponen *input*) akan menjadi bentuk matriks. Algoritma secara sistematik meminta dua *input set* untuk membentuk *term* pada (2.1),

hasilnya akan berupa persamaan *matrix* (2.2). Pada kasus ini s adalah jumlah dari *sampel*, $i_{m,n}$ mewakili *input* n pada *sampel* m, dan ym adalah *output* dari *input sampel* m. Pada *term input* untuk *neuron*, $i_1=1$, $i_2=in_1$, $i_3=in_2$, $i_4=in_1^2$, $i_5=in_2^2$ dan $i_6=in_1in_2$. Langkah berikut nya adalah mengalikan kedua bentuk persamaan dengan *transpose* dari *matrix input*. Mengalikan kedua sisi (2.3) dengan *inverse* dari 6x6 *matrix input* akan menghasilkan 6 koefisien yang diinginkan. Jika tidak ada *inverse* dari *input matrix*, persamaan tidak akan bertemu (*konvergence*), karena tidak ada cukup informasi atau cukup variasi informasi untuk menghitung dari kombinasi *term* yang diberikan.

$$\begin{bmatrix} i_{1,1} & i_{1,2} & i_{1,3} & i_{1,4} & i_{1,5} & i_{1,6} \\ i_{2,1} & i_{2,2} & i_{2,3} & i_{2,4} & i_{2,5} & i_{2,6} \\ i_{3,1} & i_{3,2} & i_{3,3} & i_{3,4} & i_{3,5} & i_{3,6} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ I_{s,1} & i_{s,2} & i_{s,3} & i_{s,4} & i_{s,5} & i_{s,6} \end{bmatrix} \begin{bmatrix} b_0 \\ b_2 \\ b_3 \\ \vdots \\ b_s \end{bmatrix} = \begin{bmatrix} y_0 \\ y_2 \\ y_3 \\ \vdots \\ y_s \end{bmatrix} \tag{2.2}$$

$$\begin{bmatrix} \sum_{j=1}^{s} i_{j,1}^2 & \sum_{j=1}^{s} i_{j,1}i_{j,2} & \sum_{j=1}^{s} i_{j,1}i_{j,3} & \sum_{j=1}^{s} i_{j,1}i_{j,4} & \sum_{j=1}^{s} i_{j,1}i_{j,5} & \sum_{j=1}^{s} i_{j,1}i_{j,6} \\ \sum_{j=1}^{s} i_{j,1}i_{j,2} & \sum_{j=1}^{s} i_{j,2}^2 & \sum_{j=1}^{s} i_{j,2}i_{j,3} & \sum_{j=1}^{s} i_{j,2}i_{j,4} & \sum_{j=1}^{s} i_{j,2}i_{j,5} & \sum_{j=1}^{s} i_{j,2}i_{j,6} \\ \sum_{j=1}^{s} i_{j,1}i_{j,3} & \sum_{j=1}^{s} i_{j,2}i_{j,3} & \sum_{j=1}^{s} i_{j,3}^2 & \sum_{j=1}^{s} i_{j,3}i_{j,4} & \sum_{j=1}^{s} i_{j,3}i_{j,5} & \sum_{j=1}^{s} i_{j,3}i_{j,6} \\ \sum_{j=1}^{s} i_{j,1}i_{j,4} & \sum_{j=1}^{s} i_{j,2}i_{j,4} & \sum_{j=1}^{s} i_{j,3}i_{j,4} & \sum_{j=1}^{s} i_{j,4}^2 & \sum_{j=1}^{s} i_{j,4}i_{j,5} & \sum_{j=1}^{s} i_{j,4}i_{j,6} \\ \sum_{j=1}^{s} i_{j,1}i_{j,5} & \sum_{j=1}^{s} i_{j,2}i_{j,5} & \sum_{j=1}^{s} i_{j,3}i_{j,5} & \sum_{j=1}^{s} i_{j,4}i_{j,5} & \sum_{j=1}^{s} i_{j,5}^2 & \sum_{j=1}^{s} i_{j,5}i_{j,6} \\ \sum_{j=1}^{s} i_{j,1}i_{j,6} & \sum_{j=1}^{s} i_{j,2}i_{j,6} & \sum_{j=1}^{s} i_{j,3}i_{j,6} & \sum_{j=1}^{s} i_{j,4}i_{j,6} & \sum_{j=1}^{s} i_{j,5}i_{j,6} & \sum_{j=1}^{s} i_{j,6}^2 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{s} i_{j,1}y_j \\ \sum_{j=1}^{s} i_{j,2}y_j \\ \sum_{j=1}^{s} i_{j,3}y_j \\ \sum_{j=1}^{s} i_{j,4}y_j \\ \sum_{j=1}^{s} i_{j,5}y_j \\ \sum_{j=1}^{s} i_{j,6}y_j \end{bmatrix} \tag{2.3}$$

