BAB 3 DIMENSION BASED FNLVQ

Pada bab ini dijelaskan mengenai modifikasi yang dilakukan pada algoritma pembelajaran FNLVQ yang sudah dibahas sebelumnya pada Bab 2. Algoritma hasil modifikasi dibahas secara lengkap pada Bab 3 ini. Pada bab ini juga akan dibahas berbagai skenario eksperimen yang dilakukan terhadap algoritma ini, termasuk eksperimen dengan data murni dan data *noisy.*

3.1 Dimension Based-FNLVQ

Dalam rangka meningkatkan tingkat pengenalan yang mampu dicapai jaringan FNLVQ, dilakukanlah modifikasi terhadap algoritma pembelajarannya. Modifikasi yang dilakukan bertujuan untuk meningkatkan presisi pembelajaran dan pada akhirnya akan menjadikan jaringan mampu mencapai tingkat pengenalan yang lebih tinggi.

Pembelajaran yang sebelumnya dilakukan dengan memperbaharui vektor pewakil per vektor, dimodifikasi agar tejadi pembaharuan per satuan dimensi, dimana setiap dimensinya berbentuk bilangan *fuzzy* segitiga. Oleh karena itu algoritma hasil modifikasi ini akan disebut dengan nama *dimension-based* FNLVQ sedangkan FNLVQ yang sebelumnya dibahas pada Bab 2 akan disebut sebagai *vector-based* FNLVQ.

Pada algoritma *dimension-based* FNLVQ ini, terjadi 3 tahap pengerjaan yang sama seperti pada *vector based* FNLVQ, yaitu pembentukan vektor pewakil, tahap *training,* kemudian tahap *testing*. Perbedaan yang ada adalah pada tahap *training,* karena pengubahan vektor pewakil sesuai dengan data masukan dilakukan per dimensi, bukan lagi per vektor. Struktur jaringan *dimension-based* FNLVQ juga sama seperti *vector-based* FNLVQ pada Bab 2.

3.1.1 Pembentukan Vektor Pewakil

Pembentukan vektor pewakil pada inisialisasi jaringan *dimension-based* FNLVQ sama persis dengan pembentukan vektor pewakil pada *vector-based* FNLVQ. Jumlah data yang digunakan masih 15 data per kelas untuk membentuk vektor pewakil yang berbentuk vektor *fuzzy.*

3.1.2 *Training*

Pada tahap *training*, terdapat perbedaan yang mencolok antara pengubahan vektor pewakil, karena yang mengalami perubahan hanya 1 dimensi yang memiliki nilai similaritas terkecil pada kelas pemenang. Untuk melakukan hal itu, ketika dilakukan penghitungan nilai similaritas, perlu dicatat indeks dimensi yang perlu diubah. Langkah-langkah tahap *training* adalah sebagai berikut.

1. Semua data membentuk suatu *training set* yang akan digunakan untuk melatih. Sebuah data input misalnya $\vec{x} = (\tilde{x}_1, \tilde{x}_2, ...\tilde{x}_n)$, dimana setiap dimensi x_i adalah sebuah bilangan *fuzzy* $\tilde{x}_i =< x_i^{(1)}, x_i^{(2)}, x_i^{(3)} >$.

   Sama seperti pada *vector based* FNLVQ, data berasal dari 1 kamera saja (1 foto) sehingga setiap dimensi hanya memiliki 1 nilai. Dengan demikian *fuzzy* number yang dibentuk memiliki nilai yang sama untuk komponen minimal, rata-rata, serta maksimalnya (menyerupai garis lurus seperti bilangan *crisp*).

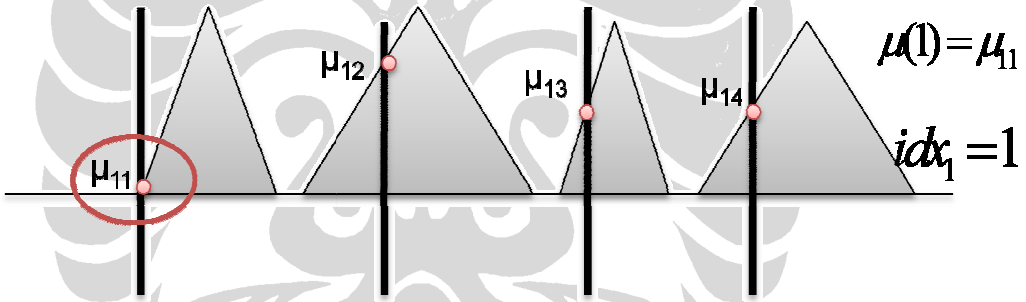
2. Hitung nilai similaritas per dimensi vektor masukan untuk setiap $\tilde{x}_i$ pada vektor input $\vec{x}$, terhadap semua vektor pewakil $\vec{w}_j = (\tilde{w}_{1j}, \tilde{w}_{2j}, ...\tilde{w}_{nj})$ sehingga diperoleh nilai similaritas $M_i = \{\mu_{i1}, \mu_{i2}, ...\mu_{in}\}$ untuk setiap kelas. Penghitungan nilai similaritas ini masih mengikuti kaidah perpotongan yang sama dengan pada persamaan 2.8 atau 2.9, tergantung nilai komponen *fuzzy*.

