BAB 2

KAJIAN PUSTAKA

2.1. Visi Komputer (*Computer Vision*)

Visi komputer (*Computer Vision*) adalah salah satu teknologi yang paling banyak dipakai pada zaman ini. Teknologi visi komputer ini merupakan salah satu bidang dari teknologi *Artificial Intelligence*. Visi komputer juga merupakan dan kumpulan dari metode-metode untuk mendapatkan, memproses, menganalisis suatu gambar atau dalam arti lain visi komputer, merupakan kumpulan metode-metode yang digunakan untuk menghasilkan angka-angka atau simbol-simbol yang didapat dari gambar yang diambil dari dunia nyata agar komputer dapat mengerti apa makna dari gambar tersebut. Inti dari teknologi visi komputer adalah untuk menduplikasi kemampuan penglihatan manusia kedalam benda elektronik sehingga benda elektronik dapat memahami dan mengerti arti dari gambar yang dimasukkan (Milan Sonka, Vaclav Hlavac and Roger Boyle - 2008).

Pemahaman gambar pada komputer ini dilakukan dengan menguraikan informasi simbolis dari data gambar dengan menggunakan model yang dibangun dengan bantuan geometri, fisika, statistika, dan teori serta metode-metode lainnya (David A. Forsyth dan Jean Ponce, 2003). Visi komputer juga dideskripsikan sebagai suatu teknologi untuk mengotomatisasi dan mengintegrasikan berbagai proses dan represetasi untuk menghasilkan presepsi penglihatan (Dana H. Ballard dan Christopher M. Brown, 1982).

Dalam disiplin ilmu, visi komputer berkaitan dengan teori-teori dibalik sistem buatan yang mengekstrak informasi dari gambar. Informasi yang diekstrak dari gambar dapat berupa data-data yang berbeda-beda, seperti urutan jalannnya video, intesitas cahaya, atau prespektif dari sudut gambar yang berbeda-beda.

Dalam disiplin teknologi, visi komputer mengusahakan cara agar teori-teori dan model dapat diterapkan untuk pembangunan sistem pada sistem komputer. Contoh beberapa aplikasi yang menggunakan visi komputer seperti, alat navigasi dan kontroler.

Visi komputer sekarang ini sering digunakan untuk mendeteksi wajah pada gambar (*face detection*), mengenali ekspresi wajah(*facial expression recognition*) dan dalam prakteknya sering digunakan bersama dengan jaringan syaraf tiruan (*artificial neural network*).

2.1.1. Pengolahan Gambar (*Image Processing*)

Pengolahan gambar dibagi menjadi 3, *low level*, *intermediate level* dan *high level image processing*. Pada pengolahan gambar tingkat rendah, hal yang utama yang dilakukan bukanlah proses penglihatan, melainkan pemrosesan langkah-langkah untuk melihat. Langkah pertama yang dilakukan adalah memperoleh gambar *digital*. Proses untuk mendapatkan gambar sangat bergantung pada *sensor* untuk mendapatkan gambar. *Sensor* ini dapat berupa sebuah kamera atau *scanner*.

Pada langkah-langkah selanjutnya pada pemrosesan ini dilakukan pekerjaan-pekerjaan yang dasar yang bertujuan untuk mengekstrasi fitur-fitur dari gambar untuk digunakan lagi pada pemrosesan lebih lanjut. pekerjaan-pekerjaan dasar yang dilakukan seperti, *edge detection, filtering, image restoration, image enhancement*, dsb.

Pada proses *edge detection*, tujuan utama yang ingin dicapai adalah untuk mendeteksi keberadaan batasan-batasan atau untuk membuat batasan-batasan yang ada pada gambar terlihat lebih jelas. Batasan-batasan *digital* ini juga dapat dikatakan sebagai batasan diantara dua *regional* yang berbeda yang akan terlihat saat tingkat keterangan pada dua *region* ini sangat berbeda.

Filtering adalah teknik yang digunakan untuk menghilangkan *noise* yang terdapat pada gambar atau untuk memperjelas batasan-batasan atau tingkat kedetailan suatu gambar. teknik yang digunakan adalah *low-frequency* and *high-frequency*, dimana bagian *low-frequency* akan menunjukan objek besar dan latar belakang yang terdapat pada gambar sedangkan *high-frequency* akan menunjukan batasan-batasan serta detail-detail kecil yang terdapat pada gambar.

Image restoration adalah teknik yang digunakan untuk memperbaiki gambar yang ada karena semua gambar yang telah didapat pasti telah mengalami pengurangan kualitas. *Image restoration* dilakukan pada gambar-gambar yang mengalami *blur*, goncangan, dsb.