3) *Neuron*

Salah satu keuntungan kenapa hanya mengambil dua buah *input* untuk satu *neuron* adalah untuk meningkatkan kemungkinan menemukan hubungan dekat antara *input* dan *output*. Jumlah dari *subset* S yang berisi E elemen dapat dibuat dari *set* N elemen, dapat di turunkan dari rumus berikut :

$$S = \frac{N!}{(N-E)!E!} \quad (2.4)$$

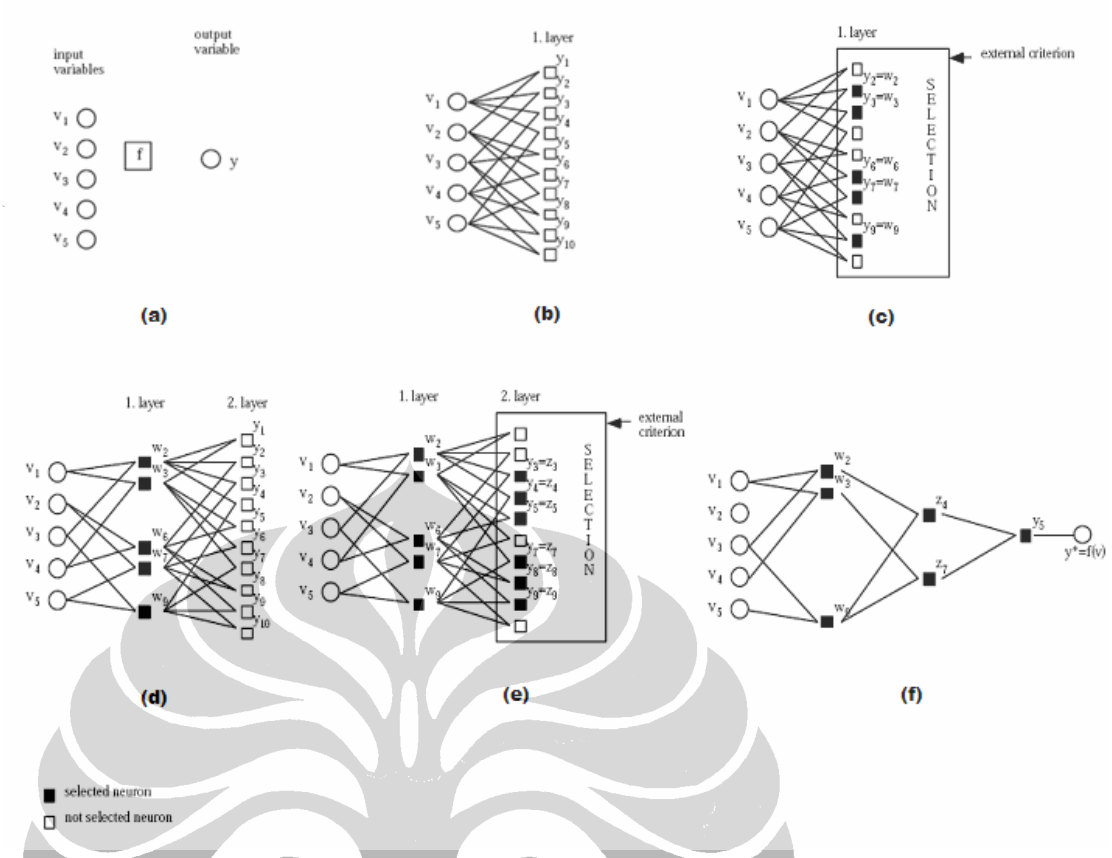
S adalah *subset*, N adalah jumlah komponen *input*, dan E adalah 2, karena setiap *neuron* hanya menggunakan 2 *input*. Sebagai contoh apabila ada 5 komponen *input* maka S = 10 karena 5 ! dibagi (3!x2!) = 4 x 5 / 2 = 10.