3. Dari $M_i = \{\mu_{i1}, \mu_{i2}, ...\mu_{in}\}$ setiap kelas (1<i<n), kemudian dicari nilai μ minimal yang ada pada kelas tersebut. Nilai μ tersebut perlu disimpan beserta dengan indeks dimensi yang memiliki nilai μ minimal tersebut, karena hanya dimensi tersebut yang akan mengalami perubahan nantinya. Misalnya saja $\min(M_i)$ untuk kelas ke-i ada pada indeks ke-j, maka perlu disimpan nilai μ_j dan indeks j pada vektor *idx* ($idx_i = j$).

4. Berikutnya, dari semua nilai similaritas minimal dari k buah kelas dicari nilai maksimal yang ada ($\mu_{final} = \max(\mu(1), \mu(2), ...\mu(k))$) sebagai nilai μ akhir. Kelas dengan nilai μ maksimal ini menjadi BMU untuk masukan tersebut.

5. Setelah terjadi klasifikasi, akan dilakukan pengubahan terhadap vektor pewakil sesuai hasilnya (Gambar 3.1). Terdapat 3 kemungkinan yang terjadi, yaitu:

    a. Nilai similaritas terbesar adalah 0. Jika hal ini terjadi, maka data *training* dianggap tidak masuk ke dalam kelas manapun. Pada kasus ini terjadi pelebaran semua dimensi vektor pewakil, namun berbeda dari *vector-based* FNLVQ yang melebarkan semua dimensi pada semua vektor pewakil, *dimension-based* FNLVQ hanya melebarkan vektor pewakil untuk kelas yang sesungguhnya saja. Indeks data masukan diketahui karena jenis pembelajaran yang bersifat *supervised.*



Gambar .3.1 Pemotongan Data Masukan dengan Vektor Pewakil serta Pencatatan Nomor Dimensi

Pelebaran dilakukan mengikuti persamaan 3.1. Persamaan ini sama dengan persamaan 2.14 tetapi hanya berlaku untuk *j*=indeks kelas yang sebenarnya.

$$w_{ij}^{(1)} = w_{ij}^{(2)} - \beta.(w_{ij}^{(2)} - w_{ij}^{(1)})$$

$$w_{ij}^{(2)} = w_{ij}^{(2)}$$

$$w_{ij}^{(3)} = w_{ij}^{(2)} + \beta.(w_{ij}^{(3)} - w_{ij}^{(2)}) \qquad (3.1)$$

b. Jika hasil klasifikasi benar, contohnya ketika data masukan adalah data dari kelas 1, kemudian nilai miu terbesar ada pada kelas 1 sehingga hasil kategorisasi jaringan juga kelas 1. Untuk dimensi indeks idx_k dimana k adalah indeks kelas pemenang, dilakukan pergeseran mendekat dan pelebaran bilangan *fuzzy* pada dimensi tersebut. Jika dimensi tersebut berupa bilangan *fuzzy* $\tilde{w}_{ik} =< w_{ik}^{(1)}, w_{ik}^{(2)}, w_{ik}^{(3)} >$, maka nilai tengah $w_{ik}^{(2)}$ didekatkan kepada data masukan seusai persamaan 2.15 yang sudah dijabarkan pada Bab 2, kemudian nilai $w_{ik}^{(1)}$ dan $w_{ik}^{(3)}$ mengikuti.

   Selanjutnya w_{ik} dilebarkan sama seperti pada persamaan 3.1, dengan β lebih besar dari 1. Ini dilakukan hanya pada dimensi indeks ke idx_k (i= idx_k) dari vektor pewakil kelas ke k (j=k).

c. Jika hasil klasifikasi salah, contohnya ketika data masukan adalah data dari kelas 1, tetapi kemudian nilai miu terbesar ada pada kelas 3 sehingga hasil kategorisasi jaringan adalah kelas 3. Untuk kasus ini dilakukan 2 (dua) buah perubahan, yaitu pada vektor pewakil kelas hasil klasifikasi yang salah serta pada vektor pewakil kelas yang seharusnya.

   Untuk dimensi indeks idx_k dimana k adalah indeks kelas pemenang, dilakukan penggeseran menjauh dan penyempitan bilangan *fuzzy* pada dimensi tersebut. Untuk dimensi $\tilde{w}_{ik}$ tersebut, nilai tengah $w_{ik}^{(2)}$ dijauhkan dari data masukan seusai persamaan 2.16 pada Bab 2, kemudian nilai $w_{ik}^{(1)}$ dan $w_{ik}^{(3)}$ mengikuti.

   Selanjutnya w_{ik} disempitkan menggunakan persamaan 3.1, tetapi dengan β lebih kecil dari 1 (satu). Ini dilakukan hanya pada dimensi indeks ke idx_k (i= idx_k) dari vektor pewakil kelas ke k (j=k).

Untuk dimensi indeks idx_l dimana l adalah indeks kelas yang seharusnya (dalam contoh adalah kelas 1) , dilakukan pelebaran pada dimensi tersebut, sama seperti pada poin b.

Ketika semua data pada *training set* sudah dimasukkan ke dalam jaringan dan dilakukan pembelajaran, maka dikatakan telah dilakukan pembelajaran 1 *epoch.* Pembelajaran dilakukan secara berulang dengan laju pembelajaran (α) yang semakin diperkecil. Ketika nilai α sudah memenuhi syarat henti (ditentukan nilai α minimal sebagai batas pembelajaran), maka fase *training* sudah selesai.

