

**SISTEM PENDETEKSIAN DIABETES MELITUS MENGGUNAKAN
JARINGAN SYARAF TIRUAN LVQ (*LEARNING VECTOR
QUANTIZATION*)**



PROPOSAL TUGAS AKHIR

**Disusun Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Komputer
pada Departemen Ilmu Komputer/ Informatika**

Disusun Oleh :

REYHAN SYARIF

24010313130104

**DEPARTEMEN ILMU KOMPUTER/ INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO
2017**

HALAMAN PENGESAHAN

Yang bertandatangan di bawah ini menyatakan bahwa Proposal Tugas Akhir yang berjudul :

**SISTEM PENDETEKSIAN DIABETES MELITUS MENGGUNAKAN
JARINGAN SYARAF TIRUAN LVQ (*LEARNING VECTOR
QUANTIZATION*)**

Dipersiapkan dan disusun oleh :

Nama : Reyhan Syarif

NIM : 24010313130104

Telah disahkan sebagai Proposal Tugas Akhir yang merupakan salah satu syarat untuk memperoleh gelar Sarjana Komputer.

Semarang, 15 September 2017

Mengetahui,
Ketua Departemen Ilmu Komputer/
Informatika

Menyetujui,
Pembimbing

Dr. Retno Kusumaningrum, S.Si, M.Kom
NIP. 198104202005012001

Priyo Sidik Sasongko, S.Si, M.Kom
NIP.197007051997021001

DAFTAR ISI

HALAMAN PENGESAHAN.....	ii
DAFTAR ISI.....	iii
DAFTAR GAMBAR	v
DAFTAR TABEL	vi
DAFTAR LAMPIRAN	vii
BAB I PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah.....	3
1.3. Tujuan dan Manfaat.....	3
1.4. Ruang Lingkup	3
BAB II TINJAUAN PUSTAKA.....	5
2.1. Diabetes Melitus	5
2.2. Jaringan Syaraf Tiruan.....	6
2.2.1. Arsitektur Jaringan Syaraf Tiruan	6
2.2.2. Pelatihan Jaringan Syaraf Tiruan	7
2.3. Algoritma Learning Vector Quantization.....	8
2.3.1. Arsitektur Jaringan Learning Vector Quantization	8
2.3.2. Pelatihan Standar <i>Learning Vector Quantization</i>	9
2.4. Evaluasi Kinerja <i>Classifier</i>	14
2.4.1. <i>K-Fold Cross Validation</i>	15
2.4.2. <i>Confussion Matrix</i>	15
2.5. Model <i>Waterfall</i>	15
2.6. Pemodelan Analisis	18
2.6.1. Pemodelan Data.....	18
2.6.2. Pemodelan Fungsional	19
2.7. Perancangan Struktur Data	21
2.8. Bahasa Pemrograman PHP	22
2.9. DBMS MySQL.....	22
2.10. Pengujian Perangkat Lunak	23
BAB III METODOLOGI PENELITIAN.....	26

3.1. Metodologi Penelitian.....	26
3.1.1. Studi Pustaka.....	26
3.1.2. Pengumpulan Data	26
3.2. Arsitektur Sistem	26
3.3. Garis Besar Penyelesaian Masalah	28
3.4. Jadwal	31
DAFTAR PUSTAKA	32
LAMPIRAN	34

DAFTAR GAMBAR

Gambar 2. 1. Struktur <i>Neuron</i> pada JST	6
Gambar 2. 2. Arsitektur Jaringan <i>Learning Vector Quantization (LVQ)</i>	9
Gambar 2. 5. Arsitektur Jaringan <i>Learning Vector Quantization (LVQ)</i>	11
Gambar 2. 6. Model <i>Waterfall</i>	16
Gambar 2. 7. Contoh <i>Conceptual Data Model (CDM)</i>	21
Gambar 2. 8. Contoh <i>Physical Data Model (PDM)</i>	22
Gambar 3. 1. Arsitektur Sistem Pendeteksian Diabetes Melitus Menggunakan Jaringan Syaraf Tiruan LVQ	27
Gambar 3. 2. Tahapan Alur Pengolahan Data Deteksi Diabetes Melitus	27
Gambar 3. 3. <i>Flow</i> Garis Besar Penyelesaian Masalah.....	28

DAFTAR TABEL

Tabel 2. 1. Data Pelatihan Kasus <i>Learning Vector Quantization</i>	10
Tabel 2. 2. Bobot Awal Data Pelatihan	11
Tabel 2. 3. Data Pelatihan	11
Tabel 2. 4. Tabel SRS.....	17
Tabel 2. 5. Tabel Notasi Pemodelan Data	19
Tabel 2. 6. Tabel Notasi Pemodelan Fungsional.....	20
Tabel 2. 7. Format Pendefinisian Rencana Pengujian.....	24
Tabel 2. 8. Format Pendefinisian Hasil Pengujian	24
Tabel 3. 1. Tabel Contoh Data Diabetes Melitus	29
Tabel 3. 2. Tabel <i>Confussion Matrix</i> Dengan Dua Kelas.....	30
Tabel 3. 3. Jadwal Rencana Tugas Akhir	31

DAFTAR LAMPIRAN

Lampiran 1. Kartu Bimbingan Tugas Akhir	35
Lampiran 2. Kartu Keikutsertaan Seminar TA1	37
Lampiran 3. Daftar Hadir Seminar Proposal Tugas Akhir.....	38
Lampiran 4. Notulensi Seminar Proposal Tugas Akhir	39

BAB I

PENDAHULUAN

Bab ini berisi latar belakang, rumusan masalah, tujuan dan manfaat, dan ruang lingkup tugas akhir mengenai prediksi diabetes melitus dengan menggunakan algoritma *Learning Vector Quantization* (LVQ).

1.1. Latar Belakang

Degenerasi merupakan penurunan fungsi jaringan sebagai akibat dari perubahan-perubahan jaringan itu sendiri (degenerasi murni), ataupun akibat dari endapan-endapan bahan lain pada jaringan tersebut (infiltrasi). Salah satu penyakit degeneratif dengan proporsi tertinggi di Indonesia dan merupakan penyebab kematian tertinggi keenam di Indonesia adalah diabetes melitus (DM). DM merupakan kumpulan gejala yang timbul pada seseorang akibat kadar gula yang tinggi (Waspadji, 2007).

Jumlah kasus diabetes melitus yang semakin naik erat kaitannya dengan transisi demografi. Transisi demografi yang disebabkan oleh peningkatan kualitas hidup berhubungan dengan peningkatan kasus diabetes melitus. Hal ini dikarenakan perubahan struktur pekerjaan penduduk, yang sebelumnya didominasi dari sektor pertanian menjadi sektor pabrik dan jasa menyebabkan berkurangnya aktivitas fisik sehingga energi yang dikonsumsi lebih besar dari energi yang dikeluarkan. Hal tersebut ditengarai menyebabkan obesitas yang merupakan salah satu faktor resiko diabetes melitus (Ramachandran, 2004).

Diabetes Melitus dapat menimbulkan berbagai dampak negatif. Diabetes merupakan faktor resiko dari penyakit kardiovaskuler (Hill, 2011). Penderita diabetes beresiko mengalami *coronary artery disease* sebanyak 3,2 kali lebih besar dibandingkan non-penderita, resiko mengalami stroke sebanyak 2,9 kali lebih besar, dan resiko mengalami penyakit terkait jantung sebanyak 1,9 kali lebih besar dari non-penderita. Resiko ini meningkat pada penderita diabetes melitus yang berusia 35-64 tahun (Ariza, 2010). Selain penyakit kardiovaskuler, DM juga merupakan salah satu penyebab utama penyakit ginjal dan kebutaan pada usia di bawah 65 tahun, dan juga amputasi (Hill, 2011). Selain itu, diabetes juga menjadi penyebab terjadinya amputasi (bukan disebabkan oleh trauma), disabilitas, hingga kematian (Praet, 2010).

Diabetes melitus mempunyai beberapa indikator sebagai penilaian untuk menentukan apakah seseorang terkena diabetes melitus atau tidak, yaitu umur, jenis kelamin, kadar glukosa darah, kadar glukosa urin, kadar aseton urin, kadar kolesterol, dan kadar trigliserida. Indikator utamanya adalah kadar glukosa darah yaitu seseorang dapat dikatakan terkena diabetes melitus

jika kadar glukosa darah ≥ 7 mmol/L pada saat puasa atau $\geq 11,1$ mmol/L saat 2 jam setelah puasa.

Saat ini penanganan atau diagnosis penyakit diabetes melitus hanya dapat dilakukan secara manual langsung oleh dokter atau tenaga medis, yang pada kenyataannya pasien harus mengeluarkan biaya dan usaha untuk datang ke pusat pelayanan kesehatan seperti puskesmas atau rumah sakit dengan prosedur pemeriksaan yang rumit sehingga banyak orang yang malas untuk memeriksakan dirinya sebagai tindakan pencegahan dini dari penyakit diabetes melitus. Selain itu, karena jumlah tenaga medis yang tersedia juga terbatas, menyebabkan proses pemeriksaan akan memakan waktu yang cukup lama. Padahal pada zaman yang serba modern ini masyarakat menuntut untuk memiliki kehidupan yang serba praktis.

Teknologi semakin berkembang pesat di era ini, memudahkan manusia dalam mengambil keputusan ataupun memprediksi suatu hal. Komputer dapat bekerja meniru cara kerja otak manusia sehingga dapat menimbang dan mengambil keputusan dengan menggunakan metode jaringan syaraf tiruan. Jaringan syaraf tiruan adalah sistem komputasi dimana arsitektur dan operasi diilhami dari pengetahuan tentang sel syaraf biologis dalam otak, salah satu representasi buatan dari otak manusia yang selalu mencoba menstimulasi proses pembelajaran pada otak manusia tersebut. Jaringan syaraf tiruan memiliki kemampuan untuk belajar, selain itu juga mampu menghasilkan aturan atau operasi dari beberapa contoh atau *input* yang dimasukkan, dan membuat prediksi tentang kemungkinan output yang akan muncul (Hermawan, 2006).

Salah satu metode dalam jaringan syaraf tiruan adalah metode LVQ. Jenis ini telah banyak diterapkan dalam sistem pembelajaran untuk prediksi. Hal ini disebabkan karena konsepnya yang sederhana dan mudah dipahami, komputasinya yang efisien, dapat meringkas data set yang besar menjadi vektor *codebook* berukuran kecil, dimensi *codebook* tidak dibatasi seperti dalam teknik *nearest neighbour*, tingkat *error* yang rendah dan memiliki kemampuan prediksi dengan tingkat keakuratan yang lebih baik dari metode-metode lainnya. Sudah banyak penelitian yang menggunakan LVQ terhadap beberapa kasus mengenai prediksi antara lain penerapan LVQ penentuan bidang konsentrasi tugas akhir (studi kasus mahasiswa teknik informatika UIN Suska Riau) (Budianita, E., Arni, U. D., 2015) dengan tingkat akurasi sebesar 80 %, sedangkan untuk dibidang medis diantaranya memprediksi penyakit jantung koroner dengan menggunakan algoritma LVQ (Fagustina, A., dkk, 2010) dengan tingkat keakuratan sebesar 80% dan pengenalan jenis penyakit THT menggunakan jaringan LVQ (Sela, I. E., Hartati, S., 2010) dengan tingkat akurasi sebesar 94%. Sehingga metode LVQ ini cocok untuk membantu mendeteksi penyakit diabetes melitus dengan mudah.