Untuk menghitung *best neuron* yang akan di ambil untuk *layer* berikut nya digunakan persamaan berikut ini

$$MSE = \frac{1}{s}\sum_{i=1}^{s}(y_{d,i} - y_{c,i})^2 \quad (2.5)$$

Dimana $y_{d,i}$ adalah y yang didapat hasil perhitungan koefisien dan $y_{c,i}$ adalah y dari *sampel*. Hasil perngurangan di kuadratkan agar tidak negatif.

Setelah memperhatikan hal-hal penting dalam pembentukan model GMDH, berikut di jelaskan proses pembentukan arsitektur GMDH pada gambar 2.3, dari pembentukan layer 1 hingga layer terakhir.



(Lemke et al, 2000)

Gambar 2.3 Arsitektur *Feed forward* GMDH

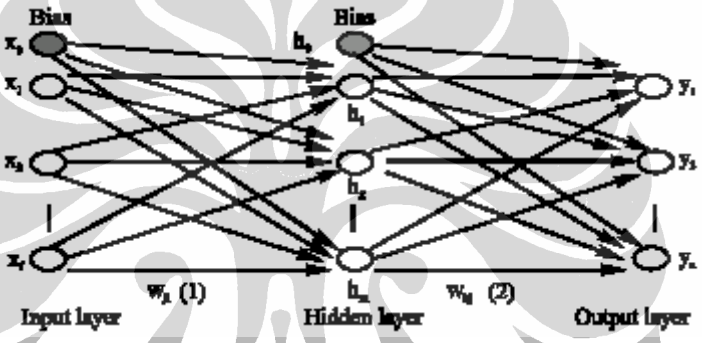
Pada gambar 2.3 diatas, pada gambar (a) ada lima buah input kemudian pada gambar (b) dari 5 input tadi akan terbentuk layer 1 yang terdiri dari 10 neuron. Kemudian pada gambar (c) 10 neuron yang terbentuk pada layer 1 akan di pilih 5 neuron terbaik. Kemudian pada gambar (d) dari 5 neuron terbaik akan menghasilkan layer 2 yang terdiri dari 10 neuron. Kemudian pada gambar (e) 10 neuron pada layer 2 akan di pilih 5 neuron terbaik yang akan menghasilkan layer 3 yang terdiri dari 10 neuron. Kemudian pada gambar (f) dari 10 neuron di ambil 1 yang terbaik. Kemudian di urutkan hubungan dari layer 3 ke layer 2 hingga layer 1. Sehingga terbentuklah arsitektur GMDH, layer 1 terdiri dari 3 neuron, layer 2 terdiri dari 2 neuron dan layer 1 terdiri dari 1 neuron.

2.6 Feedforward backpropagation Neural Network

Feed forward Backpropagation dapat dijadikan *tool* untuk meramal *time series* GNP (Giovanni, 2008). Pada bagian berikut ini akan dijelaskan mengenai *Feedforward backpropagation* yang meliputi: konsep, pelatihan serta tahapan-tahapan dalam *Feedforward backpropagation*.