Sama seperti eksperimen yang dilakukan pada *vector-based* FNLVQ, data *training* yang menyerupai data *crisp* karena nilai komponen *fuzzy* yang sama dapat dijadikan bilangan *fuzzy* melalui fuzzifikasi yang serupa dengan pada *vector-based* FNLVQ.

3.1.2 *Testing*

Pada tahap *testing* tidak diperlukan lagi penyimpanan idx_j karena tidak lagi dilakukan pengubahan vektor pewakil, hanya perlu dilakukan komputasi nilai similaritas untuk klasifikasi. Sama seperti pada tahap *training*, data *testing* juga berasal dari 1 buah citra saja, sehingga nilai setiap dimensi input hanya terdiri dari 1 nilai. Untuk dijadikan masukan pada jaringan, bisa berupa nilai *fuzzy* dengan nilai ketiga komponen yang sama (menyerupai bilangan *crisp*), atau dapat dilakukan fuzzifikasi yang sama seperti fuzzifikasi data *testing* untuk *vector based* FNLVQ.

Data masukan pada tahap *testing* dimasukkan ke dalam jaringan kemudian diproses melalui tahap-tahap yang sama seperti pada tahap *training,* namun cukup sampai mendapatkan BMU sebagai hasil klasifikasi data masukan. Melalui pembelajaran per dimensi seperti ini, diharapkan vektor pewakil yang terbentuk setelah proses *training* lebih tepat dalam mewakili vektor-vektor pada kelas tersebut. Eksperimen yang dilakukan akan dijabarkan pada Bab 4.

3.2 Side and Dimension Based FNLVQ

Dalam rangka mendapatkan tingkat pengenalan yang lebih tinggi lagi, dilakukan eksplorasi lanjutan pada algoritma *dimension-based* FNLVQ pada subbab 3.1. Pada proses pembaharuan vektor pewakil saat *training*, terdapat proses pelebaran dan penyempitan bilangan *fuzzy* pada dimensi yang mengalami pembelajaran. Algoritma *side-and-dimension based* FNLVQ ini pada dasarnya adalah algoritma dengan pembaharuan vektor pewakil berbasiskan dimensi, sama seperti *dimension-based* FNLVQ, namun terdapat modifikasi pada proses pelebaran atau penyempitan dimensi vektor pewakil.

3.3.1 Pembentukan Vektor Pewakil

Pembentukan vektor pewakil pada algoritma ini tidak ada perbedaan dari pembentukan vektor pewakil pada algoritma *vector-based* FNLVQ maupun *dimension-based* FNLVQ. Data yang diambil adalah 15 buah vektor per kelas yang kemudian dibentuk menjadi vektor pewakil masing-masing kelas. 15 nilai untuk masing-masing dimensi menjadi sebuah bilangan *fuzzy.*

3.3.2 *Training*

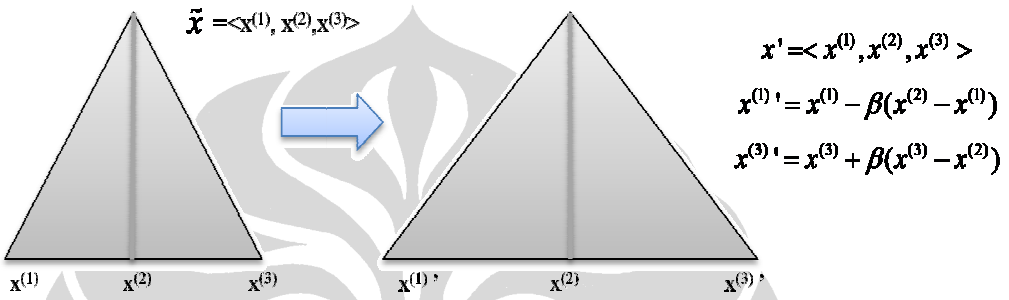
Pada tahap *training* algoritma *vector-based* maupun *dimension-based* FNLVQ, pelebaran dimensi-dimensi (atau dimensi saja) pada vektor pewakil dilakukan sesuai persamaan 2.14 atau pada persamaan 3.1. Pada persamaan tersebut, pengubahan kelebaran dilakukan baik untuk sisi kiri bilangan *fuzzy* maupun sisi kanan (Gambar 3.2).

Pada algoritma *side-and-dimension based* FNLVQ, pelebaran dan penyempitan tidak dilakukan terhadap kedua sisi bilangan *fuzzy,* melainkan hanya salah satu saja, tergantung letak perpotongan dengan data *training*. Oleh karena itu, pada tahap *training*, perlu disimpan informasi lain selain nilai similaritas dan indeks dimensi, yaitu informasi letak perpotongan. Langkah-langkah tahap *training* adalah sebagai berikut.

1. Semua data membentuk suatu *training set* yang akan digunakan untuk melatih. Sebuah data masukan misalnya $\vec{x} = (\tilde{x}_1, \tilde{x}_2, ...\tilde{x}_n)$, dimana setiap dimensi x_i adalah sebuah bilangan *fuzzy* $\tilde{x}_i =< x_i^{(1)}, x_i^{(2)}, x_i^{(3)} >$. Sama seperti *vector-based*

dan *dimension-based* FNLVQ, data berasal dari 1 kamera saja (1 foto) sehingga setiap dimensi hanya memiliki 1 nilai. Dengan demikian *fuzzy* number(menyerupai garis lurus seperti bilangan *crisp*.