Dari permasalahan tersebut, maka tugas akhir yang akan dilakukan adalah menerapkan metode LVQ dalam prediksi penyakit diabetes melitus.

1.2. Rumusan Masalah

Berdasarkan permasalahan yang telah disampaikan dalam latar belakang sebelumnya, perumusan masalah tugas akhir ini bagaimana membangun sistem yang dapat digunakan untuk prediksi diabetes melitus dengan menggunakan algoritma LVQ.

1.3. Tujuan dan Manfaat

Tujuan dari tugas akhir ini adalah untuk menghasilkan sistem yang dapat digunakan untuk prediksi penyakit diabetes melitus dengan menggunakan algoritma LVQ. Manfaat yang diharapkan dari penelitian ini adalah:

1. Bagi Mahasiswa
 - a. Memenuhi persyaratan menyelesaikan pendidikan di Universitas Diponegoro Semarang
 - b. Mengimplementasikan ilmu yang diperoleh selama perkuliahan dalam merancang implementasi algoritma LVQ pada sistem prediksi penyakit diabetes melitus.
2. Bagi Instansi
 - a. Hasil penelitian dapat digunakan untuk mempermudah serta mempercepat proses dalam mendiagnosa penyakit diabetes melitus berdasarkan kriteria yang telah ditentukan.
3. Bagi Masyarakat
 - a. Hasil penelitian dapat digunakan dalam mempermudah serta mempercepat proses pemeriksaan, sehingga masyarakat mau untuk melakukan pemeriksaan dini.

1.4. Ruang Lingkup

Dalam penyusunan penelitian ini diberikan ruang lingkup yang jelas agar pembahasan lebih terarah dan tidak menyimpang dari tujuan penulisan. Ruang lingkup pada tugas akhir ini adalah sebagai berikut:

1. Data untuk studi kasus diperoleh dari RS Pusat Pertamina, Jakarta.
2. Pengembangan perangkat lunak menggunakan model *waterfall*.
3. Perangkat lunak diimplementasikan berbasis web dengan menggunakan bahasa pemrograman PHP dan DBMS MySQL.

4. Pengujian menggunakan metode *Black Box*.
5. Menggunakan metode LVQ untuk prediksi penyakit diabetes melitus.
6. Keluaran yang dihasilkan berupa hasil prediksi penyakit diabetes melitus berdasarkan kriteria yang telah diberikan.

BAB II

TINJAUAN PUSTAKA

Bab ini menjelaskan teori-teori yang digunakan untuk merancang sistem yang akan dibangun meliputi Diabetes Melitus, jaringan syaraf tiruan, algoritma *Learning Vector Quantization (LVQ)*, evaluasi kinerja *classifier*, model pengembangan waterfall, pemodelan analisis, perancangan struktur data, bahasa pemrograman PHP, DBMS MySQL dan pengujian perangkat lunak.

2.1. Diabetes Melitus

Diabetes melitus merupakan penyakit kronis dengan metabolisme yang tidak teratur (Hospital Authority, 2016).

Ketika kita mengonsumsi karbohidrat (termasuk gula dan pati, dll), bahan-bahan tersebut akan dihancurkan dan menjadi glukosa lalu diserap oleh usus kecil ke dalam sistem peredaran darah. Kemudian pankreas mengeluarkan insulin, yang membantu glukosa masuk ke dalam sel untuk digunakan oleh tubuh. Kadar glukosa akan meningkat bila sekresi insulin tidak mencukupi atau tubuh tidak bias menggunakan insulin yang dihasilkan. Keadaan ini mengakibatkan gangguan metabolisme lemak dan protein, serta penghancuran berbagai macam sistem tubuh dan organ dalam jangka waktu lama. (Hospital Authority, 2016).

Penderita diabetes melitus memiliki gejala-gejala awal. Gejala awalnya adalah sebagai berikut (Hospital Authority, 2016).

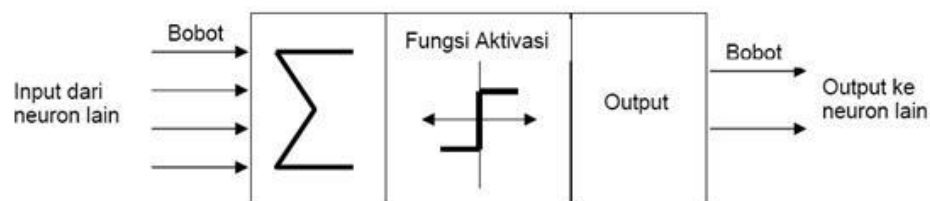
1. Sering merasa haus
2. Sering buang air kecil
3. Sering merasa lapar
4. Penurunan berat badan
5. Kelelahan
6. Penglihatan yang kabur
7. Tingkat penyembuhan luka yang lambat

Berkembangnya penyakit diabetes melitus di Indonesia berkaitan dengan gaya hidup yang kurang sehat, kondisi sosial ekonomi yang buruk, serta kurang memadainya fasilitas kesehatan masyarakat yang ada sehingga memudahkan berkembangnya penyakit DM.

2.2. Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan (JST) adalah sistem terkomputasi dimana arsitektur dan operasinya diilhami dari pengetahuan tentang sel saraf biologis di dalam otak. JST merupakan salah satu pembelajaran pada otak manusia yang selalu mencoba menstimulasi proses pembelajaran pada otak manusia tersebut. Model jaringan syaraf ditunjukkan dengan kemampuannya dalam emulasi, analisis, prediksi dan asosiasi. Kemampuan yang dimiliki jaringan syaraf tiruan dapat digunakan untuk belajar dan menghasilkan aturan atau operasi dari beberapa contoh atau *input* yang dimasukkan dan membuat prediksi tentang kemungkinan *output* yang akan muncul atau menyimpan karakteristik input yang diberikan kepada jaringan syaraf tiruan. [Hermawan, 2006]

Setiap pengolahan elemen membuat perhitungan berdasarkan pada jumlah masukan (*input*). Sebuah kelompok pengolahan elemen disebut layer atau lapisan dalam jaringan. Lapisan pertama adalah *input* dan yang terakhir adalah *output*. Lapisan di antara lapisan *input* dan *output* disebut dengan *hidden layer*. JST merupakan suatu bentuk arsitektur yang terdistribusi paralel dengan sejumlah besar titik dan hubungan antar titik tersebut. Tiap titik yang berhubungan mempunyai harga yang diasosiasikan sebagai bobot. Setiap titik mempunyai nilai yang diasosiasikan sebagai nilai aktivasi titik. Struktur neuron pada jaringan syaraf tiruan diilustrasikan pada Gambar 2.1.



Gambar 2. 1. Struktur *Neuron* pada JST

2.2.1. Arsitektur Jaringan Syaraf Tiruan

Ada beberapa arsitektur pada jaringan syaraf tiruan yang sering digunakan dalam berbagai aplikasi (Puspitaningrum, D., 2006), antara lain:

1. Jaringan Layar Tunggal

Merupakan jaringan dengan lapisan tunggal yang terdiri dari 1 *layer input* dan 1 *layer output*. Setiap *neuron/unit* yang terdapat di dalam lapisan / *layer input* selalu terhubung dengan setiap *neural* yang terdapat pada *layer output*. Jaringan

ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi. Contoh algoritma jaringan syaraf tiruan yang menggunakan metode ini yaitu : ADALINE, Hopfield, Perceptron.

2. Jaringan Layar Jamak

Merupakan jaringan yang memiliki ciri khas tertentu yaitu memiliki 3 jenis *layer* yakni *layer input*, *layer output* dan *layer* tersembunyi (*hidden layer*). Jaringan dengan banyak lapisan ini dapat menyelesaikan permasalahan yang kompleks dibandingkan jaringan dengan lapisan tunggal. Namun proses pelatihan sering membutuhkan waktu yang cenderung lama. Contoh algoritma jaringan syaraf tiruan yang menggunakan metode ini yaitu : MADALINE, backpropagation, neocognitron.

3. Jaringan dengan lapisan kompetitif

Pada jaringan ini sekumpulan neuron bersaing untuk mendapatkan hak menjadi aktif. Contoh algoritma yang menggunakan metode ini adalah : LVQ.

2.2.2. Pelatihan Jaringan Syaraf Tiruan

Berdasarkan cara pembelajaran atau pelatihannya, jaringan syaraf tiruan dikelompokkan menjadi 3 (Puspitaningrum, D., 2006), yaitu:

1. *Supervised Learning* (Pembelajaran Terawasi)

Pada jenis pembelajaran ini, setiap pola yang diberikan kedalam jaringan syaraf tiruan telah diketahui *outputnya*. Selisih antara pola *output* aktual (*output* yang dihasilkan) dengan pola *output* yang dikehendaki (*output target*) yang disebut *error* digunakan untuk mengoreksi bobot jaringan syaraf tiruan sehingga jaringan syaraf tiruan mampu menghasilkan output sedekat mungkin dengan pola target yang telah diketahui oleh jaringan syaraf tiruan. Contoh algoritma jaringan syaraf tiruan yang menggunakan metode ini adalah : Hebbian, Perceptron, ADALINE, Boltzman, Hopfield, *Backpropagation*.

2. *Unsupervised Learning* (Pembelajaran Tidak-Terawasi)

Pada jenis pembelajaran ini tidak memerlukan target *output*. Pada metode ini tidak dapat ditentukan hasil seperti apakah yang diharapkan selama proses pembelajaran. Selama proses pembelajaran, nilai bobot disusun dalam suatu range tertentu tergantung pada nilai input yang diberikan. Tujuan pembelajaran ini adalah mengelompokkan unit-unit yang hampir sama dalam suatu area

tertentu. Pembelajaran ini biasanya sangat cocok untuk klasifikasi pola. Contoh algoritma jaringan syaraf tiruan yang menggunakan metode ini adalah : Competitive, Hebbian, Kohonen, LVQ, Neocognitron.

3. *Hybrid Learning* (Pembelajaran Hibrida)

Merupakan kombinasi dari metode pembelajaran *supervised learning* dan *unsupervised learning*, sebagian dari bobot-bobotnya ditentukan melalui pembelajaran terawasi dan sebagian lainnya melalui pembelajaran tak terawasi. Contoh algoritma jaringan syaraf tiruan yang menggunakan metode ini adalah: algoritma RBF. Metode algoritma yang baik dan sesuai dalam melakukan pengenalan pola-pola gambar adalah algoritma *Backpropagation* dan Perceptron. Untuk mengenali teks berdasarkan tipe font akan digunakan algoritma *Backpropagation*.