Feedforward Neural Network

Hampir beberapa dekade, beragam jenis ANN dikembangkan untuk memecahkan masalah. Namun yang lebih terkenal dan sukses dalam diimplementasikan dalam peramalan adalah *feedforward neural network.*



(Ismail et al, 2008)

Gambar 2.4 *Feedforward Neural Network*

Gambar 2.4 memperlihatkan arsitektur dari 3 buah *layer* dalam *feedforward neural network* yang terdiri dari *input layer* yang berisi 4 *neuron*, *hidden layer* yang berisi 4 *neuron*, dan *output layer* yang berisi 1 *neuron*. *Neuron* pada *input layer* berhubungan dengan variabel prediksi (x) yang di yakini berguna untuk peramalan *dependent variabel* (y) yang merupakan *output neuron*. *Neuron* pada *hidden layer* terhubung dengan *input layer* dan *output layer*, dan merupakan kunci dalam mempelajari pola dari data dan memetakan hubungan relasi dari variabel *input* ke variabel *output* untuk proses lebih jauh untuk membuat peramalan. Pada *feedforward* ANN. Alur informasi adalah satu arah dari *input layer* ke *hidden layer* kemudian ke *output layer* tanpa *feedback* dari *output layer* (Giovanis, 2008).

Backpropagation training

Sebelum *neural network* dapat di gunakan untuk peramalan. *Neural network* harus di latih. B*ackpropagation* merupakan Algoritma pelatihan *neural network* yang paling banyak di gunakan (Kamruzzaman, 2006; Bulgakova et al, 2007). Pertama kali dikembangkan oleh Verbos(1974) dan Rumelhart (1986) (Zang, 2004). Ide dasar dari pelatihan *backpropagation* adalah menggunakan pendekatan *gradient-descent* untuk menyesuaikan dan menetapkan *weight* sehingga fungsi *error* seperti misalnya fungsi SSE (*Sum of Squared Error)* dapat minimalisasi (Zang, 2004; Kamruzzaman, 2006, N.Q Hung et al, 2008; Iranmanesh et al, 2008; Rohitash Chandra, 2008)

Langkah-langkah dalam *Feedforward backpropagation*

Adapun langkah-langkah dalam algoritma *Feedforward Backpropagation* adalah :

- *Forward Propagation*

Pada proses ini *input* di salurkan ke dalam *network*, dan tiap *layer* akan mengeluarkan *output*. *Output* dari satu *layer* akan menjadi *input layer* bagi *layer* berikutnya. *Input* pada *layer input* disalurkan ke *hidden layer*, *output* dari *hidden layer* merupakan *input* untuk *output layer*. Fungsi *sigmoid* dipilih sebagai fungsi aktivasi.

- *Backpropagation*

Merupakan proses penghitungan nilai sensitivitas untuk tiap layer. Dimana sensitivitas untuk layer m dihitung dari sensitivitas pada layer m+1, penghitungan sensitivitas berjalan mundur.

- *Weight Update*

Menyesuaikan nilai parameter bobot (W) dan bias (b) dengan menggunakan pendekatan *steepest descent*.

Apabila semua nilai *input* bernilai 0 maka *weight matrix* dari satu *layer* ke *layer* berikutnya tidak dapat berubah untuk pola ini. Oleh karena itu di berikan bias yang berupa *konstant output* yang bernilai 1. Langkah diatas dilakukan berulang hingga n Epoch, atau Nilai *Error* melebihi 0,001. *Backpropagation* dengan *least mean square* memang menjamin penyelesaian dengan minimum *mean square error* selama *learning rate*-nya tidak terlalu besar. Kekurangannya adalah bila *learning rate*–nya kecil, maka pencapaian nilai konvergennya lambat, sedangkan bila *learning rate* nya besar, pencapaian nilai konvergensinya cepat namun ada bahaya osilasi yang dapat mengakibatkan nilai minimum global tidak tercapai. Untuk mengatasi hal ini maka digunakanlah variasi *backpropagation* sebagai berikut :

a. Momentum

Metode ini bekerja dengan tujuan untuk menghaluskan osilasi yang terjadi. Momentum ini akan ditambahkan pada persamaan *weight matrix* dan bias.

b. Variable Learning Rate

Metode ini bekerja dengan berusaha menaikkan *learning rate* bila menjumpai permukaan yang datar dan kemudian menurunkan *learning rate* bila terjadi peningkatan *slope.*

2.7 *Elman Recurrent Neural Network.*

Untuk peramalan *time series short term* (Kifah tout et al, 2008), *performance elman recurrent neural* kalah dibandingkan dengan MLP, sedangkan untuk peramalan *time series long term*, *Elman* lebih baik di bandingkan MLP. Penjelasan mengenai model Elman recurrent neural network dipecah menjadi beberapa tahap, yaitu :

Recurrent Neural Network.