Image enhancement adalah proses peningkatan kualitas gambar. Proses ini dilakukan terhadap suatu gambar yang tidak mengalami degradasi dari gambar aslinya. Metode peningkatan kualitas gambar terhadap suatu gambar akan berbeda dengan metode yang dilakukan terhadap gambar lainnya

Image registration adalah proses registrasi gambar dilakukan berdasarkan beberapa atau banyak gambar dari objek yang diambil secara terpisah. Selanjutnya, tiap-tiap gambar digabungkan menjadi suatu objek agar dapat dianalisis lebih lanjut. Contoh registrasi gambar adalah proses pengambilan gambar bumi oleh satelit. Gambar bumi yang ingin daimbil tidak dapat diambil langsung secara keseluruhan secara langsung, tetapi harus diambil bagian per bagian. Kemudian bagain-bagian ini disatukan sehingga membentuk gambar bumi secara utuh.

Image analysis adalah operasi yang bertujuan untuk menghitung besaran kuantitatif gambar untuk menghasilkan deskripsinya. Teknik analisis gambar mengekstraksi fitur tertentu yang membantu dalam identifikasi objek. Proses segmentasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya. Analisis gambar biasanya melakukan pendeteksian batas, ekstrasi batas, *fourier transform,* dll.

Image reconstruction adalah operasi yang bertujuan untuk membentuk ulang objek dari beberapa gambar hasil proyeksi. Operasi rekonstruksi gambar banyak digunakan dalam bidang medis. Contohnya adalah foto rontgen dengan sinar X digunkan untuk membentuk ulang gambar organ tubuh

Image compression adalah proses kompresi gambar bertujuan untuk mendapatkan suatu gambar baru yang dapat merepresentasikan suatu gambar dalam bentuk ukuran bit yang lebih sedikt dibandingkan gambar aslinya dan dalam gambar terkompresi ini tidak terjadi penurunan kualitas yang berarti. Dengan kompresi data ini diharapkan dapat mengurangi besarnya ruang penyimpanan data dan lebarnya pita frekuensi yang dibutuhkan dalam sistem transmisi data.

Pada pengolahan gambar tingkat menengah, hal yang dilakukan adalah menyatukan semua token-token yang didapatkan dari pengolahan gambar tingkat rendah menjadi suatu kesatuan dan mengekstrak suatu yang berarti dari gambar atau kesatuan tersebut. Ekstrasi fitur termasuk salah satu teknik yang paling terkenal, termasuk juga didalamnya pemetaanan vektor yang akan diobservasi. Tujuan utama dari fitur ekstraksi adalah untuk mengurangi jumlah data dengan mengukur fitur-fitur yang menjadi penentu pola input.

Pengolahan gambar tingkat atas termasuk didalamnya adalah pengenalan dan penginterpretasian. Pengenalan pola merupakan bagian besar yang paling sering digunakan dalam visi komputer. Di dalamnya termasuk pula pengenalan wajah dan pengenalan ekspresi wajah.

2.1.2. Pengenalan Wajah dan Ekspresi (*Face Recognition and Facial Expression Recognition*)

Pengenalan ekspresi wajah telah menjadi topik penelitian psikologi yang telah dilakukan oleh Charles Darwin sejak abad ke-19 dalam penelitiannya yang berjudul The *Expression of the Emotions in Man and Animals*. Pada tahun 1990, Kenji Mase melakukan penelitian pengenalan wajah dengan teknik *optical flow*.

Teknologi pengenalan wajah dapat diaplikasikan dalam berbagai keperluan seperti mencocokan identitas pada dokumen-dokumen penting, foto, pengenalan orang, pencocokan identitas dan pengawasan keamanan. (Rama, 1994) Pengenalan wajah tersebut dilakukan baik pada gambar statis maupun pada *video*.

Paul Ekman dan Wallace V. Friesen (1987) membentuk suatu sistem pengenalan ekspresi wajah yang dikenal dengan FACS (*Facial Action Coding System*). FACS mengukur tingkah laku- tingkah laku yang muncul dengan mengukur 46 unit aksi yang menggambarkan gerakan dasar pada wajah. Gerakan pada wajah yang merupakan ekspresi dari emosi adalah bentuk komunikasi *non-verbal*. Pengenalan emosi melalui ekspresi wajah tetap bergantung pada kemampuan wajah untuk menunjukkan ekspresi seperti takut, marah, kesal, senang, dsb. dan ditentukan juga oleh tingkat emosi tersebut pada setiap orang. Ekspresi umum yang digunakan dalam penelitian pengenalan ekspresi wajah adalah keenam ekspresi universal atau ekspresi dasar manusia, yaitu terkejut (*surprise*), sedih (*sadness*), marah (*anger*), senang (*happiness*), jijik (*disgust*) dan takut (*fear*).