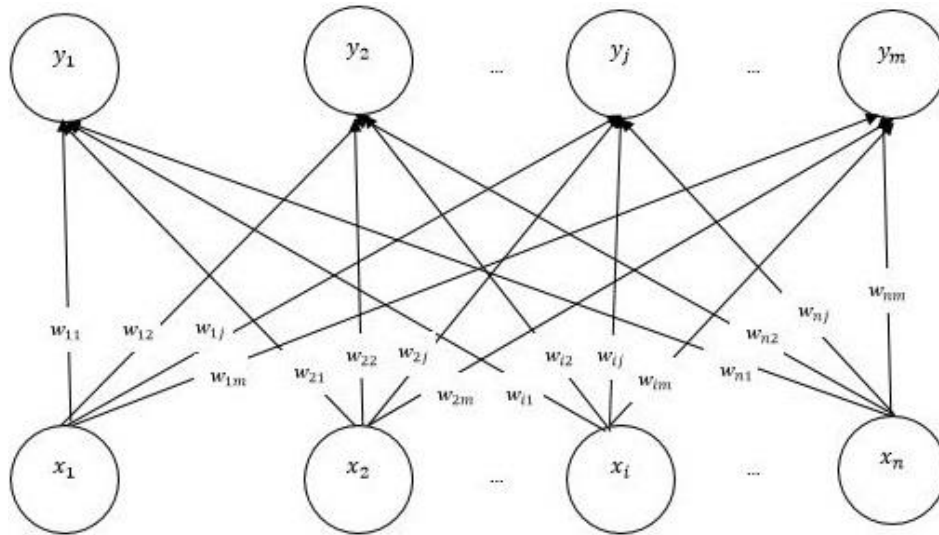
2.3. Algoritma Learning Vector Quantization

Learning Vector Quantization merupakan suatu metode jaringan syaraf tiruan untuk melakukan pembelajaran pada lapisan kompetitif. Lapisan kompetitif akan secara otomatis mempelajarinya untuk mengklasifikasikan vektor-vektor input. Kelas-kelas yang diperoleh sebagai hasil dari lapisan kompetitif ini hanya bergantung pada jarak antara vektor-vektor input. Jika dua vektor input mendekati sama, maka lapisan kompetitif akan meletakkan vektor input tersebut ke dalam kelas yang sama (Ranadhi, Indarto, dan Hidayat, 2006).

2.3.1. Arsitektur Jaringan Learning Vector Quantization

Arsitektur jaringan LVQ hanya terdapat dua *layer* saja yaitu *layer input* dan *layer output*. Setiap unit dari *layer input* pada jaringan LVQ selalu terhubung dengan setiap unit yang berada pada *layer output*. Jaringan LVQ terdiri dari banyak lapisan/*layer* (multilayer network), yaitu *layer input* yang terdiri dari 1 hingga n unit *input* dan *layer output* yang terdiri dari 1 hingga m unit *output*.

Arsitektur jaringan LVQ secara umum dapat dilihat pada Gambar 2.2.



Gambar 2. 2. Arsitektur Jaringan *Learning Vector Quantization* (LVQ)

Penjelasan :

x : vektor input, $x = [x_1, x_2, \dots, x_i, \dots, x_n]$

y : vektor output, $y = [y_1, y_2, \dots, y_j, \dots, y_m]$

$$w : \text{bobot}, w = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & \dots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \dots & w_{nm} \end{bmatrix}$$

2.3.2. Pelatihan Standar *Learning Vector Quantization*

Secara rinci algoritma pelatihan jaringan LVQ dapat diuraikan sebagai berikut (T.Sutojo, 2011) :

1. Menetapkan bobot awal (w) berukuran $t \times n$, maksimum epoch (MaxEpoch), error minimum (ϵ) dan *learning rate* (α) yang bernilai $0 < \alpha < 1$.
2. Memasukkan input x dan target (T)
3. Menetapkan kondisi awal epoch = 0
4. Dikerjakan jika ($epoch \leq \text{MaxEpoch}$) atau ($\alpha < \epsilon$)
 - a. $Epoch = epoch + 1$
 - b. Untuk $i = 1$ sampai n
 - Untuk $k = 1$ sampai t

$$\text{Menghitung nilai } S_{ik} \sqrt{\sum_{j=1}^m (x_{ij} - w_{kj})^2}$$

- i. Menentukan nilai S_{ik} terkecil dan simpan k

ii. Membandingkan nilai k dengan T_{1k}

a. Jika $k = T_{1k}$ maka,

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) + \alpha (x_i - w_{kj}(\text{lama}))$$

b. Jika $k \neq T_{1k}$ maka,

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) - \alpha (x_i - w_{kj}(\text{lama}))$$

5. Mengurangi nilai α

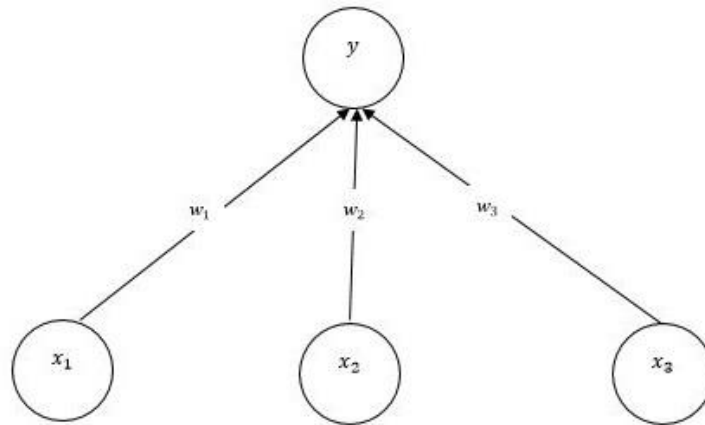
Penjelasan :

- i : indeks untuk jumlah data pelatihan
- j : indeks untuk variabel penentu kelas output
- k : indeks untuk kelas output
- m : jumlah variabel penentu kelas output
- n : jumlah data pelatihan
- t : jumlah kelas output
- x : vektor pelatihan dalam hal ini berukuran $m \times n$
- T : Target pelatihan dalam hal ini berukuran $m \times 1$
- w_k : Vektor bobot kelas ke- k
- S_{ik} : Nilai Euclidean Distance vektor pelatihan ke- i dan kelas ke- k

Perhitungan kasus metode *Learning Vector Quantization* dapat dilihat pada contoh berikut. Diketahui 6 buah data yang terbagi menjadi dua buah kelompok, yaitu kelompok data yang menjadi bobot awal data pelatihan serta kelompok data pelatihan pada Tabel 2.1 dengan arsitektur jaringannya dapat dilihat pada Gambar 2.5.

Tabel 2. 1. Data Pelatihan Kasus *Learning Vector Quantization*

No	Input Vektor	Kelas (T)
1	001	1
2	111	2
3	011	1
4	010	2
5	110	2
6	101	1



Gambar 2. 3. Arsitektur Jaringan *Learning Vector Quantization (LVQ)*

1. Menetapkan *learning rate* (α) = 0,05, maksimum *epoch* = 2, penurunan α = 0,1. α (lama) dan dua input pertama dijadikan bobot awal (w) sebagai penentu kelas output (k), dapat dilihat pada tabel 2.2.

Tabel 2. 2. Bobot Awal Data Pelatihan

No	Input Vektor	Target (T)
1	001	1
2	111	2

2. Memasukkan input x dan target (T), dapat dilihat pada tabel 2.3.

Tabel 2. 3. Data Pelatihan

No	Input Vektor	Target (T)
1	011	1
2	010	2
3	110	2
4	101	1

3. Menetapkan kondisi awal *epoch* = 0
4. Dikerjakan jika (*epoch* \leq *MaxEpoch*)

a. *Epoch* = *epoch* + 1

b. Untuk $i = 1$ sampai 4

Data ke $i = 1$ (011)

i. Untuk $k = 1$ sampai 2

$$\text{Menghitung nilai } S_{ik} = \sqrt{\sum_{j=1}^m (x_{ij} - w_{kj})^2}$$

Kelas $k = 1$

$$S_{11} = \sqrt{(0-0)^2 + (1-0)^2 + (1-1)^2} = 1$$

Kelas $k = 2$

$$S_{12} = \sqrt{(0-1)^2 + (1-1)^2 + (1-1)^2} = 1$$

ii. Menentukan nilai S_{ik} terkecil dan simpan k

Nilai terkecil pada $k = 1$

iii. Membandingkan nilai k dengan T_{Ik}

$k = 1$ dan $T_{Ik} = 1$, jadi $k = T_{Ik}$ maka :

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) + \alpha (x_i - w_{kj}(\text{lama}))$$

$$w_{11}(\text{baru}) = 0 + 0,05 * (0-0) = 0$$

$$w_{12}(\text{baru}) = 0 + 0,05 * (1-0) = 0,05$$

$$w_{13}(\text{baru}) = 1 + 0,05 * (1-1) = 1$$

Jadi $w_1 = (0 \ 0,05 \ 1)$; $w_2 = (1 \ 1 \ 1)$

Data ke $i = 2$ (010)

i. Untuk $k = 1$ sampai 2

$$\text{Menghitung nilai } S_{ik} = \sqrt{\sum_{j=1}^m (x_{ij} - w_{kj})^2}$$

Kelas $k = 1$

$$S_{21} = \sqrt{(0-0)^2 + (1-0,05)^2 + (0-1)^2} = 1,3793$$

Kelas $k = 2$

$$S_{22} = \sqrt{(0-1)^2 + (1-1)^2 + (0-1)^2} = 1,4142$$

i. Menentukan nilai S_{ik} terkecil dan simpan k

Nilai terkecil pada $k = 1$

ii. Membandingkan nilai k dengan T_{Ik}

$k = 1$ dan $T_{Ik} = 2$, jadi $k \neq T_{Ik}$ maka :

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) - \alpha (x_i - w_{kj}(\text{lama}))$$

$$w_{11}(\text{baru}) = 0 - 0,05 * (0-0) = 0$$

$$w_{12}(\text{baru}) = 0 - 0,05 * (1-0,05) = 0,0025$$

$$w_{13}(\text{baru}) = 1 - 0,05 * (0-1) = 1,05$$

$$\text{Jadi } w_1 = (0 \ 0,0025 \ 1,05); w_2 = (1 \ 1 \ 1)$$

Data ke $i = 3$ (110)

i. Untuk $k = 1$ sampai 2

$$\text{Menghitung nilai } S_{ik} = \sqrt{\sum_{j=1}^m (x_{ij} - w_{kj})^2}$$

Kelas $k = 1$

$$S_{31} = \sqrt{(1 - 0)^2 + (1 - 0,0025)^2 + (0 - 1,05)^2} = 1,7599$$

Kelas $k = 2$

$$S_{32} = \sqrt{(1 - 1)^2 + (1 - 1)^2 + (0 - 1)^2} = 1$$

ii. Menentukan nilai S_{ik} terkecil dan simpan k

Nilai terkecil pada $k = 2$

iii. Membandingkan nilai k dengan T_{Ik}

$k = 2$ dan $T_{Ik} = 2$, jadi $k = T_{Ik}$ maka :