Recurrent Neural Network mempunyai struktur dan algoritma pelatihan yang lebih kompleks di bandingkan *Feedforward Neural Network* (Xiaolin Hu et al, 2008). Pada *recurrent neural network, output* dari *network* digunakan kembali sebagai *input* dengan mengirimkan kembali sebagai *input network*. *Recurrent network* mungkin dapat di bagi menjadi dua jenis (Ali Gulbag et al, 2004**)**:

a. *Full recurrent network*

   Network ini adalah *network* yang mempunyai hubungan *forward* dan *backward*, pelatihan dilakukan dalam setiap hubungan baik *forward* maupun *backward*, contoh nya *Hopfield Network.*

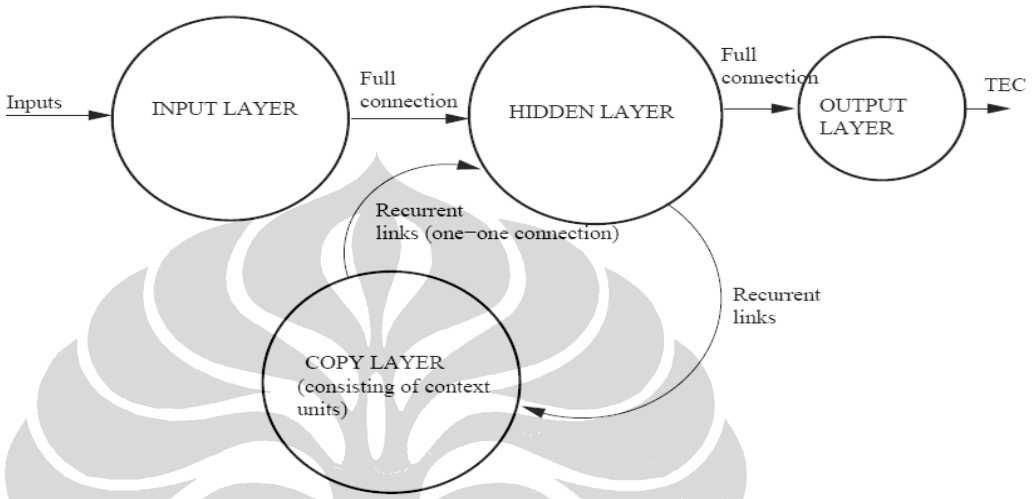
b. *Partial recurrent network*

   Pada *network* ini ditambahkan *layer* konteks sebagai tambahan *layer* proses. Biasanya *network* ini merupakan *feedforward neural network*. Pelatihan di lakukan pada hubungan *forward*. Sedangkan hubungan *backward* dari *output layer* ke *input layer* tidak dapat di lakukan pelatihan. *Layer* konteks digunakan untuk mengingat status terakhir dari *hidden layer*. *Output* dari *network* tergantung dari status sebelum maupun status *network* pada saat ini. Kemampuan dalam mengingat status terakhir, menjadikan *network* ini memiliki memori yang dinamis. *Network* yang memiliki struktur dasar dan paling mudah digunakan diantara *recurrent neural network* adalah *Elman Recurrent Neural NetworkNetwork*.

Elman Recurrent Neural Network

Elman merupakan *recurrent arficial neural network* yang sering digunakan (Cernansky, 2008). *Elman network* mempunyai empat buah *layer* yaitu : *layer input, layer hidden, layer output* dan *layer* konteks. *Input* dan *output layer* berinteraksi dengan informasi di luar. *Input layer* menerima informasi dari luar Hubungan *weight* dari *hidden* ke *input* l*ayer* adalah *konstant, irreversible*. *Elman recurrent neural* disebut *partial recurrent neural network* karena *recurrent weight* adalah tetap.