2. Hitung nilai similaritas per dimensi vektor masukan untuk setiap x_i pada vektor input *x,* terhadap semua vektor pewakil $w_j = (w_{1j}, w_{2j}, ...w_{nj})$ sehingga didapatkan nilai similaritas $M_i = \{\mu_{i1}, \mu_{i2}, ...\mu_{in}\}$ untuk setiap kelas.

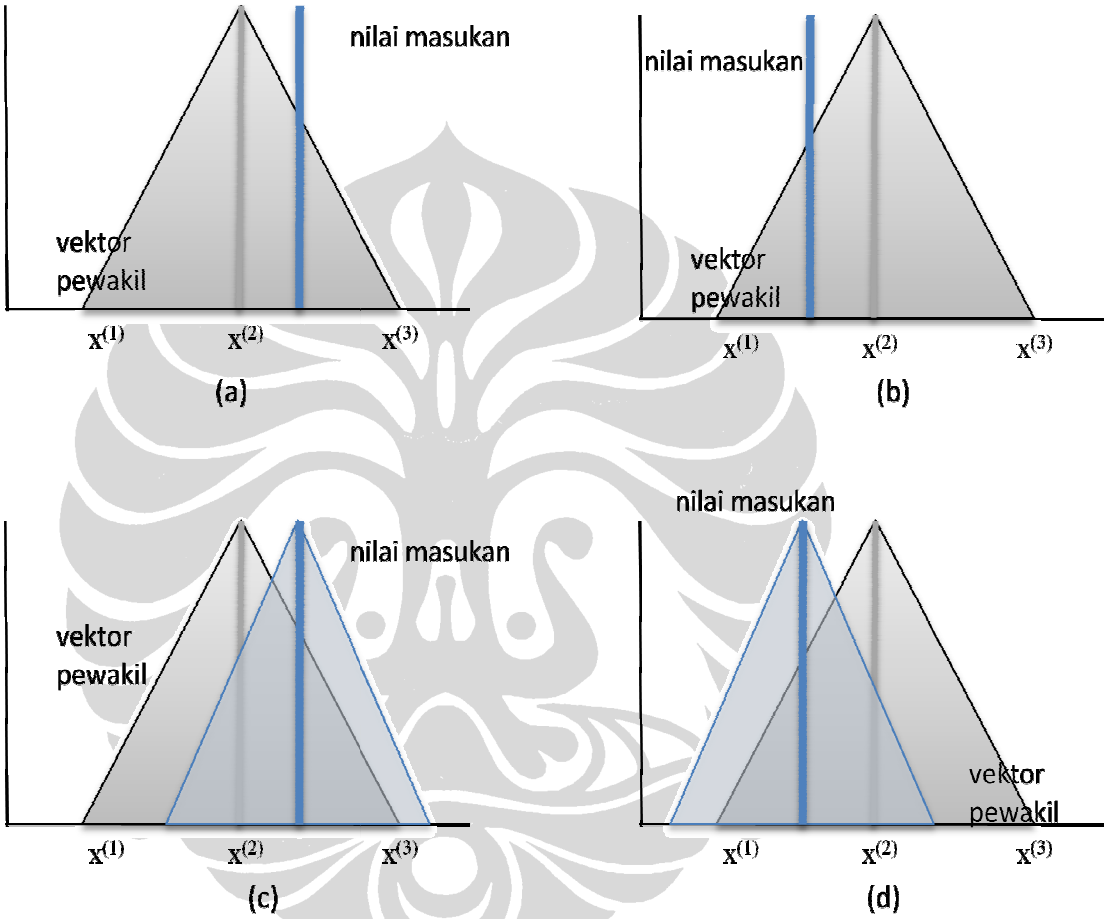


Gambar 3.2 Pelebaran Dimensi Vektor Pewakil

3. Dari $M_i = \{\mu_{i1}, \mu_{i2}, ...\mu_{in}\}$ setiap kelas (1<i<n), kemudian dicari nilai μ minimal yang ada pada kelas tersebut. Nilai μ tersebut perlu disimpan beserta dengan indeks dimensi yang memiliki nilai μ minimal tersebut, karena hanya dimensi tersebut yang akan mengalami perubahan nantinya. Misalnya saja min(M_i) untuk kelas ke-i ada pada indeks ke-j, maka perlu disimpan nilai μ_j dan indeks j pada vektor *idx* ($idx_i = j$). Untuk dimensi tersebut, perlu dicari letak perpotongan data masukan terhadap vektor pewakil.

   Perpotongan data masukan dengan vektor pewakil dapat diperiksa dari nilai yang dimiliki bilangan masukan. Jika bilangan masukan berbentuk *fuzzy* dengan ketiga komponen bernilai sama, maka nilai yang diperiksa adalah 1 nilai tersebut, sedangkan jika bilangan masukan berbentuk *fuzzy* hasil fuzzifikasi data, maka nilai yang diperiksa adalah nilai komponen rata-rata (tengah) dari bilangan *fuzzy* tersebut.

Dari nilai data masukan yang dibandingkan dengan vektor pewakil, jika nilai bilangan masukan lebih kecil daripada nilai rata-rata komponen *fuzzy* pada vektor pewakil, maka letak perpotongan dikatakan di sebelah kiri. Sebaliknya, jika nilai bilangan masukan lebih besar, maka letak perpotongan dikatakan di sebelah kanan (Gambar 3.3).