2.1.3. Pengekstraksian Fitur (*Feature Extraction*)

Dalam melakukan pengenalan pola, dari data yang ada diambil fitur-fitur dari gambar tersebut. Tujuan utama pengekstraksian fitur ini adalah untuk mengambil informasi-informasi paling relevan dan penting dari data gambar dengan dimensi sekecil mungkin. Pola-pola dari gambar dipetakan dalam vektor fitur. Vektor fitur digunakan untuk menentukan pada kelas dari suatu contoh masukan.

Berbagai teknik pengekstraksian fitur yang ada seperti *Principal Component Analysis* (PCA), *Kernel* PCA, *Multilinear* PCA, *Independent Component Analysis* (ICA), *Linear Discriminant Analysis* (LDA), *Non-Negative Matrix Factorization* (NMF).

Teknik Pengekstraksian fitur yang umum digunakan adalah *Principal Component Analysis* (PCA). PCA merupakan suatu teknik transformasi linear. PCA yang diterapkan pada pengenalan wajah melakukan pembelajaran secara holistik atau keseluruhan seperti yang dilakukan oleh Calder, Burton, et al. (2000) dengan membentuk *eigenfaces*. Teknik lainnya yaitu *Independent Component Analysis* (ICA) yang dikembangkan oleh Hyvärinen, *et al*. pada tahun 2001. Metode ICA dikembangkan untuk mencari representasi linear dari data *non-gaussian* sehingga setiap komponennya independen. *Linear Discriminant Analysis* (LDA) juga termasuk teknik transformasi linear. Teknik LDA mencoba untuk membedakan antara kelas-kelas data.

2.1.4. Algoritma Pembelajaran Faktorisasi Matriks Non-Negatif (Non-Negative Matrix Factorization Learning Algorithm)

Faktorisasi matrik telah banyak digunakan dalam berbagai perhitungan aljabar untuk memecahkan permasalahan kompleks. Faktorisasi matriks mendekomposisi suatu matriks menjadi beberapa matriks lain. Pada tahun 1999, Daniel D. Lee dan H. Sebastian Seung mengembangkan algoritma pembelajaran faktorisasi matriks non-negatif / *Non-Negative Matrix Factorization* (NMF). Algoritma pembelajaran ini memfaktorisasi matriks V yang berdimensi n *x* m menjadi 2 matriks lain yang tidak mengandung nilai negatif, yaitu W yang berdimensi n *x* r dan H yang berdimensi r *x* m, dimana nilai r < min(n,m).

$$V \approx W.H \tag{1}$$

Matriks V terdiri atas m data vektor yang berdimensi n. Sifat ketidak-negatif-an digunakan karena dalam neurofisiologi nilai awal neuron tidak pernah bersifat negatif dan tidak akan berubah tanda. Selain itu dalam kasus penelitian ini, yaitu pemrosesan gambar, intensitas dari gambar tidaklah bernilai negatif.

Nilai dari matriks W dipelajari secara *unsupervised* dengan mengaplikasikan NMF kepada data pelatihan V, yang mana data-data pelatihan yang berbeda terletak pada kolom yang berbeda pula. Dapat dikatakan juga bahwa setiap data vektor v disesuaikan dengan kombinasi linear dari kolom pada matriks W dengan data h (kolom pada matriks H).

Algoritma pembelajaran NMF ini melakukan perubahan nilai pada matriks W dan H secara iteratif. Dalam setiap iterasi nilai baru pada matriks W dan H diperoleh dengan mengalikan nilai saat ini dengan suatu faktor yang bergantung pada kualitas perkiraan dari persamaan (1). Untuk mencari nilai faktorisasi V ~ WH yang mendekati, harus ditentukan suatu *cost function* yang menentukan kualitas pendekatan nilai tersebut. *Cost function* dibentuk dengan mengukur perbedaan nilai antara 2 matriks non-negatif A dan B. Pengukuran yang digunakan oleh D.D. Lee dan H. S. Seung adalah mengukur jarak Euclidean (2) dari matriks V dan WH serta mengukur divergensi Kullback-Leibler (3)

$$||A - B||^2 = \sum_{ij} (\mathrm{A_{ij}} - \mathrm{B_{ij}})^2 \qquad (2)$$

Batas bawah dari pengukuran ini bernilai 0 dan terjadi apabila nilai A sama dengan nilai B.

$$D(A||B) = \sum_{ij} (\mathrm{A_{ij}} \log \frac{\mathrm{A_{ij}}}{\mathrm{B_{ij}}} - \mathrm{A_{ij}} + \mathrm{B_{ij}}) \qquad (3)$$

Sama seperti pengukuran jarak Euclidean, persamaan (3) juga memiliki batas bawah nilai 0 dan terjadi apabila nilai A=B. Tetapi yang dimaksud bukanlah jarak, tetapi divergensi antara A dan B