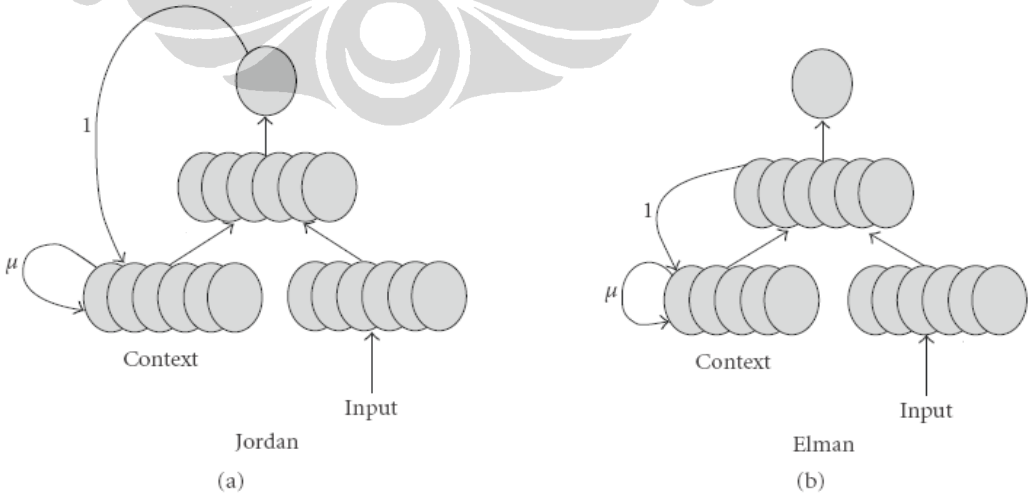
Arsitektur *Elman* hampir sama dengan arsitektur *Feedforward backpropagation*, namun di tambah *layer* konteks untuk menampung hasil *output* dari *hidden* layer gambar 2.6 (J. B. Habarulema, 2009; V. R. Mankar, 2009). Fungsi aktivasi dari *hidden layer* dan *output layer* menggunakan Fungsi *Sigmoid*.



(J. B. Habarulema, 2009)

Gambar 2.5 Arsitektur *Elman Neural Network*.

Terlihat dari gambar 2.5, hasil dari *hidden layer* masuk kembali ke *layer* konteks. *Weight* dari *layer* konteks ke *hidden layer*, berubah sama seperti *weight* dari *layer input* ke *hidden layer*.



(V. R. Mankar, 2009)

Gambar 2.6 Perbedaan antara *Elman* dengan *Jordan Neural Network*

Perbedaan mendasar antara model Elman dan Jordan gambar 2.6 adalah, pada model Elman hasil dari hidden layer akan di kopi ke dalam konteks layer, sedangkan pada model elman hasil dari output yang akan di kopi ke dalam konteks layer

2.8 Moving Average dan Regresi Linear

Moving average metode kuantitatif (Scott Nadler et al, 2008) yang paling mudah digunakan untuk peramalan. Metode ini merupakan metode yang istimewa, karena merupakan *smoothing model* yang mana membuat peramalan lebih akurat. Semakin lama dan panjang data di peroleh maka hasil peramalan akan semakin baik. Persamaan untuk menghitung *moving average* dapat dilihat pada persamaan berikut :

$$MA_n = \frac{\sum_{t=1}^{n} D_t}{n} \quad (2, 6)$$

Dimana n adalah periode dalam *moving average* dan D_t adalah data pada periode t.

Regresi linear adalah metode untuk peramalan yang digunakan operasional di lingkungan yang stabil dan mempunyai akses pada data *history*. Analisa regresi digunakan untuk menguji hubungan antara dua atau lebih variabel. Tujuan dari regresi linear adalah mencari garis lurus yang dapat memprediksikan X dan Y ketika X dan Y diketahui (Nadler et al, 2008).