Gambar 3.3 Penentuan Sisi Pemotongan Data Masukan Terhadap Vektor Pewakil. (a) dan (c) Pemotongan di Sisi Kanan. (b) dan (d) Pemotongan di Sisi Kiri

4. Berikutnya, dari semua nilai similaritas minimal dari k buah kelas dicari nilai maksimal yang ada ($\mu_{final} = \max(\mu(1), \mu(2), ...\mu(k))$) sebagai nilai μ akhir. Kelas dengan nilai μ maksimal ini menjadi *best matching unit* untuk masukan tersebut.

Pada saat dilakukan pelebaran atau penyempitan vektor pewakil, pelebaran hanya dilakukan pada 1 sisi saja dari bilangan *fuzzy* sesuai letak perpotongan dengan data masukan (Gambar 3.4). Pelebaran atau penyempitan bilangan *fuzzy* pada *training* kemudian berubah, sesuai persamaan 3.2.

Pelebaran yang terjadi akan membentuk dimensi-dimensi vektor pewakil yang lebih tepat, karena pembaharuan vektor pewakil dengan perhitungan yang lebih presisi. Metode ini terutama diharapkan akan meningkatkan pengenalan *outlier*, karena pelebaran dimensi yang dibatasi dengan lebih ketat hanya akan mengambil nilai yang benar-benar sesuai untuk diwakilkan.

$$w_{ij}^{(2)} = w_{ij}^{(2)}$$

jika letak perpotongan di sebelah kiri:

$$w_{ij}^{(1)} = w_{ij}^{(2)} - \beta.(w_{ij}^{(2)} - w_{ij}^{(1)})$$

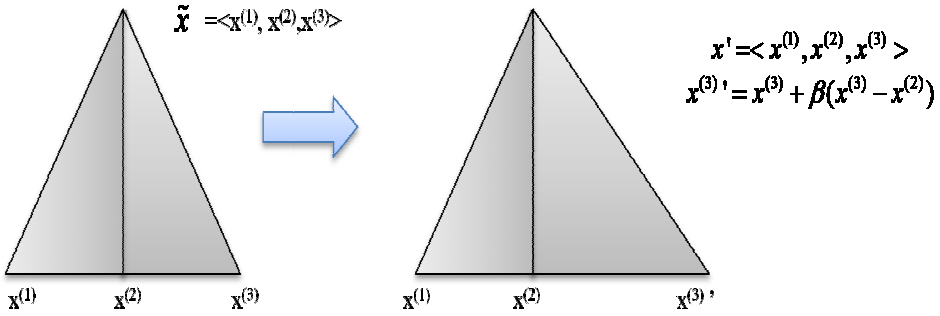
jika letak perpotongan di sebelah kanan:

$$w_{ij}^{(3)} = w_{ij}^{(2)} + \beta.(w_{ij}^{(3)} - w_{ij}^{(2)}) \qquad (3.2)$$

Sama seperti eksperimen yang dilakukan pada *vector-based* dan *dimension-based* FNLVQ, data *training* yang menyerupai data *crisp* karena nilai komponen *fuzzy* yang sama dapat dijadikan bilangan *fuzzy* melalui fuzzifikasi yang serupa dengan pada *vector-based* FNLVQ.

3.3.3 *Testing*

Pada tahap *testing* tidak lagi diperlukan penyimpanan nilai *idx* untuk menyimpan indeks dimensi yang akan diperbaharui maupun informasi lokasi perpotongan data terhadap vektor pewakil. Sama seperti pada algoritma *vector-based* dan *dimension-based,* tahap *testing* hanya terdiri dari proses klasifikasi data masukan. Klasifikasi ini dilakukan melalui komputasi nilai similaritas seperti pada tahap *training,* kemudian menentukan BMU yang sesuai.



Gambar .3.4 Pelebaran Dimensi Vektor Pewakil

Tujuan dilakukan pembelajaran seperti ini adalah untuk mendapatkan vektor pewakil yang lebih tepat lagi dalam mewakili vektor-vektor pada masing-masing kelas. Dengan demikian tingkat pengenalan jaringan juga bisa meningkat.

3.3 Eksperimen Terhadap Citra Asli

Untuk eksperimen terhadap data murni, rancangan eksperimen yang dilakukan masih sama seperti sebelumnya, dengan variabel-varibel berikut ini: (1) Penerapan fuzzifikasi (2) laju pembelajaran (α), serta (3) perbandingan data *training* dan data *testing*. Untuk penerapan fuzzifikasi pada data, dilakukan 3 skenario, yaitu (1) tanpa fuzzifikasi pada data *training* dan *testing*, (2) fuzzifikasi hanya pada data *training*, dan (3) fuzzifikasi pada data *training* dan *testing*. Eksperimen ini dilakukan dengan algoritma *side-and-dimension-based* FNLVQ.

Eksperimen dilakukan dengan melatih 10 kelas data citra wajah orang, kemudian menguji dengan 12 kelas data citra wajah orang. Karena yang dilatihkan ada 10 kelas data, maka pada lapisan keluaran akan ada 10 neuron dan nilai k pada eksperimen ini adalah 10. Hasil eksperimen berupa perhitungan 3 jenis tingkat pengenalan: (1) tingkat identifikasi (2) tingkat klasifikasi. Hasil yang diperoleh masih ingin dikembangkan lebih lanjut menjadi *side-and-dimension-based* FNLVQ. Hasil eksperimen pengenalan citra dengan menggunakan algoritma *side-and-dimension-based* FNLVQ adalah seperti pada Tabel 3.1 untuk tingkat identifikasi dan Tabel 3.2 untuk tingkat klasifikasi.