Namun, kedua algoritma tersebut hanya memberikan nilai konvergen pada salah satu matriks saja (matriks W saja atau matriks H saja). Sehingga, D. D. Lee dan H. S. Seung mengusulkan pendekatan multiplikatif yang lebih sederhana namun menghasilkan matriks W dan H yang keduanya konvergen. D. D. Lee dan H. S. Seung mengusulkan 2 teorema berikut:

Teorema 1:

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^T V)_{a\mu}}{(W^T WH)_{a\mu}} \quad (4)$$

$$W_{ia} \leftarrow W_{ia} \frac{(VH^T)_{ia}}{(WHH^T)_{ia}} \quad (5)$$

Teorema 2:

$$H_{a\mu} \leftarrow H_{a\mu} \frac{\sum_i W_{ia} V_{i\mu}/(WH)_{i\mu}}{\sum_k W_{ka}} \quad (6)$$

$$W_{ia} \leftarrow W_{ia} \frac{\sum_\mu H_{a\mu} V_{i\mu}/(WH)_{i\mu}}{\sum_v H_{av}} \quad (7)$$

Sebenarnya, selain algoritma multiplikatif ini, terdapat algoritma lain yang memberikan hasil konvergen pada kedua matriks tersebut yaitu Gradient Descent. Namun pendekatan dengan *Gradient Descent* ini memerlukan waktu yang lama. (*Algorithms for Non-negative Matrix Factorization* - D. D. Lee, H. S. Seung)

Sehingga telah dibuktikan oleh pembuat algoritma ini, D.D. Lee dan H.S.Seung bahwa kualitas perkiraan meningkat (semakin baik) dalam setiap iterasinya. Dalam prakteknya, pengulangan dalam perubahan nilai matriks tersebut dipastikan selalu konvergen. Dan nilai matriks W dan H selama iterasi tetaplah bernilai positif. Sehingga berbeda dari algoritma lain seperti *Principal Component Analysis* (PCA) atau *Linear Discriminant Analysis* (LDA), algoritma NMF hanya melakukan penambahan dan tidak melakukan pengurangan dalam perhitungannya.

Dalam pengembangan NMF ditemukan sebuah masalah yaitu bagaimana cara meminimalkan nilai $D(V\|WH)$ dengan tetap memenuhi syarat W, H ≥ 0. Untuk mendapatkan W dan H yang tidak negatif, diterapkan *multiplicative update rules* pada setiap iterasi.

$$W_{ia} = W_{ia} \frac{\sum_{\mu=1}^{m} \frac{V_{i\mu}}{(WH)_{i\mu}} H_{a\mu}}{\sum_{v} H_{av}} \quad (8)$$

$$H_{a\mu} = H_{a\mu} \frac{\sum_{i=1}^{n} W_{ia} \frac{V_{i\mu}}{(WH)_{i\mu}}}{\sum_{k} W_{ka}} \quad (9)$$

Dengan adanya batasan non-negatif yang diberikan pada setiap faktor, setiap komponen yang berhasil dipelajari oleh NMF dapat dikombinasikan tanpa perlu melakukan pengurangan seperti yang terjadi pada ICA. Namun Li *et al.* (2001) menemukan bahwa bagian-bagian yang dipelajari oleh NMF ternyata tidak terlokalisasi seperti yang ditunjukkan oleh Lee dan Seung (1999).

Penelitian mengenai teknik NMF ini sendiri telah berkembang sedemikian rupa sehingga terdapat banyak pengembangan dari NMF. Patrick O. Hoyer (2004) mengembangkan NMF dengan menambahkan *sparseness constraints* dengan berdasarkan konsep *sparse coding*, yang artinya hanya mengambil nilai yang efektif sebagai representasi data. Algoritma NMF mempelajari representasi data secara *part-based*. NMF dengan *sparseness constraints* membuat pembelajaran representasi *part-based data* yang lebih baik daripada NMF dasar. (*Non-negative Matrix Factorization with Sparseness Constraints* - Patrick O. Hoyer, 2004)

Pada NMF yang ditemukan oleh D. D. Lee dan H. S. Seung, fitur dari gambar dibuat dalam vektor 1 dimensi pada kolom-kolom matriks, sedangkan pada 2D-NMF fitur pada gambar dibentuk menjadi matriks 2 dimensi. Berdasarkan hasil penelitian yang dilakukan oleh Zhang, Chen, dan Zhou (2005) menunjukkan bahwa 2D-NMF menghasilkan akurasi yang lebih baik daripada NMF dengan perbandingan kompresi yang sama serta waktu eksekusi yang lebih singkat. (*Two-Dimensional Non-negative Matrix Factorization for Face Representation and Recognition* - Zhang, Chen, Zhou, 2005)