2.9 Model Keynesian

2.9.1 Sejarah dan Filsafat Teori Keynes

Pada mulanya (hanani, 2004), selama lebih dari 100 tahun setelah revolusi industri yang dimulai di Inggris, negara-negara barat mengalami pertumbuhan ekonomi

yang pesat. Keberhasilan ini merupakan keberhasilan penerapan teori klasik yang mengandalkan sistem *laissez-faire*. Namun, pada tahun 1930-an, negara-negara tersebut mengalami depresi dan pengangguran yang hebat dan berkepanjangan. Dalam keadaan demikian kaum Klasik dan Neo-Klasik tidak berdaya untuk memberi pemecahan permasalahan yang dihadapai dalam perekonomian masyarakat Kaum sosialis di negara tersebut mengatakan bahwa penyebab depresi itu adalah kesalahan pada sistem perekonomian itu sendiri, yaitu sistem *laissez faire* atau *liberalisme* atau *kapitalisme* . Kaum sosialis berpandangan, selama suatu negara mempercayakan pengelolaan perekonomian pada para produsen swasta yang per definisi hanya bertujuan mengejar keuntungan sebesar-besarnya untuk mereka pribadi, maka depresi, pengangguran, dan juga inflasi akan tetap menjadi penyakit perekonomian yang menghantui dari waktu ke waktu. Oleh karenanya kaum sosialis mengusulkan perombakan sistem perekonomian menjadi sistem sosialis, yaitu sistem di mana faktor-faktor produksi tidak bisa dimiliki oleh pengusaha swasta, tetapi hanya dimiliki oleh masyarakat (negara). Semua kegiatan produksi dikuasai negara, yang secara teoritis, akan mengutamakan kepentingan masyarakat di atas kepentingan pribadi/golongan. Motif mengejar keuntungan tidak lagi sebagai motif utama seperti pada sistem kapitalis. “Obat” semacam itu ternyata dianggap terlalu radikal, sehingga orang-orang di negara-negara Barat yang telah lama terbiasa dengan kebebasan berusaha tidak dapat menerima begitu saja. Mengubah sistem seperti itu berarti mengubah kebiasaan dan cara hidup yang sudah mendarah daging pada mereka. Mereka menghendaki *obat yang tidak terlalu pahit* yang dapat menolong memecahlan masalah perekonomian mereka. Dalam situasi demikian **John Maynard Keynes (1883-1946)** muncul menawarkan suatu pemecahan yang merupakan “jalan tengah”. Keynes menawarkan untuk meninggalkan pemikiran kaum Klasik murni. Keynes berpendapat, untuk mengatasi masalah krisis ekonomi tersebut, Pemerintah harus melakukan lebih banyak campur tangan secara aktif dalam mengendalikan perekonomian nasional. Kegiatan produksi dan pemilikan faktor-faktor produksi masih dapat dipercayakan kepada swasta, tetapi Pemerintah wajib melakukan kebijakan-kebijakan untuk mempengaruhi perekonomian. Misalnya, dalam masa depresi Pemerintah harus bersedia melakukan kegiatan-kegiatan yang langsung

dapat menyerap tenaga kerja yang tidak dapat bekerja pada swasta, walaupun hal ini dapat menyebabkan defisit dalam anggaran belanja negara. Dalam hal ini Keynes tidak percaya pada sistem liberalisme yang mengoreksi diri sendiri, untuk kembali pada posisi *full employment* secara otomatis. Full employment hanya bisa dicapai dengan tindakan-tindakan tertencana, bukan datang dengan sendirinya. Inilah inti dari ideologi "keynesianisme"**.** Pemikiran-pemikiran Keynes tersebut dituangkan dalam bukunya yang berjudul ***"The General Theory of Employment, Interest, and Money (1936)".***

2.9.2 Model prakiraan ekonomi Indonesia berdasarkan tipe Keynesian (Sodikin, 2006).

Model ini merupakan model simultan ekonometrik yang dapat menjelaskan fenomena ekonomi sesuai dengan teori ekonomi, yaitu bagaimana indikator ekonomi. mempengaruhi PDB. Berdasarkan tipe model Keynesian, terdapat beberapa fungsi yang akan digunakan sebagai model prakiraan ini.