Tabel 3.1 Tingkat Identifikasi *Side-and-Dimension-Based* FNLVQ Terhadap Citra Asli

VARIABEL				
Training*: *Testing	**α**	***Crisp***	**Data *Testing Fuzzy***	**Data *Training* dan Data *Testing Fuzzy***
5050	0.1	68%	68%	52%
	0.2	68%	68%	52%
	0.3	69%	69%	50%
	0.4	69%	68%	61%
	0.5	68%	68%	67%
7030	0.1	82%	9%	10%
	0.2	82%	9%	10%
	0.3	82%	10%	10%
	0.4	72%	10%	10%
	0.5	59%	59%	76%

Tingkat identifikasi yang dapat dicapai jika menggunakan data *crisp* mampu mencapai 82% pada keadaan paling optimal. Tingkat pengenalan tertinggi ini tercapai ketika data yang digunakan berupa data *crisp,* perbandingan data *training* dan *testing* 50%:50% dan laju pembelajaran 0.5. Tingkat pengenalan pada skenario yang lain dengan data *crisp* cenderung stabil, berkisar antara 59%-82%.

Dengan data *testing fuzzy,* tingkat identifikasi maksimal yang dicapai adalah sebesar 69%. Pada skenario lainnya, tingkat identifikasi tetap pada angka 68%-69% ketika perbandingan data *training:testing* sebesar 50%:50%, namun ketika perbandingan data *training:testing* sebesar 70%:30% tingkat identfikasi menurun sampai kisaran 9%-10%, dengan pengecualian ketika laju pembelajaran 0.5, yang mencapai tingkat identifikasi sebesar 59%.

Ketika digunakan data *training* dan *data testing,* tingkat identifikasi maksimal adalah sebesar 67%, yaitu ketika perbandingan data *training:testing* sebesar 50%:50% dan laju pembelajaran 0.5. Pada skenario yang lainnya, tingkat identfikasi yang dicapai

cenderung menurun dan lebih buruk daripada ketika digunakan data *crisp* atau data *testing* saja yang berupa bilangan *fuzzy.*

Dari Tabel 3.2 dapat dilihat tingkat klasifikasi terhadap citra asli dengan menggunakan *side-and-dimension-based* FNLVQ. Tingkat klasifikasi pada umumnya stabil dan tinggi untuk skenario manapun. Untuk data *crisp,* tingkat klasifikasi data teregistrasi sangat tinggi, yaitu pada kisaran 80%-91%, serta untuk data tidak teregistrasi berkisar antara 69%-82%. Jika dilihat dari nilai rata-rata antara keduanya, terdapat rata-rata tertinggi sebesar 63%, yaitu ketika perbandingan data *training:testing* sebesar 50%:50% dan laju pembelajaran 0.5. Pada skenario itu tingkat klasifikasi data teregistrasi ebesar 80% dan data tidak teregistrasi sebesar 85%.

Untuk data *testing fuzzy,* tingkat klasifikasi data teregistrasi mencapai 100% pada keadaan tertentu, tetapi pada saat itu pula tingkat klasifikasi data tidak teregistrasi 0%. Dengan kata lain, pada skenario tersebut jaringan mengenali data manapun. Tentunya hal ini tidak diinginkan pada suatu sistem. Tingkat klasifiksi paling seimbang jika dilihat dari rata-rata antara tingkat klasifikasi data teregistrasi dan tak teregistrasi adalah sebesar 81% ketika perbandingan data *training:testing* sebesar 50%:50% dan laju pembelajaran 0.1-0.3. Pada skenario tersebut tingkat klasifikasi data teregistrasi sebesar 84% dan data tidak teregistrasi sebesar 78%.

Untuk data *training* dan *testing,* keadaan yang sama terjadi dengan tingkat klasifikasi sebesar 100%. Dari rata-rata antara data teregistrasi dan data tidak teregistrasi, rata-rata tertinggi adalah sebesar 86%, saat tingkat klasifikasi data teregistrasi sebesar 80% dan data tidak teregistrasi sebesar 91%.

3.4 Eksperimen Terhadap Citra *Noisy*

Eksperimen ini dilakukan dengan algoritma *side-and-dimension-based* FNLVQ terhadap citra yang tidak ideal. Rancangan eksperimen masih sama seperti eksperimen

Tabel 3.2 Tingkat Klasifikasi Side-and-Dimension-Based FNLVQ Terhadap Citra Asli

VARIABEL		*Crisp*			Data *Testing Fuzzy*			Data *Training* dan Data *Testing Fuzzy*		
Training*: *Testing	**α**	**Data Teregistrasi**	**Data Tak Teregistrasi**	**Rata-Rata**	**Data Teregistrasi**	**Data Tak Teregistrasi**	**Rata-Rata**	**Data Teregistrasi**	**Data Tak Teregistrasi**	**Rata-Rata**
5050	0.1	80%	82%	81%	84%	78%	81%	84%	77%	81%
	0.2	80%	82%	81%	84%	78%	81%	80%	90%	85%
	0.3	80%	83%	82%	84%	78%	81%	74%	81%	78%
	0.4	80%	83%	82%	82%	78%	80%	80%	80%	80%
	0.5	80%	85%	83%	80%	85%	83%	80%	91%	86%
7030	0.1	90%	69%	80%	9%	99%	54%	100%	0%	50%
	0.2	90%	69%	80%	9%	99%	54%	100%	0%	50%
	0.3	90%	69%	80%	100%	0%	50%	100%	0%	50%
	0.4	90%	72%	81%	100%	0%	50%	100%	0%	50%
	0.5	91%	74%	83%	91%	74%	83%	90%	61%	81%