Cao pada tahun 2007 mengembangkan Online Non-negative Matrix Factorization (ONMF) yang secara otomatis memperbarui faktor-faktor laten dengan mengkombinasikan faktor-faktor lama dengan data yang baru. Pada saat yang bersamaan, ONMF dapat mencari hubungan antara faktor lama dan faktor yang baru, sehingga dapat mengikuti perubahan pola faktor laten secara alami. Metode ini menambahkan batasan yaitu dengan menggunakan matriks ortogonal pada semua faktor yang laten. (*Detect and Track Latent Factors with Online Nonnegative Matrix Factorization* – Cao et al., 2007)

Pada tahun 2010, Liu et al. mengembangkan NMF khususnya dalam mengolah data matriks berukuran besar yang dalam penelitian tersebut diaplikasikan pada model pemograman MapReduce clusters. Liu et al. mempartisi data-data dan menyusun perhitungan agar menghasilkan lokalitas data yang maksimum dan paralel. Partisi dilakukan pada matriks W dan H. Penelitian tersebut menghasilkan NMF yang terdistribusi yang dapat dengan efektif memfaktorisasi matriks

berdimensi puluhan ribu kali puluhan ribu. (*Distributed Nonnegative Matrix Factorization for Web-scale Dyadic Data Analysis on MapReduce* – Liu et al., 2010)

Pada tahun 2008, Vavasis dalam penelitiannya mengukur kompleksitas NMF membuktikan bahwa NMF memiliki kompleksitas tingkat NP-complete dan masih dapat dikembangkan agar memiliki kompleksitas yang polinomial. Sehingga pada tahun 2011 Arora, Ge, Kannan dan Moitra melakukan optimasi pada algoritma NMF tersebut hingga dapat difaktorisasi dalam tingkat kompleksitas polinomial. (*Computing a Non-negative Matrix Factorization – Provably* - Arora, Ge, Kannan dan Moitra - 2012)

2.1.5. Local Non-negative Matrix Factorization

Pengembangan lain dari NMF dilakukan oleh Li pada tahun 2001. Metode baru yang diajukannya adalah *Local Non-Negative Matrix Factorization* yang mempelajari bagian-bagian secara lokal. Metode ini menambahkan proses penguatan sifat lokal pada basis gambar untuk membantu proses pengenalan pola dalam NMF. Penguatan sifat lokal pada LNMF dilakukan dengan memberikan 3 batasan, yaitu:

- Komponen dari suatu vektor basis tidak boleh didekomposisi lebih lanjut menjadi komponen lain dan jumlah dari vektor basis haruslah sekecil mungkin
- Basis yang berbeda harus se-ortogonal mungkin
- Komponen yang memiliki informasi yang penting harus terus dijaga

Ketiga batasan tersebut didefinisikan dalam persamaan berikut

$$D(A\|B) = \sum_{ij}\left(A_{ij}\log\frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij}\right) + \alpha\sum_{ij}u_{ij} - \beta\sum_{ij}v_{ij} \qquad (10)$$

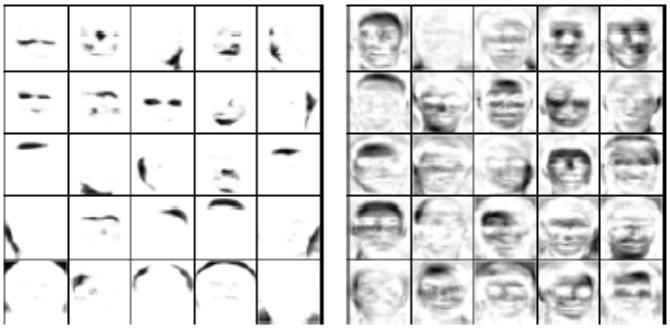
Dimana α, β > 0, $u_{ij} = W^T W$ dan $v_{ij} = HH^T$. Perubahan nilai W dan H yang terjadi pada LNMF didefinisikan sebagai

$$H_{a\mu} = \sqrt{H_{a\mu}\sum_{i=1}^{n} W_{ia}\frac{V_{i\mu}}{(WH)_{i\mu}}} \qquad (11)$$

$$W_{ia} = W_{ia}\sum_{\mu=1}^{m}\frac{V_{i\mu}}{(WH)_{i\mu}}H_{a\mu} \qquad (12)$$

$$W_{ia} = \frac{W_{ia}}{\sum_{j=1}^{n} W_{ja}} \qquad (13)$$

Dengan menggunakan rumus tersebut, akan didapatkan hasil *basis image* yang terlokalisasi dengan baik.