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) + \alpha (x_i - w_{kj}(\text{lama}))$$

$$w_{11}(\text{baru}) = 1 + 0,05 * (1-1) = 1$$

$$w_{12}(\text{baru}) = 1 + 0,05 * (1-1) = 1$$

$$w_{13}(\text{baru}) = 1 + 0,05 * (0-1) = 0,95$$

$$\text{Jadi } w_1 = (0 \ 0,0025 \ 1,05); w_2 = (1 \ 1 \ 0,95)$$

Data ke $i = 4$ (101)

i. Untuk $k = 1$ sampai 2

$$\text{Menghitung nilai } S_{ik} = \sqrt{\sum_{j=1}^m (x_{ij} - w_{kj})^2}$$

Kelas $k = 1$

$$S_{41} = \sqrt{(1 - 0)^2 + (0 - 0,0025)^2 + (1 - 1,05)^2} = 1,0012$$

Kelas $k = 2$

$$S_{42} = \sqrt{(1-1)^2 + (0-1)^2 + (1-0,95)^2} = 1,0010$$

ii. Menentukan nilai S_{ik} terkecil dan simpan k

Nilai terkecil pada $k = 2$

iii. Membandingkan nilai k dengan T_{Ik}

$k = 2$ dan $T_{Ik} = 1$, jadi $k \neq T_{Ik}$ maka :

$$w_{kj}(\text{baru}) = w_{kj}(\text{lama}) + \alpha (x_i - w_{kj}(\text{lama}))$$

$$w_{11}(\text{baru}) = 1 - 0,05 * (1-1) = 1$$

$$w_{12}(\text{baru}) = 1 - 0,05 * (0-1) = 0,05$$

$$w_{13}(\text{baru}) = 0,95 - 0,05 * (1-0,95) = 0,9475$$

Jadi $w_1 = (0 \ 0,0025 \ 1,05)$; $w_2 = (1 \ 0,05 \ 0,9475)$

Proses diteruskan sampai *epoch* ke-2, dengan menggunakan cara yang sama. Berdasarkan aturan berhenti $epoch \leq MaxEpoch$, karena jumlah *epoch* sudah sama dengan *max epoch*, yaitu 2, maka proses perhitungan berhenti hanya sampai pada *epoch* ke-2. Setelah mencapai *epoch* yang ke-2 diperoleh bobot akhir berikut.

$$w_1 = (0,0000 \ 0,0030 \ 1,0540)$$

$$w_2 = (1,0000 \ 0,0550 \ 0,9420)$$

Misal data yang akan diuji (010)

Kelas $k = 1$

$$S_1 = \sqrt{(0-0)^2 + (1-0,0030)^2 + (0-1,0540)^2} = 1,4508$$

Kelas $k = 2$

$$S_2 = \sqrt{(0-1)^2 + (1-0,0550)^2 + (0-0,9420)^2} = 1,6674$$

Diketahui nilai minimum ada pada $k = 1$ maka data (010) masuk ke kelas 1

2.4. Evaluasi Kinerja Classifier

Ukuran kinerja dari model pada *test set* seringkali berguna karena ukuran tersebut memberikan estimasi yang tidak bias dari error generalisasinya. Akurasi dari tingkat *error* yang dihitung dari *test set* dapat digunakan untuk membandingkan kinerja relatif dari *classifier* pada domain yang sama. Evaluasi kinerja *classifier* digunakan untuk mengevaluasi tingkat akurasi pada proses klasifikasi.

2.4.1. *K-Fold Cross Validation*

K-Fold Cross Validation merupakan salah satu metode yang umum digunakan untuk mengevaluasi kinerja *classifier*. Metode *K-Fold Cross Validation* dilakukan dengan membagi dataset secara acak menjadi k himpunan bagian (subset) (Kohavi, 1995). *K-Fold Cross Validation* melakukan iterasi sebanyak k kali untuk data pelatihan dan data pengujian. Metode ini berguna untuk memvalidasi keakuratan sebuah prediksi atau klasifikasi terhadap suatu data yang belum muncul dalam dataset. Dataset dibagi menjadi k subset secara acak yang masing-masing subset memiliki jumlah *instance* dan perbandingan jumlah kelas yang sama. Pembagian data ini digunakan pada proses iterasi klasifikasi. Iterasi dilakukan sesuai nilai k . Setiap iterasi satu subset digunakan untuk menguji sedangkan subset-subset lainnya digunakan untuk pelatihan.

Kelebihan dari metode ini adalah tidak adanya masalah dalam pembagian data. Setiap data akan menjadi *test set* sebanyak satu kali dan akan menjadi *training set* sebanyak $K-1$ kali. Kekurangan dari metode ini adalah algoritma pembelajaran harus dilakukan sebanyak K kali. Yang berarti menggunakan K kali waktu komputasi. Metode *K-Fold Cross Validation* pada umumnya menggunakan $k = 10$ (Fitri, 2007). Hasil uji coba metode ini dengan $k = 10$ menghasilkan rata-rata akurasi yang cukup tinggi yaitu 97% lebih baik dibandingkan metode Naïve Bayesian yang menghasilkan rata-rata akurasi 96,24% (Zamani A M, 2012).

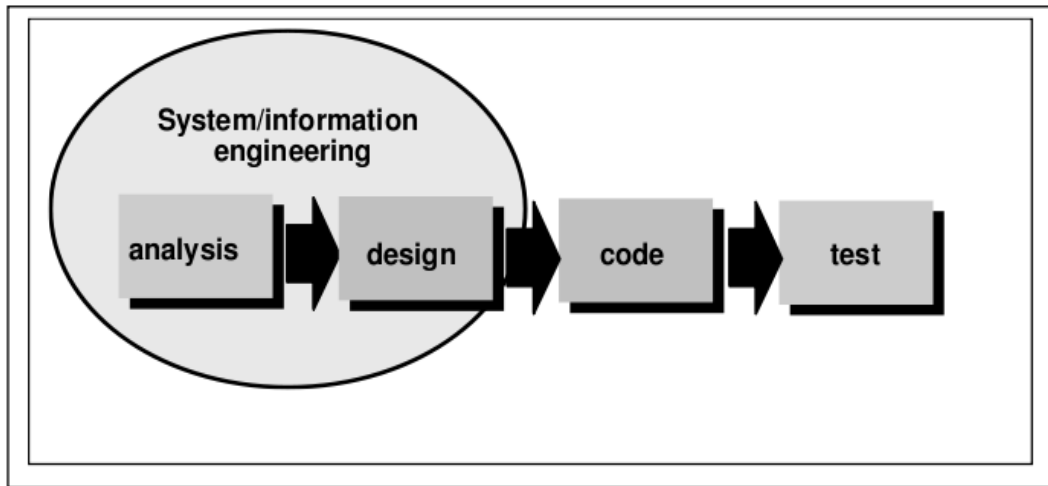
2.4.2. *Confussion Matrix*

Evaluasi dari kinerja klasifikasi didasarkan pada banyak *test record* yang diprediksi secara benar dan secara tidak benar oleh sebuah model. *Test record* ini ditabulasikan dalam sebuah tabel yang dikenal sebagai *confussion matrix*. *Confussion matrix* berisi informasi tentang klasifikasi aktual dan prediksi yang dilakukan oleh sistem klasifikasi (Kohavi, 1995). Kinerja sistem tersebut umumnya dievaluasi dengan menggunakan data dalam matriks.

2.5. Model *Waterfall*

Model *waterfall* adalah model klasik yang bersifat sistematis, berurutan dalam membangun *software* (Pressman, 2001).

Model ini juga sering disebut dengan *linear sequential model* atau *classic life cycle*. Model ini merupakan model yang paling banyak digunakan dalam *software engineering*. Disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Tahap-tahap tersebut adalah *analysis*, *design*, *code*, dan *test* yang dapat dilihat pada Gambar 2.6.



Gambar 2. 4. Model *Waterfall*

Tahap-tahap dalam model *Waterfall* adalah sebagai berikut (Pressman, 2001).

a. *Software Requirement Analysis*

Seluruh kebutuhan *software* harus bisa didapatkan dalam tahap ini, kemudian dianalisis dan didefinisikan termasuk di dalamnya kegunaan *software* yang diharapkan pengguna dan batasan *software* seperti fungsi yang dibutuhkan, perilaku, performa, dan *interface*. Tahap ini harus dikerjakan secara lengkap untuk bisa menghasilkan desain yang lengkap. Tahap ini menghasilkan SRS (*Software Requirement Specification*), ERD (*Entity Relationship Diagram*) dan DFD (*Data Flow Diagram*). SRS adalah dokumen yang berisi deskripsi lengkap mengenai kemampuan *software*. SRS diperlukan karena banyak kesalahan yang timbul pada tahap *requirement* dan tidak terdeteksi sejak dini, serta untuk menghemat biaya perbaikan. ERD menggambarkan hubungan antara objek data. Sedangkan DFD menyediakan informasi yang digunakan untuk menganalisa ruang lingkup dan sebagai dasar dari pemodelan fungsional.

Standar penulisan SRS yang digunakan adalah IEEE/ANSI 830-1998. Setiap *requirement* dalam SRS diberikan *unique identifier label*. Berikut format pendefinisian kebutuhan fungsional dalam SRS pada Tabel 2.4.

Tabel 2. 4. Tabel SRS

SRS ID	Deskripsi
SRS-XXXX-F-YY

Keterangan:

SRS : *Software Requirement Specification*

XXXX : *Nickname* sistem yang dibangun

FXX : F adalah fungsional dapat *optional* berupa NF (Non Fungsional)

YY : Nomorurut SRS

b. *Design*

Desain sistem merupakan proses desain dengan menggunakan analisis kebutuhan sebagai acuannya. Proses desain akan menerjemahkan syarat kebutuhan ke sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat pengkodean. Proses ini berfokus pada perancangan struktur data, arsitektur perangkat lunak, representasi *interface*, dan detail algoritma prosedural. Tahapan ini akan menghasilkan dokumen desain. Dokumen inilah yang akan digunakan untuk melakukan aktivitas pembuatan sistem.

c. *Code Generation*

Desain program diterjemahkan ke dalam kode-kode dengan menggunakan bahasa pemrograman yang sudah ditentukan. Tahap ini merupakan implementasi dari tahap desain yang digunakan untuk membangun sistem. Jika desain dilakukan secara detail, maka pengkodean dapat dilakukan secara mekanis. Setelah pengkodean selesai, maka akan dilakukan *testing* terhadap sistem yang telah dibuat dengan tujuan untuk mencari dan menemukan kesalahan-kesalahan pada sistem sehingga dapat diperbaiki.

d. *Testing*

Ketika proses pengkodean selesai, maka akan dilakukan *testing*. Proses *testing* berfokus pada logika di dalam *software*, memastikan semua *statement* dan fungsi eksternal telah dites, sehingga dapat menemukan kesalahan untuk dapat diperbaiki selanjutnya sehingga didapatkan hasil yang sesuai dengan yang diinginkan.