Struktur Model::

GDP = PC + PCF + GC + GCF + EX-IM+ S

PC = f (GDPV/PGDP, PC(-1), PPC)

CF = f ((LO+IMEQ)/PCF, GDP(-1), FI, ER)

IM = f (GDP, IM(-1), PIM/PGDP)

PGDP = f (M1, M2, PEX, P_RICE, P_ GASOL, ER)

XNO = f (GDP-USA, GDP-Japan, PXNO, ER, IM)

EX = XNO + XO

PPC = f (PGDP)

Model ini memusatkan pada sisi permintaan, di mana PDB adalah total konsumsi, investasi (CF) dan ekspor (EX) dikurangi impor (IM). Konsumsi terdiri dari konsumsi swasta (PC) dan konsumsi pemerintah (GC). Perubahan stok diperlakukan sebagai item penyeimbang. Ada empat fungsi utama, yaitu konsumsi, investasi, ekspor, dan fungsi impor. Konsumsi sangat berperan dalam perekonomian hingga kini, oleh karenanya sangat mempengaruhi pertumbuhan perekonomian. Pertumbuhan konsumsi swasta tergantung pada pendapatan dan harga. Investasi adalah fungsi PDB, posisi kredit perbankan dalam rupiah dan

valuta asing, impor mesin dan peralatan, Investasi asing, nilai tukar, serta harga investasi (deflator investasi). Ekspor, yang berhubungan erat dengan produksi dan kegiatan jasa, dibagi ke dalam dua kategori dalam model, ekspor bukan minyak dan gas, dan ekspor minyak dan gas (exogenous). Ekspor bukan minyak dan gas adalah fungsi dari PDB

Jepang, PDB AS, harga ekspor atas nonminyak, nilai tukar, dan juga impor terutama untuk bahan baku. Impor mempunyai peran penting dalam perekonomian untuk memperbesar sektor produksi, maupun untuk menyediakan input dan bahan baku seperti halnya barang modal. Impor dalam model ini adalah fungsi dari PDB dan harga impor. Di samping fungsi ini, ada beberapa fungsi lain tentang harga. Harga konsumsi swasta (PPC), konsumsi pemerintah (PGC), investasi (PCF), ekspor (PXO) dan impor (PIM) tergantung pada fungsi deflator PDB (PGDP). Harga impor juga dipengaruhi oleh nilai tukar. Tetapi PGDP adalah fungsi harga dari beras, bensin, suplai uang (M1) dan PDB itu sendiri. M1 adalah pendekatan dari tingkat bunga. Dalam memprakirakan dibutuhkan pertimbangan beberapa asumsi laju pertumbuhan beberapa variabel exogenous. Dalam memperkirakan PDB tentunya akan mempertimbangkan asumsi-asumsi penampilan indikator ekonomi dari sebelumnya hingga kondisi saat ini.

2.10 Peramalan menggunakan model Vector Auto Regresi dan General to Specific Modelling.

Selain metode yang telah disebutkan diatas peramalan makro ekonomi juga dapat dilakukan menggunakan model yang lain (Siswantoro, 2008), yaitu menggunakan model *Vector Auto Regresi* dan *General to Specific Modelling.* Peramalan menggunakan model ini menggunakan 4 variabel input yaitu *Term of Trade* (TOT), *RER* (nilai tukar riil), *CPI* (indeks harga konsumen) dan RGDP (produk domestik bruto riil). Sumber data nya di peroleh dari BPS dan *International Financial Statistic* (IFS) terbitan *International Monetery Fund* (IMF). Pada penelitian yang dilakukan (Siswantoro, 2008) terbukti kedua model tersebut mampu meramalkan trend musiman untuk variabel RGDP, terutama *General to Specific Modelling,* model ini mampu meramalkan lebih baik dari pada model *Vector Auto Regresi* (VAR). Akan tetapi hasil peramalan (Siswantoro, 2008) tidak