Gambar 2.1 Perbandingan Basis Gambar pada LNMF (kiri) dan NMF (kanan)

Walaupun, metode ini umumnya membutuhkan jumlah iterasi yang lebih banyak dibandingkan dengan NMF (Zhang *et al*. - 2005), LNMF menghasilkan tingkat akurasi yang lebih baik daripada PCA maupun NMF.

2.2. Jaringan Syaraf Tiruan (*Artificial Neural Network*)

Jaringan syaraf tiruan (a*rtificial neural network*) adalah suatu model pemecahan masalah yang mengadaptasi sistem jaringan syaraf biologis, baik secara struktur maupun secara kinerja. Jaringan syaraf tiruan diimplementasikan dalam suatu bentuk model matematika yang mana terdapat beberapa *neuron* yang saling terhubung satu sama lain. Metode jaringan syaraf tiruan umumnya digunakan untuk membangun perangkat yang dapat melakukan proses asosiasi dan pengenalan terhadap pola masukan berdasarkan data training yang telah perangkat tersebut pelajari.

Walaupun secara umum, performa yang dapat dihasilkan oleh jaringan syaraf tiruan ini masih tidak sebaik jaringan syaraf biologis, namun *artificial neural network* dianggap sebagai model yang cukup memadai dalam upaya pengenalan dan pembelajaran pola.

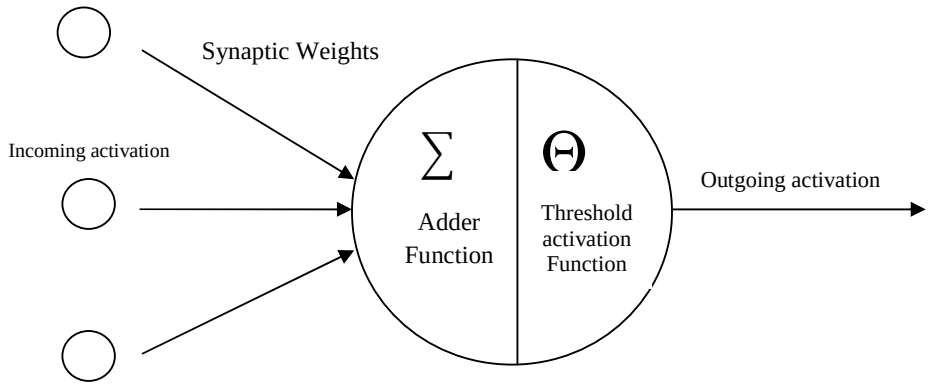
Jaringan syaraf tiruan menjadi populer dan banyak digunakan karena sifat jaringan syaraf tiruan yang fleksibel. Jaringan syaraf tiruan tidak memiliki kelemahan seperti beberapa teknik lain yang mana umumnya proses pelatihan (*training*) dan pengenalan (*recognition*) memerlukan ruang lingkup yang ketat untuk masalah yang spesifik. Artinya, jaringan syaraf tiruan mampu melakukan

proses pengenalan pada masalah yang lebih beragam tanpa perlu memberikan banyak batasan-batasan baru dalam modelnya.

Walaupun begitu, model *artificial neural network* tetap masih memiliki kelemahan. Banyak kritik menyebutkan bahwa model ANN memiliki keterbatasan dalam komputasi (A. K. Dewdney). Selain itu, untuk mendapatkan model ANN yang dapat digunakan dalam robotik, dibutuhkan sejumlah besar data yang beragam.

2.2.1. Model *Neuron*

Neuron adalah dasar pemrosesan dari sebuah jaringan syaraf tiruan. Setiap *neuron* mempunyai satu keluaran yang akan disalurkan ke beberapa *neuron* lainnya. Setiap *neuron* menerima beberapa masukkan dari koneksi yang disebut sinapsis ini. Nilai masukkan adalah nilai dari *neuron* keluaran dikalikan dengan bobot dari sinapsis.