Kelebihan *Waterfall Model* adalah lebih disiplin. Selain itu, dokumentasi selalu tersedia dalam setiap tahapan (*documentation driven*). Masalah dengan *Waterfall Model*

adalah akan menyebabkan “*blocking states*”, yaitu keadaan dimana suatu tugas hanya dapat dilakukan ketika tugas sebelumnya diselesaikan. Sehingga terdapat waktu tunggu pada tugas selanjutnya untuk dapat dilakukan.

2.6. Pemodelan Analisis

Pemodelan analisis merupakan gabungan dari beberapa model yang pertama kali merepresentasikan sistem secara teknis. Terdapat dua pemodelan, yaitu pemodelan data dan pemodelan fungsional. Pemodelan analisis harus memenuhi tiga tujuan yaitu, mendeskripsikan keinginan *customer*, sebagai dasar dalam membuat desain *software*, dan mendefinisikan kebutuhan yang dapat divalidasi ketika membangun *software*.

2.6.1. Pemodelan Data

Pemodelan data menjelaskan semua hal tentang objek utama yang akan diproses, apa komposisi dari tiap objek serta atributnya, dan hubungan antar objek. Semua hal tersebut digambarkan dalam bentuk ERD (*Entity Relationship Diagram*). Keluaran yang dihasilkan adalah struktur basis data (Pressman, 2001).

ERD merupakan suatu *graph* yang menyajikan objek data, atribut dan hubungannya yang bertujuan untuk mengetahui hubungan antar objek data. Pada pemodelan data ada tiga bagian informasi yang digunakan, yaitu: (Pressman, 2001)

a. Objek Data

Objek data merupakan objek yang mewakili sesuatu informasi yang nyata dan dapat dibedakan dari sesuatu yang lain. Objek data harus berhubungan satu sama lainnya. Properti dari objek data antara lain: nama, daftar atribut, dan *primary key*.

b. Relasi

Hubungan antara sejumlah objek data yang berasal dari himpunan objek yang data berbeda. Relasi yang terjadi di antara dua himpunan objek data (misalnya A dan B) dalam satu basis data yaitu:

i. One to one (1:1).

Hubungan satu ke satu yaitu setiap *instance* pada objek data A berhubungan paling banyak dengan satu *instance* pada objek data B.

ii. One to many (1:N).

Setiap *instance* pada objek data A dapat berhubungan dengan banyak *instance* pada objek data B, tetapi setiap *instance* pada objek data B dapat berhubungan dengan satu *instance* pada objek data A.


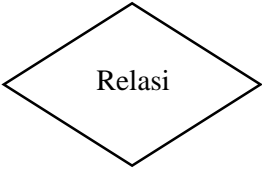

iii. *Many to many* (M:N).

Setiap *instance* pada objek data A dapat berhubungan dengan banyak *instance* pada objek data B.

c. Atribut

Atribut mendefinisikan properti dari objek data. Satu atau lebih atribut dari suatu objek data harus menjadi penanda sebagai *primary key* yang membedakan dengan objek data lain. Berikut ini adalah tabel notasi pemodelan data dapat dilihat pada Tabel 2.5.

Tabel 2. 5. Tabel Notasi Pemodelan Data

Komponen	Deskripsi
	Objek data adalah sebuah objek yang dapat dibedakan dengan objek lain.
	Relasi menunjukkan adanya hubungan di antara sejumlah objek data yang berbeda.
	Garis sebagai penghubung antara relasi dengan objek data, dan objek data dengan atribut
1 : 1 1 : N M : N	Kardinalitas merupakan angka yang menunjukkan banyaknya kemunculan suatu objek terkait dengan kemunculan objek lain pada relasi.

2.6.2. Pemodelan Fungsional

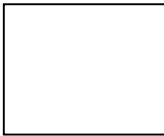


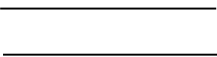

Seluruh fungsi yang tercakup dalam sistem ini digambarkan dalam *Data Context Diagram* (DCD) dan *Data Flow Diagram* (DFD). DCD dapat juga dikatakan sebagai DFD Level-0 atau *Context Diagram* (Pressman, 2001). Untuk DFD merupakan penjabaran lebih lanjut dari DCD. DFD berguna untuk

menggambarkan fungsi-fungsi yang mentransformasikan data, serta berguna untuk menggambarkan bagaimana data ditransformasikan pada perangkat lunak.

Data Context Diagram (DCD) merupakan tingkatan tertinggi dalam diagram aliran data yang hanya memuat satu proses, dan menunjukkan sistem secara keseluruhan. Proses tersebut diberi nomor nol. Semua entitas eksternal yang ditunjukkan pada diagram konteks berikut aliran-aliran data utama menuju dan dari sistem. Diagram tersebut tidak memuat penyimpanan data dan tampak sederhana untuk diciptakan, entitas-entitas eksternal serta aliran-aliran data menuju dan dari sistem diketahui oleh seorang analis dari wawancara dengan pengguna sebagai hasil analisis dokumen.

Data Flow Diagram (DFD) atau dapat juga disebut dengan model sistem fundamental dan *bubble chart* adalah representasi grafik dari aliran informasi serta transformasi atau proses yang dilakukan mulai dari data dimasukkan sampai mendapatkan *output* (Pressman, 2001). DFD menggambarkan komponen-komponen sebuah sistem, aliran-aliran data di mana komponen-komponen tersebut, asal, tujuan, dan penyimpanan dari data tersebut. DFD digunakan untuk dua hal utama, yaitu untuk membuat dokumentasi dari sistem informasi yang ada serta untuk menyusun dokumentasi untuk sistem informasi yang baru. DFD mempunyai empat komponen utama, yaitu *external entity*, *data flow*, proses, dan *data store*. Notasi yang digunakan untuk pemodelan fungsional dapat dilihat pada Tabel 2.6.

Tabel 2. 6. Tabel Notasi Pemodelan Fungsional

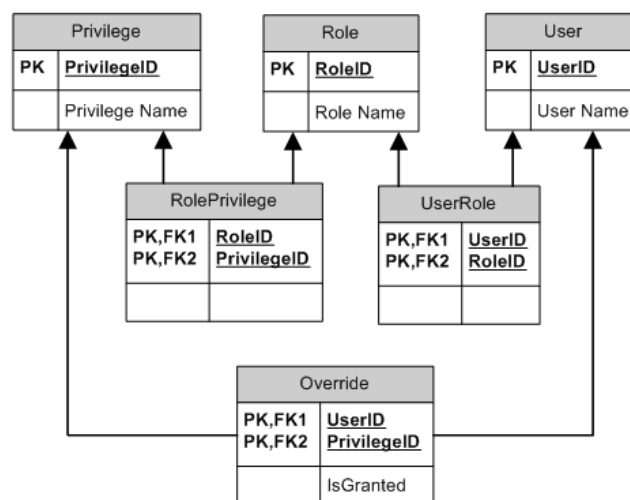
Notasi	Keterangan
	<i>External Entity</i> (Entitas Eksternal)
	<i>Data Flow</i> (Aliran Data)
	<i>Process</i> (Proses)
	<i>Data Store</i>
	<i>Split / merge</i>

2.7. Perancangan Struktur Data

Struktur data merupakan sebuah representasi dari hubungan logis antara elemen individual dari data. Karena struktur informasi akan selalu mempengaruhi perancangan prosedural akhir, struktur data sama pentingnya dengan struktur program. Perancangan struktur data mentransformasikan pemodelan data yang telah dibuat pada fase analisis menjadi struktur data yang akan digunakan dalam implementasi perangkat lunak. Desain atau perancangan struktur data meliputi level pemodelan yaitu *Conceptual Data Model (CDM)* dan *Physical Data Model (PDM)*.

1. *Conceptual Data Model (CDM)*

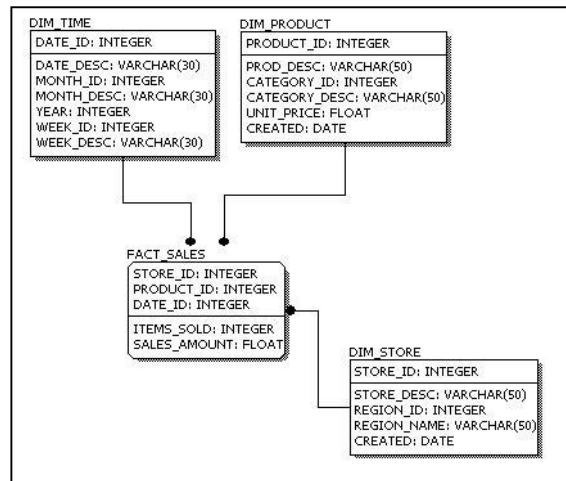
CDM merupakan model data dalam bentuk konseptual yang berisi hubungan antar entitas dalam sistem yang dibangun. CDM dibuat sudah dalam bentuk tabel-tabel tanpa tipe data yang menggambarkan relasi antar tabel untuk keperluan implementasi ke basis data. CDM merupakan hasil penjabaran lanjut dari ERD. (Rosa & M.Shalahuddin, 2013). Berikut ilustrasi contoh dari CDM yang dapat dilihat pada Gambar 2.7.



Gambar 2. 5. Contoh *Conceptual Data Model (CDM)*

2. *Physical Data Model (PDM)*

PDM merupakan hasil mapping dari CDM, merupakan model data yang menggunakan sejumlah tabel untuk menggambarkan data serta hubungan antar data. Setiap tabel mempunyai sejumlah kolom dimana setiap kolom memiliki nama yang unik beserta tipe datanya. PDM merupakan bentuk fisik perancangan basis data yang sudah siap diimplementasikan ke dalam DBMS (Rosa & M.Shalahuddin, 2013). Berikut ilustrasi contoh PDM yang dapat dilihat pada Gambar 2.8.



Gambar 2. 6. Contoh *Physical Data Model* (PDM)

2.8. Bahasa Pemrograman PHP

PHP (*PHP Hypertext Preprocessor*) adalah bahasa pemrograman *scripting* sisi *server* artinya sintaks-sintaks dan perintah yang diberikan akan sepenuhnya dijalankan oleh *server* tetapi disertakan pada halaman HTML biasa. Sistem yang dibangun oleh PHP pada umumnya akan memberikan hasil pada web browser, tetapi prosesnya secara keseluruhan dijalankan di *server*. Pada prinsipnya *server* akan bekerja apabila ada permintaan dari *client*. Dalam hal ini *client* menggunakan kode-kode PHP untuk mengirimkan permintaan ke *server*.