yang menggunakan *noise* pada subbab 2.6. *Noise* ditambahkan terhadap citra yang ideal tersebut dengan menggunakan *mathematical tools* MATLAB. *Noise* yang digunakan adalah (1) *gaussian blur* 10%, (2) *poisson,* dan (3) *salt & pepper*. Citra yang didapatkan kemudian adalah citra yang terdistorsi (Gambar 2.12).

Pada skenario eksperimen ini, *training* jaringan masih dilakukan dengan data biasa. Pada tahap *testing,* jaringan diuji dengan menggunakan data yang dimanipulasi agar mengandung *noise*. Dengan demikian, dapat diuji kemampuan jaringan untuk mengenali data *noise* dari pembelajaran data biasa. Variabel eksperimen lain yang digunakan adalah (1) perbandingan data *training: testing* dan (2) laju pembelajaran (α).Untuk variabel penerapan fuzzifikasi pada data, untuk eksperimen ini tidak digunakan, dan data yang digunakan adalah data tanpa fuzzifikasi.

Eksperimen ini menggunakan algoritma *side-and dimension-based* FNLVQ. Diharapkan ada suatu peningkatan pada pengenalan dibandingkan *vector-based* FNLVQ, baik itu pengenalan data teregistrasi maupun data tidak teregistrasi. Seperti pada eksperimen dengan data tanpa *noise,* diharapkan jaringan yang mampu mengenali data teregistrasi dan data tidak teregistrasi secara seimbang.

Tngkat identfikasi yang dicapai dijabarkan dalam Tabel 3.3. Dapat dilihat pada tingkat pengenalan cenderung rendah untuk citra dengan *gaussian noise* dan *sat&pepper noise.* Untuk data dengan *gaussian noise,* jaringan hanya mampu mengidentfikasi paling tinggi sebesar 29%, dan untuk data dengan *salt&pepper noise* tingkat identfikasi lebih rendah lagi, yaitu berkisar antara 8%-14% saja.

Sementara itu, tingkat klasifikasi citra *noisy* dapat dilihat pada Tabel 3.4. Tingkat klasifikasi yang dicapai oleh *side-and-dimension-based* FNLVQ terhadap citra *noisy* masih mengikuti pola yang sama seperti tingkat identfikasinya, yaitu tingkat klasifikasi yang cukup rendah untuk citra dengan *gaussian noise* dan *salt&pepper noise,* serta tingkat klasifikasi yang masih cukup besar untuk klasifikasi citra yang mengandung *poisson noise.*

Tabel 3.3 Tingkat Identfikasi *Side-and-Dimension-Based* FNLVQ Terhadap Citra yang Mengandung No*ise*

VARIABEL		GAUSSIAN NOISE	POISSON NOISE	SALT & PEPPER NOISE
Training*: *Testing	**α**			
5050	0.1	29%	72%	8%
	0.2	28%	72%	8%
	0.3	28%	72%	8%
	0.4	22%	66%	6%
	0.5	24%	59%	6%
7030	0.1	25%	66%	14%
	0.2	25%	66%	14%
	0.3	24%	66%	14%
	0.4	22%	56%	13%
	0.5	21%	54%	13%

Untuk citra dengan *gaussian noise,* tingkat klasifikasi paling seimbang jika dilihat dari rata-rata keduanya adalah pada saat ketika perbandingan data *training:testing* sebesar 50%:50% dan laju pembelajaran 0.5 dengan rata-rata 75%. Saat itu tingkat klasifikasi data teregistrasi sebesar 75% dan data tidak teregistrasi 74%.

Untuk citra dengan *poisson noise,* klasifikasi rata-rata tertinggi adalah sebesar 83% pada skenario eksperimen yang sama. Pada saat tersebut tingkat klasifikasi data teregistrasi sebesar 92% dan data tidak teregistrasi sebesar 74%. Untuk citra dengan *salt&pepper noise* tingkat klasifikasi yang dicapai paling rendah. Rata-rata antara klasifikasi data teregistrasi dan tidak teregistrasi tertinggi hanya sebesar 20%. Hal ini terjadi pada skenario eksperimen yang masih sama, ketika tingkat klasifikasi data teregistrasi hanya sebesar 6% dan data tidak teregistrasi 33%. Tingkat klasifikasi data teregitrasi tertinggi yang bisa dicapai hanya sebesar 14%.