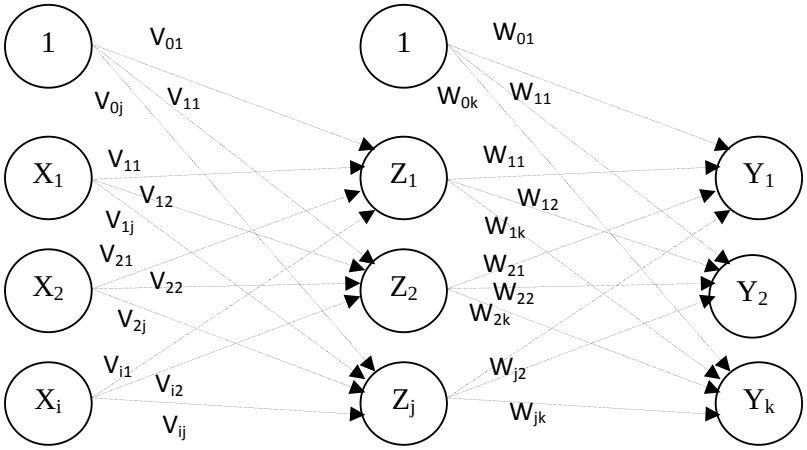


Gambar 2.2 Model Sebuah *Neuron*

2.2.2. Arsitektur *Back-Propagation*

Bentuk paling umum dalam penerapan jaringan syaraf tiruan adalah *Back-Propagation*, dimana arsitektur jaringan dibentuk terlebih dahulu untuk kemudian diuji dengan sejumlah data pelatihan yang sesuai. *Back-Propagation* adalah algoritma pembelajaran *supervised* dan terdiri atas banyak lapisan (*multi-layer network*) yang terdiri atas lapisan masukan (*input*), lapisan tersembunyi (*output*) dan lapisan tersembunyi (*hidden*).

Back-Propagation akan mencoba menelusuri hasil berdasarkan data pelatihan masukkan dan kemudian membandingkannya dengan nilai keluaran yang diharapkan (*target*), apabila nilai error yang dicapai masih lebih besar daripada target error yang diharapkan, maka *Back-Propagation* akan memperbaharui nilai bobot jaringan hingga kesalahan (*error*) maksimum yang diminta dapat dicapai. Pada prakteknya, *Back-Propagation* menggunakan *gradient descent* untuk mendapatkan nilai koreksi bobot agar nilai yang didapat sesuai dengan data masukkan dan data keluaran dalam pelatihan.



Gambar 2.3 Arsitektur Jaringan *Back-Propagation*

Keterangan :

X = Lapisan Masukan (*input layer*)

Z = Lapisan Tersembunti (*hidden layer*)

Y = Lapisan keluaran (*output layer*)

i = Jumlah unit pengolah pada lapisan masukkan

j = Jumlah unit pengolah pada lapisan tersembunyi

k = Jumlah unit pengolah pada lapisan keluaran

V_{ij} = Bobot pada lapisan tersembunyi

W_{jk} = Bobot pada lapisan keluaran

V_{0j} = Bias pada lapisan tersembunyi

W_{0j} = Bias pada lapisan tersembunyi dan lapisan keluaran.

Back-Propagation memiliki lapisan tersembunyi (*hidden layer*) dimana diantara lapisan masukkan (*input layer*) dan lapisan keluaran (ouput layer) disisipkan satu atau beberapa lapisan tersembunyi dengan jumlah *neuron* tertentu. Back-Propagation memiliki 3 fase yaitu fase *feed forward*, fase *feed backward dan* fase *modification.*

2.2.3. Algoritma *Back-Propagation*

Ketiga fase utama dibagi dalam 7 langkah yang dilakukan dalam algoritma *Back-Propagation*. Sebelum fase pertama dimulai, dilakukan inisialisasi bobot. Isi bobot awal dengan nilai acak. Tentukan besarnya skala β (*learning rate*), yaitu $\beta = 0.7(p)^{1/n}$, dengan p adalah jumlah *neuron* pada lapisan tersembunyi dan n adalah jumlah *neuron* pada lapisan input. Inisialisasi bobot V_{ij} secara acak dengan nilai inisialisasi V_{ij} adalah $-0.5 \leq V_{ij} \leq 0.5$.

Berikut adalah penjelasan lengkap tiap-tiap fase dan langkah-langkah dalam setiap fase tersebut:

Fase I : Propagasi Maju (*Feed Forward*)

Langkah 1: Tiap unit masukan menerima sinyal dan meneruskannya ke unit tersembunyi di atasnya.

Langkah 2: Hitung semua keluaran di unit tersembunyi Z_j (j = 1, 2, ..., p)

$$Z\_in_j = V_{0j} + \sum X_i \,.\, V_{ij} \quad (14)$$

$$Z_j = f\left(Z\_in_j\right) = \frac{1}{(1+ e^{-Z\_in_j})} \quad (15)$$

Langkah 3: Hitung semua keluaran jaringan di unit Y_k (k = 1, 2, …, m)

$$Y\_in_k = W_{0k} + \sum Z_j \,.\, W_{jk} \quad (16)$$

$$Y_k = f(Y\_in_k) = \frac{1}{(1+ e^{-Y\_in_k})} \quad (17)$$

Fase II : Propagasi Mundur (*Feed Backward*)

Langkah 4: Hitung faktor δ unit keluaran berdasarkan nilai *error* di setiap unit keluaran Y_k = (k = 1,2,…,m)

$$\delta_k = (T_k - Y_k).f'\left(Y_{in_k}\right) = (T_k - Y_k).Y_k(1 - Y_k) \quad (18)$$