Kelebihan PHP dari bahasa pemrograman lain (Peranginangin, 2006)

- Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.
- Web Server* yang mendukung PHP dapat ditemukan di mana-mana dari mulai Apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
- Lebih mudah dalam pengembangan.
- Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
- PHP adalah *open source* yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.9. DBMS MySQL

MySQL adalah sistem manajemen *database* SQL yang bersifat *open source* dan paling populer saat ini. Sistem database MySQL mendukung beberapa fitur seperti *multi-threaded*, *multi-user*, dan *SQL database management system* (DBMS). Database ini dibuat untuk keperluan sistem database yang cepat, handal dan mudah digunakan (Fathansyah, 2007).

Berikut ini beberapa kelebihan MySQL sebagai *database server* antara lain (Fathansyah, 2007):

- a. *Source* MySQL dapat diperoleh dengan mudah dan gratis.
- b. Sintaksnya lebih mudah dipahami dan tidak rumit.
- c. Pengaksesan basis data dapat dilakukan dengan mudah.
- d. MySQL merupakan program yang *multi-threaded*, sehingga dapat dipasang pada *server* yang memiliki *multi* CPU.
- e. Didukung program-program umum seperti C, C++, Java, Perl, PHP, Python, dan lain sebagainya.
- f. Bekerja pada berbagai platform.

2.10. Pengujian Perangkat Lunak

Menurut IEEE, pengujian perangkat lunak adalah proses sistem operasi atau komponen menurut kondisi tertentu, pengamatan atau pencatatan hasil dan mengevaluasi beberapa aspek sistem atau komponen, proses analisis item perangkat lunak untuk mendeteksi perbedaan antara kondisi yang ada dengan yang diinginkan dan mengevaluasi fitur perangkat lunak. Tujuan pengujian perangkat lunak (Peranginangin, 2006):

- a. Tujuan langsung
 - i. Identifikasi dan menemukan beberapa kesalahan yang mungkin ada dalam perangkat lunak yang diuji.
 - ii. Setelah perangkat lunak dibetulkan, diidentifikasi lagi kesalahan dan dites ulang untuk menjamin kualitas level penerimaan.
 - iii. Membentuk tes yang efisien dan efektif dengan anggaran dan jadwal yang terbatas.

- b. Tujuan tidak langsung

Mengumpulkan daftar kesalahan untuk digunakan dalam daftar pencegahan kesalahan. Terdapat dua strategi pengujian perangkat lunak berdasarkan konsep pengujian sebagai berikut.

- i. *Black box (functionality) testing*. Mengidentifikasi kesalahan yang berhubungan dengan kesalahan fungsionalitas perangkat lunak yang tampak dalam kesalahan *output*. Definisi menurut IEEE, adalah pengujian yang mengabaikan mekanisme internal sistem atau komponen dan fokus semata-mata pada *output* yang dihasilkan yang merespon *input* yang dipilih dan kondisi eksekusi. Pengujian
- ii. yang dilakukan untuk mengevaluasi pemenuhan sistem atau komponen dengan kebutuhan fungsional tertentu.

- iii. *White box (structural) testing*. Memeriksa kalkulasi internal *path* untuk mengidentifikasi kesalahan. Definisi menurut IEEE, adalah pengujian yang memegang perhitungan mekanisme internal sistem atau komponen

Format pendefinisian rencana pengujian dengan standar penulisan IEEE dapat dilihat pada pada Tabel 2.7.

KELAS UJI	BUTIR UJI	IDENTIFIKASI		TINGKAT PENGUJIAN	JENIS PENGUJIAN	PENGUJI
		SKPL	DUPL			

Tabel 2. 7. Format Pendefinisian Rencana Pengujian

Keterangan:

Kelas Uji : Kelompok pengujian, misalnya pengujian *login*

Butir Uji: Sub kelompok pengujian

SKPL: Spesifikasi Kebutuhan Perangkat Lunak (SRS)

DUPL: Dokumen Uji Perangkat Lunak. Mengacu pada butir uji.

Tingkat Pengujian: Level dimana pengujian dilakukan (pengujian sistem, regresi, integrasi, dsb)

Jenis Pengujian: *Black-box*, *white-box*, dsb

Penguji: Subjek pengujian

Format pendefinisian hasil pengujian dengan standar penulisan IEEE dapat dilihat pada pada Tabel 2.8.

Tabel 2. 8. Format Pendefinisian Hasil Pengujian

IDENTIFIKASI	DESKRIPSI	PROSEDUR PENGUJIAN	MASUKAN	KELUARAN YANG DIHARAPKAN	KRITERIA EVALUASI HASIL	HASIL YANG DIDAPAT	KESIMPULAN

Keterangan:

Identifikasi: Mengacu pada DUPL.

Deskripsi: Mengacu pada kelas uji.

Prosedur Pengujian: Cara bagaimana pengujian dilakukan.

Masukan: nilai yang menjadi inputan.

Keluaran yang diharapkan: *Output* yang diharapkan.

Kriteria Evaluasi Hasil: Misalnya jika *username/password* salah, maka keluar pesan ..., jika tidak keluar pesan ...

Hasil Yang Didapat: Kejadian yang muncul.

Kesimpulan: Diterima atau ditolak.

BAB III

METODOLOGI PENELITIAN

Bab ini membahas tentang metodologi penelitian yang digunakan dalam implementasi *Learning Vector Quantization* (LVQ) pada sistem deteksi penyakit diabetes melitus meliputi metodologi penelitian, arsitektur sistem, garis besar penyelesaian masalah, dan jadwal.

3.1. Metodologi Penelitian

Ada dua metodologi yang digunakan dalam membuat proposal tugas akhir ini, yaitu studi pustaka dan pengumpulan data.

3.1.1. Studi Pustaka

Metodologi yang digunakan dalam menyusun proposal tugas akhir ini adalah studi kepustakaan. Studi pustaka seperti yang dikutip oleh (perkuliahan.com, 2016) bertujuan untuk memperoleh informasi yang sesuai dengan topik ataupun masalah yang sedang diteliti. Topik yang diangkat pada penelitian ini mengenai deteksi penyakit diabetes melitus dan perhitungan akurasi. Informasi yang diperlukan dapat diperoleh dari literatur buku, laporan penelitian, jurnal, karangan ilmiah, ensiklopedia, laporan berkala, ketetapan-ketetapan, tesis dan sumber-sumber tertulis baik cetak maupun elektronik lain. Sumber-sumber tersebut diharapkan dapat membantu pemecahan masalah dan menghasilkan keluaran yang bermanfaat pada pemahaman tentang teori dan rumus yang akan digunakan.

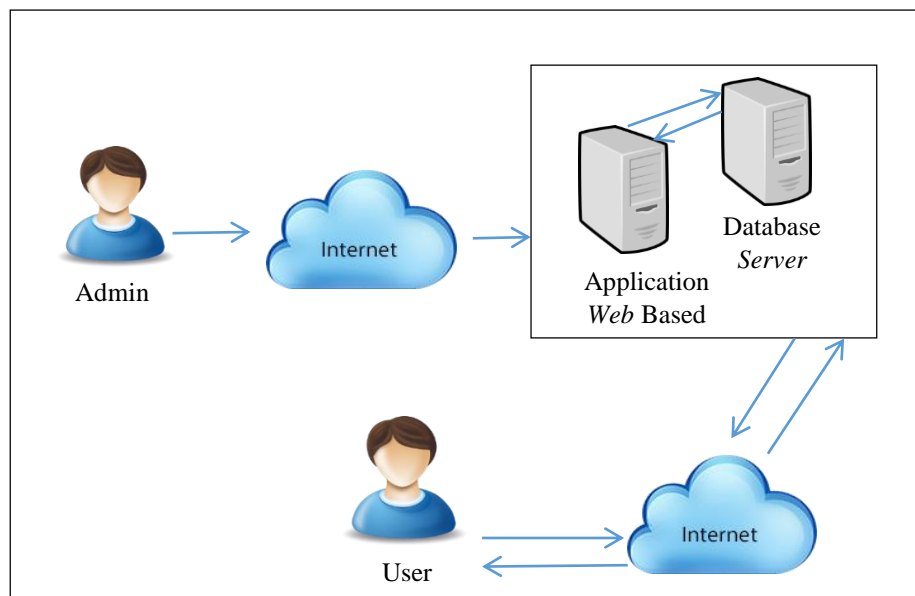
3.1.2. Pengumpulan Data

Pengumpulan data dalam penelitian ini menggunakan metode observasi. Observasi merupakan metode pengambilan data melalui pengamatan secara langsung terhadap obyek penelitian. Observasi yang dilakukan adalah dengan cara menganalisa data rekam medis dari pasien-pasien penderita diabetes melitus yang diperoleh dari Rumah Sakit Pusat Pertamina, Jakarta. Data yang diperlukan yaitu data rekam medis pasien penyakit diabetes melitus sebanyak 100 data.

3.2. Arsitektur Sistem

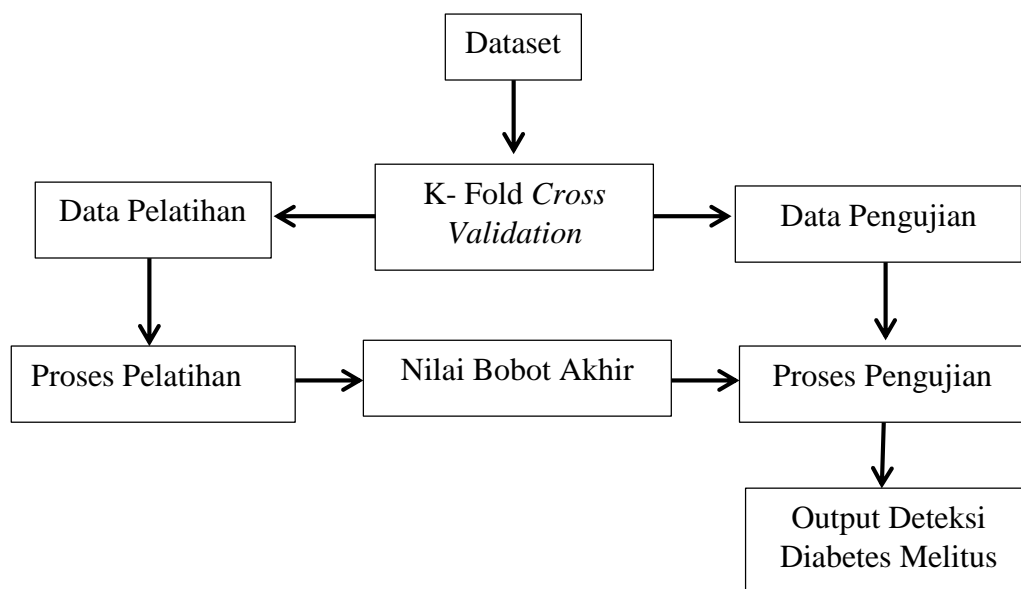
Sistem pendeteksian diabetes melitus ini dibangun berbasis web sehingga membutuhkan internet untuk dapat terhubung serta memiliki *database server* untuk penyimpanan data. *User*

dalam mengakses sistem ini juga harus terhubung dengan internet. Gambaran mengenai arsitektur sistem ini dapat dilihat pada Gambar 3.1.