δ_k merupakan nilai *error* yang akan dipakai dalam perubahan bobot lapisan dibawahnya (langkah 7). Hitung suku perubahan bobot W_{jk} (yang akan dipakai nanti untuk merubah bobot W_{kj}) dengan laju percepatan α. (k= 1,2, ..., m ; j = 0,1,2, ..., p)

$$\Delta W_{jk} = \alpha.\delta_k.Z_j \quad (19)$$

Langkah 5: Hitung faktor δ unit tersembunyi berdasarkan *error* di setiap unit tersembunyi Z_j = (j=1,2,…,p)

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k . W_{kj} \quad (20)$$

Faktor δ unit tersembunyi:

$$\delta_j = \delta\_\text{in}_j . f'(Z\_in_j) = \delta_{\text{in}_j} . Z_j . (1 - Z_j) \quad (21)$$

Hitung suku perubahan bobot v_{ij} (yang akan dipakai untuk merubah V_{ij}) (j = 1, 2, ..., p; i = 0,1,2, ..., n)

$$\Delta V_{ij} = \alpha . \delta_j . X_i \quad (22)$$

Fase III : Perubahan Bobot

Langkah 6: Hitung semua perubahan bobot. Perubahan bobot garis yang menuju ke unit keluaran: (k=1,2, ...m ; j =0,1,2,...,p)

$$W_{jk}(baru) = W_{jk}(lama) + \Delta W_{jk} \quad (23)$$

Perubahan bobot garis yang menuju ke unit tersembunyi (j=1,2, ...p ; i =0,1,2,...,n)

$$V_{ij}(baru) = V_{ij}(lama) + \Delta V_{ij} \quad (24)$$

Langkah 7: Berhenti

Terdapat dua kondisi perhentian pada algoritma *Back-Propagation*, yaitu :

1. *Error* < *Error* maksimum

   Error adalah perbedaan yang terjadi antara ouput terhadap target yang diinginkan. Proses ANN akan berhenti jika bessarnya error yang terjadi telah bernilai lebih kecil dari nilai error maksimum yang telah ditetapkan. Besarnya nilai error dihitung dengan menggunakan fungsi error kuadratis.

$$E = 0.5 \,.\sum_k (t_k - Y_k)^2 \qquad (25)$$

2. *Epoch* > *Epoch* maksimum

   Epoch adalah suatu langkah yang dilakukan dalam pembelajaran pada ANN. Jika besarnya *epoch* lebih besar dari besarnya *epoch* maksimum yang telah ditetapkan, maka proses pembelajaran akan berhenti.

   Kedua kondisi penghentian di atas digunakan dengan logika **OR**. Jadi kondisi penghentian terjadi jika besarnya *Error* < *Error* maksimum atau *Epoch* > *Epoch* maksimum.

2.2.4. Aplikasi Back-Propagation

Banyak penelitian pengenalan pola menggunakan jaringan syaraf tiruan *Back-Propagation*. Pada tahun 2000, Feitosa, Vellasco, Maffra, Andrade & Oliveira melakukan penelitian pengenalan ekspresi wajah dengan membandingkan *Radial Basis Function* (RBF) dan Back-Propagation dengan menggabungkannya dengan metode ekstraksi fitur PCA. Hasil penelitian menunjukkan tingkat akurasi yang lebih tinggi oleh RBF, tetapi kestabilan atau presisi pengenalan Back-Propagation lebih baik dibanding RBF. (*Facial Expression Classification Using RBF and Back Propagation Neural Networks.* - Feitosa, Vellasco, Maffra, Andrade & Oliveira - 2000)

Pada tahun 2004, Reda, Wilson dan Silohey melakukan penelitian pengenalan ekspresi wajah dengan menggunakan jaringan syaraf tiruan. Dalam penelitian tersebut Back-Propagation dapat mengenali wajah hingga akurasi sebesar 100%. Dilakukan pula pengujian dengan menggunakan gambar yang memiliki *noise* dan menghasilkan akurasi sekitar 82% hingga 100% dengan jenis *noise* yang berbeda-beda. (*Artificial Neural Network-Based Face Recognition* – Reda, Wilson & Silohey 2004)

Pada tahun 2011, Ganatra, Kosta, Panchal dan Gajjar. Penelitian tersebut membuktikan bahwa jaringan syaraf tiruan *Back-Propagation* lebih baik daripada *multi-layer network* lain. Dan dibandingkan dengan Algoritma Genetika, *K-nearest Neighbors*, *Back-Propagation* memberikan hasil yang lebih baik dengan membandingkan beberapa parameter yang digunakan. (*Initial Classification Through Back Propagation In a Neural Network Following Optimization Through GA to Evaluate the Fitness of an Algorithm* – Ganatra, Kosta, Panchal, Gajjar - 2011)