Gambar 3. 1. Arsitektur Sistem Pendeteksian Diabetes Melitus Menggunakan Jaringan Syaraf Tiruan LVQ

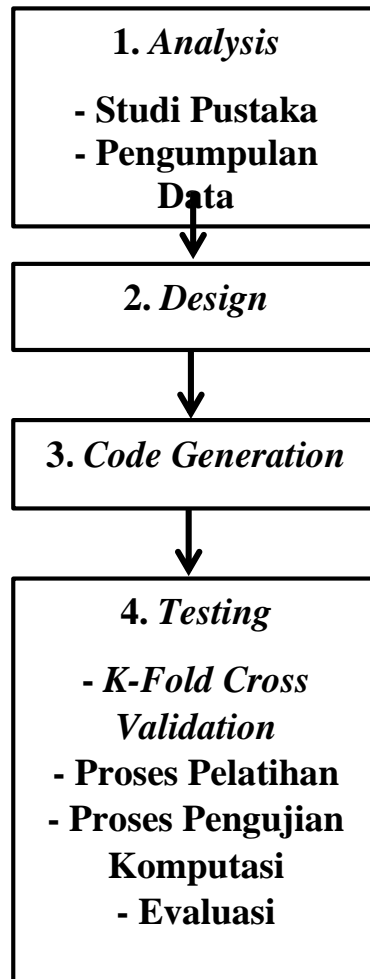
Sedangkan untuk tahapan alur pengolahan data deteksi penyakit diabetes melitus di dalam sistem dapat dilihat pada Gambar 3.2.



Gambar 3. 2. Tahapan Alur Pengolahan Data Deteksi Diabetes Melitus

3.3. Garis Besar Penyelesaian Masalah

Tahapan penyelesaian masalah pada topik penelitian, lebih khusus pada metode pengolahan data menggunakan *Learning Vector Quantization* dapat dilihat pada Gambar 3.3.



Gambar 3. 3. *Flow* Garis Besar Penyelesaian Masalah

1. *Analysis*

1.1. Studi Pustaka

Mencari sumber-sumber yang diharapkan dapat membantu pemecahan masalah dan menghasilkan keluaran yang bermanfaat pada pemahaman tentang teori dan rumus yang akan digunakan.

1.2. Pengumpulan Data

Mencari data yang digunakan untuk melatih sistem, yaitu berupa data rekam medis penyakit diabetes melitus.

2. Design

Proses ini berfokus pada perancangan struktur data, arsitektur perangkat lunak, representasi *interface*, dan detail algoritma prosedural. Tahapan ini akan menghasilkan dokumen desain.

3. Code Generation

Tahap ini merupakan implementasi dari tahap desain yang digunakan untuk membangun sistem. Kemudian akan dilakukan *testing* menggunakan metode *white box* dan *black box*..

4. Testing

4.1. Normalisasi Data

Pada penelitian ini setiap gejala perlu dilakukan normalisasi karena data input belum dalam rentang [0,1]. Normalisasi data bertujuan untuk menghindari terjadinya kesalahan, anomali data dan tidak konsistensinya data sehingga data nantinya dapat dilatih dan memperoleh hasil pengujian yang maksimal.

4.2. K-Fold Cross Validation

Setelah dilakukan normalisasi, kemudian data akan diproses menggunakan K-Fold Cross Validation. Dengan menggunakan $K = 10$, dataset akan dibagi menjadi 10 partisi dimana tiap partisi akan memiliki jumlah kelas yang sama.

4.3. Proses Pelatihan

Proses pelatihan bertujuan untuk melatih metode *Learning Vector Quantization* untuk mengenali suatu pola pada data diabetes melitus. Pertama menghitung bobot dengan mencari nilai *euclidean distance*, kemudian dilakukan perubahan bobot pada kelas yang memiliki nilai *euclidean distance* terkecil. Hasil dari proses ini adalah didapatkannya nilai bobot yang akan digunakan pada proses pengujian. Data-data yang akan digunakan pada proses pelatihan dapat dilihat pada Tabel 3.1.

Tabel 3. 1. Tabel Contoh Data Diabetes Melitus

Sample	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	T
Data 1	0	1	0	0	0	0	0	1	0	0	0	0	1
Data 2	0	0	1	0	1	1	0	1	1	0	1	1	0
Data 3	1	1	0	1	0	0	1	1	1	1	1	0	1
Data 4	1	0	0	1	0	1	1	0	1	1	1	1	1
Data 5	1	0	1	0	1	0	0	0	1	0	1	1	0
Data 6	0	0	0	0	0	0	0	0	0	0	0	1	0

Penjelasan :

x_1 = Umur (> 40 tahun = 1, ≤ 40 tahun = 0)

x_2 = Jenis kelamin (Pria = 1, Wanita = 0)

- x_3 = Aseton urin puasa ($>0 = 1, \leq 0 = 0$)
 x_4 = Aseton urin 2 jam sebelum puasa ($>0 = 1, \leq 0 = 0$)
 x_5 = Kolesterol total ($>200 = 1, \leq 200 = 0$)
 x_6 = Glukosa darah puasa ($>140 = 1, \leq 140 = 0$)
 x_7 = Glukosa darah 2 jam sebelum puasa ($\geq 140 = 1, < 140 = 0$)
 x_8 = Kolesterol Hdl ($\geq 60 = 1, < 60 = 0$)
 x_9 = Kolesterol Ldl ($>130 = 1, \leq 130 = 0$)
 x_{10} = Trigliserida ($\geq 150 = 1, < 150 = 0$)
 x_{11} = Glukosa urin 2 jam sebelum puasa ($>0 = 1, \leq 0 = 0$)
 x_{12} = Glukosa urin puasa ($>0 = 1, \leq 0 = 0$)
 T = Target (Positif = 1, Negatif = 0)

4.4. Proses Pengujian

Proses pengujian merupakan proses untuk menghasilkan hasil uji. Menggunakan data pengujian dan bobot akhir yang diperoleh pada proses pelatihan. Kemudian dengan menggunakan *euclidean distance* untuk mencari jarak pada data pengujian dan bobot akhir dari masing-masing kelas. Kelas yang memiliki nilai *euclidean distance* terkecil akan dikenali sebagai kelas dari hasil prediksi diabetes melitus.

4.5. Evaluasi

Evaluasi akurasi dari klasifikasi pada penelitian ini dilakukan dengan menggunakan *confussion matrix* dengan keluaran *accuracy* dan *error rate* untuk menganalisa keberhasilan algoritma dalam melakukan deteksi penyakit diabetes melitus. *Accuracy* merupakan perbandingan dari kasus yang diidentifikasi benar dengan jumlah keseluruhan kasus, sedangkan *error rate* merupakan perbandingan dari kasus yang diidentifikasi salah dengan jumlah keseluruhan kasus. Tabel *confussion matrix* dengan dua kelas dapat dilihat pada Tabel 3.2.

Tabel 3. 2. Tabel *Confussion Matrix* Dengan Dua Kelas

Prediksi	Aktual	
	T ₁	T ₂
T ₁	M ₁₁	M ₁₂
T ₂	M ₂₁	M ₂₂

Penjelasan :

$M_{11} - M_{12}$ = Jumlah data kelas 1-2 (T_1-T_2) yang di prediksi di kelas 1 (T_1)

$M_{21} - M_{22}$ = Jumlah data kelas 1-2 (T_1-T_2) yang di prediksi di kelas 2 (T_2)

$$akurasi = \frac{\text{banyaknya prediksi yang benar}}{\text{total banyaknya prediksi}} = \frac{M_{11} + M_{22}}{M_{11} + M_{12} + M_{21} + M_{22}}$$

$$error\ rate = \frac{\text{banyaknya prediksi yang salah}}{\text{total banyaknya prediksi}} = \frac{M_{12} + M_{21}}{M_{11} + M_{12} + M_{21} + M_{22}}$$

3.4. Jadwal

Jadwal penelitian merupakan estimasi waktu mulai dari persiapan, perhitungan, pembuatan sistem hingga penyelesaian sistem. Kegiatan tugas akhir dilaksanakan pada Juli 2017 hingga Desember 2017. Jadwal kegiatan dapat memberikan gambaran mengenai tahapan yang dilakukan dan dapat dilihat pada Tabel 3.3

Tabel 3. 3. Jadwal Rencana Tugas Akhir

Aktifitas	Jadwal																							
	Juli 2017				Agustus 2017				September 2017				Oktober 2017				November 2017				Desember 2017			
Minggu ke-	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Studi pustaka																								
Penyusunan proposal TA1																								
Seminar TA1																								
Analysis																								
Design																								
Code Generation																								
Testing																								
Penyusunan laporan TA2																								
Sidang TA2																								


DAFTAR PUSTAKA

- Ariza, M. A. (2010). *The economic consequences of diabetes and cardiovascular disease in the United States*. Boston: University School of Medicine.
- Budianita, E. (2013). Penerapan Learning Vector Quantization untuk klasifikasi status gizi anak. *Indonesian Journal of Computing and Cybernetics Systems*.
- Fagustina, A., Syaifuddin, K., Ardita, V. R., & Hakim, F. A. (2011). *Memprediksi penyakit jantung koroner dengan menggunakan algoritma LVQ*. Surakarta: Universitas Sebelas Maret.
- Fathansyah. (2007). *Buku Teks Komputer Basis Data*. Bandung: Informatika.
- Fitri, R. (2007). Retrieved November 09, 2015, from digilib.itb.ac.id/files/.../jbptitbpp-gdl-radenfitri-28974-3-2007ta-2.pdf
- Gelinas, U. J., Oram, A. E., & Wiggins, W. P. (1990). *Accounting Information System*. PWS-KENT: Publishing Company.
- Hermawan, A. (2006). *Jaringan Syaraf Tiruan Teori dan Aplikasi*. Yogyakarta: Penerbit Andi.
- Hill, J. (2011). *Diabetes monitoring: risk factors, complications and management*. Birmingham: Birmingham Community Healthcare NHS Trust.
- Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation. *International Joint Conference on Artificial Intelligence (IJCAI)*.
- Leman. (1998). *Metodologi Pengembangan Sistem Informasi*. Jakarta: PT Elex Media Komputindo.
- Peranginangin, K. (2006). *Aplikasi Web dengan PHP dan MySQL*. Yogyakarta: Penerbit Andi.
- Praet, S. F. (2009). *Exercise therapy in type 2 diabetes*. Rotterdam: Erasmus University Medical Center.
- Pressman, R. S. (2001). *Software Engineering : A Practitioner's Approach Fifth Edition*. McGraw - Hill Companies, Inc.
- Pusptaningrum, D. (2006). *Pengantar jaringan Syaraf Tiruan*. Yogyakarta: Penerbit Andi.
- Ramachandran, A. d. (2004). Temporal changes in prevalence of diabetes and impaired glucose tolerance associated with lifestyle transition occurring in the rural population in India. *Diabetologia*, 860-865.
- Ranadhi, D., Indarto, W., & Hidayat, T. (2006). *Implementasi Learning Vector Quantization Untuk Pengenalan Pola Sidik Jari Pada Sistem Informasi Narapidana LP Wirogunan*. Media Informatika.
- Rosa, A., & M. Shalahuddin. (2013). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek* (1st ed.). Bandung: Informatika.
- T. Sutojo, E. M. (2011). *Kecerdasan Buatan*. Yogyakarta: ANDI Yogyakarta.

- Waspadji, S. (2007). *Diabetes Melitus : Apakah itu. Dalam Hidup Sehat dengan Diabetes*. Jakarta: Balai Penerbit FKUI.
- Zamani A M, A. B. (2012). Implementasi Algoritma Genetika pada Struktur Backpropagation Neural Network untuk Klasifikasi Kanker Payudara. *Jurnal Teknik ITS*, 222.

LAMPIRAN

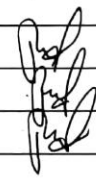
Lampiran 1. Kartu Bimbingan Tugas Akhir



KARTU BIMBINGAN TUGAS AKHIR
JURUSAN ILMU KOMPUTER / INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO

Data Peserta TA

Nama	: REYHAN SYARIF
NIM	: 24010313130104
Topik Riset	: SISTEM CERDAS
Judul Tugas Akhir	: SISTEM PENDETEKSIAN DIABETES MELITUS MENGGUNAKAN JARINGAN SYARAF TIRUAN LQ (LEARNING VECTOR QUANTIZATION)

Diisi oleh koordinator TA					Persetujuan pembimbing TA	
Tanggal Registrasi	Semester Ke -	Status TA*	Tahun Ajaran*	Catatan	Nama Dosen Pembimbing	Paraf Pembimbing
/ /20	6	B/L/U	/			
/ /20	7	B/L/U	/		Priyo Sodik S, S.Si, M.Kom	
/ /20	8	B/L/U	/		Priyo Sodik S, S.Si, M.Kom	
/ /20	9	B/L/U	/		Priyo Sodik S, S.Si, M.Kom	
/ /20	10	B/L/U	/			
/ /20	11	B/L/U	/			
/ /20	12	B/L/U	/			
/ /20	13	B/L/U	/			
/ /20	14	B/L/U	/			

*) (B)aru, (L)anjut, (U)lang

Koordinator Tugas Akhir

Helmie Arif Wibawa, S.Si, M.Cs
NIP. 19780516 200312 1 001

Catatan:

- Bawalah kartu ini pada waktu bimbingan TA.
- Setiap selesai Registrasi perpanjangan, kartu bimbingan di foto kopi kemudian diserahkan kepada pembimbing TA (sebagai arsip).
- Jaga dan rawat kartu ini hingga lulus. Jangan dilipat, jangan terkena air, dan jangan sampai hilang.
- Jika kartu ini rusak atau hilang, mahasiswa segera mengajukan permintaan kartu bimbingan pengganti dengan menyertakan Surat Pernyataan tidak rusak atau hilang lagi, bermaterai, dan wajib melengkapi semua data-data yang telah diisikan pada kartu sebelumnya.


Halaman 1 dari 2

Catatan Pembimbingan

No	Tanggal	Paraf Peserta TA	Catatan Materi	Paraf Pembimbing
1	12/06-2017		Judul + Latar Belakang	
2	14/06-2017		Bab 1	
3	11/08-2017		Revisi Bab 1	
4	14/08-2017		Bab 2 & Bab 3	
5	23/08-2017		Revisi Bab 2 & Bab 3	
6	25/08-2017		Revisi Bab 2 & Bab 3	
7	28/08-2017		Revisi Bab 2 & Bab 3	
8	1/09-2017		Revisi Bab 2 & Bab 3	
9	11/09-2017		Ace	
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				

No	Tanggal	Persetujuan Pembimbing TA	Paraf Pembimbing TA	Catatan/ Keterangan
1	11/09-2017	Telah disetujui seminar Tugas Akhir 1		
2		Telah menyerahkan Laporan TA1		
3		Telah disetujui Ujian Tugas Akhir 2		
4		Telah menyerahkan Laporan TA 2		
5		Telah membuat Artikel Ilmiah		

Lampiran 2. Kartu Keikutsertaan Seminar TA1

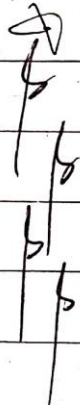


KARTU KEIKUTSERTAAN SEMINAR TA1

JURUSAN ILMU KOMPUTER/ INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO

Nama : REYHAN SYARIF

NIM : 24010313130104

No	Seminar yang Diikuti				TTD Dosen
	Tgl	Judul Tugas Akhir	Oleh: (Nama/NIM)	Keaktifan	
1	26/10-16	Implementasi telegram API dan Raspberry Pi 3 pada sistem penunjang suhu, pH, dan kelembaban	Kevin Prabowo.T 24010312130035	X	
2.	23/11-16	SISTEM Pemodelan efisiensi unit kerja maskapai penerbangan menggunakan metode DEA pemodelan BCC output oriented	Kelvin Ika Pratita Amriza W.P. 24010313130079	X	
3.	23/11-16	Sistem pemodelan efisiensi unit kerja maskapai penerbangan menggunakan metode DEA pemodelan BCC input oriented	Mirza .C. G. 24010313140038	X	
4.	23/11-16	Sistem pemodelan efisiensi unit kerja maskapai penerbangan menggunakan metode DEA pemodelan CCR output oriented	Pizka .R. 24010313130091	X	
5.	23/11-16	Sistem pemodelan efisiensi unit kerja maskapai penerbangan menggunakan metode DEA pemodelan input oriented	Arial F. 24010313130090	X	

Keterangan:

Aktif : ✓ (Nilai 4)

Tidak aktif : X (Nilai 2)

Koordinator Tugas Akhir

Helmie Arif Wibawa, S.Si, M.Cs
NIP. 19780516 200312 1 001

Lampiran 3. Daftar Hadir Seminar Proposal Tugas Akhir

DAFTAR HADIR SEMINAR PROPOSAL TUGAS AKHIR

Hari/Tanggal : Jumat, 15 September 2017
 Nama Mahasiswa : Reyhan Syarif
 Nomor Induk Mahasiswa : 24010313130104
 Departemen : Ilmu Komputer/ Informatika
 Judul Tugas Akhir : Sistem Pendeteksian Diabetes Melitus Menggunakan Jaringan Syaraf Tiruan LVQ (*Learning Vector Quantization*)

No.	Nama	NIM	Tanda Tangan
1.	Dini Aulia Dewi	24010313120002	1.
2.	Nora Herna	24010313120046	2.
3.	Fita Rosmanira	24010313120059	3.
4.	Ishaq Tanjung	24010313120056	4.
5.	Puty Dwi Amunda	24010313120033	5.
6.	Henas Ayu N.H.	24010313120035	6.
7.	M. Fagihun Arman	24010313120025	7.
8.	Nur Amalina	24010313120030	8.
9.	Harks Prasetya	24010313120058	9.
10.	Darmawan Nur Kusuma	24010313120029	10.
11.	Fabian Alhajili	24010313140127	11.
12.	Miliza C. B.	24010313140088	12.
13.	Amriza Wibowo P.	24010313130079	13.
14.	Alfira Zaki	24010313120036	14.
15.	Dicky Fabio S.	24010313120057	15.
16.	Inhad Kamil	24010313130098	16.
17.	Bagus A.	24010313120034	17.
18.	Ayu Ruyudita	24010313120024	18.
19.	Pesa Ilham Maulana	24010313120028	19.
20.	Andres Mangara	24010313130120	20.
21.	Praga Rin Prayogo	24010313130068	21.

Lampiran 4. Notulensi Seminar Proposal Tugas Akhir

1. Nama: Dini

NIM: 2002

Pertanyaan: Input serta outputnya berbentuk kata-kata atau bagaimana?

Jawaban: Input berupa data lab dalam bentuk angka dengan range 0-1, sedangkan output berupa penentuan apakah positif diabetes atau tidak.

2. Nama: Tanjung

NIM: 20056

Pertanyaan: Max Epoch ditentukan sendiri atau tidak? Dan berapa?

Jawaban: Ditentukan sendiri, max epoch berjumlah 100 agar didapat hasil prediksi yang lebih akurat.

3. Nama: Hemas

NIM: 20035

Pertanyaan: Karena sistem berhubungan dengan kesehatan, apakah diagnosis berbeda dengan diagnosis dokter?

Jawaban: Sistem menggunakan kriteria dokter, nantinya akan digunakan oleh staf medis untuk membantu proses diagnosa lebih cepat.

4. Nama: Harits

NIM: 20058

Pertanyaan: Menggunakan berapa subset untuk k fold? Dan pengelompokan datanya bagaimana?

Jawaban: Subset berjumlah 10, dataset berjumlah 100, dengan data pelatihan 90 dan data pengujian 10

5. Nama: Nora

NIM: 20046

Pertanyaan: Alasan menggunakan metodenya? Dan kelebihananya.

Jawaban: LVQ menghasilkan prediksi dengan akurasi yang tinggi, tingkat error rendah dan komputasinya cukup mudah.

6. Nama: Falah

NIM: 40132

Pertanyaan: Apakah menggunakan framework tambahan?

Jawaban: Karena konvensional maka tidak menggunakan framework

7. Nama: Faqih

NIM: 20019

Pertanyaan: Mengapa memilih kasus tersebut? Apakah metode yang digunakan bisa untuk kasus pada bidang lain?

Jawaban: Diagnosis merupakan salah satu penyakit dengan jumlah penderita terbanyak maka dirasa perlu untuk mengangkat kasus ini, LVQ jg bias diterapkan pada bidang lain misalnya image recognition

8. Nama: Jihad

NIM: 30098

Pertanyaan: Apakah sudah melakukan perhitungan manual sebelumnya? Bagaimanaantisipasi jika akurasi kurang bagus?

Jawaban: sudah dicoba, tapi belum sampai tahap menentukan akurasi,antisipasi untuk saat ini belum ada karena dari jurnal yang ada lvq menghasilkan tingkat akurasi yang baik.

9. Nama: Pak Priyo

Pertanyaan: Kesan pesan selama mencari judul, menyusun proposal, apakah waktu yang dijadwalkan cukup atau tidak?

Jawaban: Cukup cepat dalam menyusun proposal, jika sesuai dengan yang dijadwalkan mudah-mudahan cukup.