Tabel 3.4 Tingkat Klasifikasi Side-and-Dimension-Based FNLVQ Terhadap Citra yang Mengandung Noise

VARIABEL		GAUSSIAN NOISE			POISSON NOISE			SALT & PEPPER NOISE		
Training*: *Testing	**α**	**Data Teregistrasi**	**Data Tak Teregistrasi**	**Rata-Rata**	**Data Teregistrasi**	**Data Tak Teregistrasi**	**Rata-Rata**	**Data Teregistrasi**	**Data Tak Teregistrasi**	**Rata-Rata**
5050	0.1	53%	69%	61%	85%	69%	77%	8%	13%	11%
	0.2	53%	69%	61%	85%	69%	77%	8%	13%	11%
	0.3	52%	69%	61%	85%	69%	77%	8%	13%	11%
	0.4	68%	72%	70%	87%	72%	80%	6%	28%	17%
	0.5	75%	74%	75%	92%	74%	83%	6%	33%	20%
7030	0.1	27%	9%	18%	73%	9%	41%	14%	19%	17%
	0.2	27%	9%	18%	73%	9%	41%	14%	19%	17%
	0.3	26%	9%	18%	73%	9%	41%	14%	19%	17%
	0.4	32%	56%	44%	76%	56%	66%	13%	33%	23%
	0.5	33%	61%	47%	77%	61%	69%	13%	35%	24%

Secara umum, tingkat pengenalan data teregistrasi sesuai kelasnya pada *side-and-dimension-based* FNLVQ tidak terlalu tinggi, terutama pada citra dengan *gaussian* dan *salt&pepper noise.* Tingkat pengenalan data teregistrasi tidak terlalu buruk pada citra dengan *poisson noise.* Tetapi dalam hal keseimbangan antara pengenalan data teregistrasi dan data tidak teregistrasi, tingkat pengenalan yang mampu dicapai cenderung seimbang dan lebih tinggi daripada *vector-based* FNLVQ.

BAB 4 KOMPARASI ALGORITMA DAN ANALISIS

Pada bab ini akan dijabarkan analisis terhadap hasil eksperimen yang telah dijabarkan pada Bab 3. Analisis yang dilakukan adalah perbandingan performa antar berbagai skenario eksperimen yang sudah dilakukan agar bisa didapatkan skenario dengan tingkat pengenalan yang paling tinggi.

4.1 Perbandingan Tingkat Pengenalan Citra Wajah Murni

Modifikasi yang dilakukan terhadap algoritma *vector-based* FNLVQ diharapkan akan mampu meningkatkan kemampuan jaringan untuk mengenali data, baik itu data teregistrasi maupun data tidak teregistrasi. Untuk skenario eksperimen pertama, citra yang digunakan adalah citra wajah murni yang tidak mengandung *noise.*

Jaringan FNLVQ diharapkan mampu mengenal data teregistrasi maupun data tidak teregistrasi. Dari hasil eksperimen, cenderung terjadi *trade-off* antara tingkat pengenalan data teregistrasi dan tingkat pengenalan data tidak teregistrasi. Ketika jaringan mampu mencapai tingkat pengenalan yang tinggi terhadap data teregistrasi, pengenalan terhadap data tidak teregistrasi akan mengalami penurunan, dan begitu pula sebaliknya. Hal ini dapat dilihat dari keadaan tingkat pengenalan optimal yang mampu dicapai jaringan FNLVQ.

Tingkat identifikasi kedua algoritma jika dibandingkan adalah seperti pada tabel 4.1. Tingkat pengenalan tertinggi pada *vector-based* FNLVQ hanya mencapai 33%, dan pada *side-and-dimension-based* FNLVQ mampu mencapai 82%. Peningkatan yang dapat dicapai adalah sebesar 49%. Sedangkan perbandingan tingkat klasifikasi kedua algoritma adalah seperti pada Tabel 4.2.

Jika tujuan yang dituju adalah mencari tingkat klasifikasi data teregistrasi tertinggi, maka peningkatan yang bisa dicapai adalah sebesar 9% untuk data teregistrasi dan 63% untuk data tidak teregistrasi. Jika tujuannya adalah mencari tingkat klasifikasi data yang paling seimbang, maka peningkatan yang bisa dicapai adalah sebesar 19% untuk data teregistrasi dan 63% untuk data tidak teregistrasi.

Tabel 4.1 Peningkatan Tingkat Identifikasi Side-and-Dimension-Based FNLVQ dari Vector-Based FNLVQ pada Skenario Optimal

	Skenario	**Tingkat Pengenalan**
***Vector-Based* FNLVQ**	Tanpa Fuzzifikasi, 70%:30%, α 0.5	
	Fuzzifikasi *Testing*,50%:50%, α 0.5	
	Fuzzifikasi *Testing*,70%:30%, α 0.5	
	Fuzzifikasi *Training* dan *Testing*, 70%:30%, α 0.5	33%
Side-and-Dimension-Based FNLVQ	Tanpa Fuzzifikasi, 70%:30%, α 0.1	82%
Peningkatan Pengenalan		49%

Tingkat klasifikasi ketika tujuan adalah mencari tingkat klasifikasi data teregistrasi tertinggi adalah sebesar 82% untuk data teregistrasi dan 11% untuk data tidak teregistrasi pada *vector-based* FNLVQ dan 91% untuk data teregistrasi dan 74% untuk data tidak teregistrasi pada *dimension-based* FNLVQ.

Sementara itu jika tujuan adalah mencari tingkat klasifikasi data teregistrasi dan tingkat data tidak teregistrasi yang paling seimbang adalah sebesar 61% untuk data teregistrasi dan 22% untuk data tidak teregistrasi pada *vector-based* FNLVQ dan 80% untuk data teregistrasi dan 85% untuk data tidak teregistrasi pada *dimension-based* FNLVQ.

Untuk peningkatan tingkat pengenalan secara umum pada semua skenario eksperimen, dapat dilihat pada Tabel 4.3. Pada umumnya, tingkat pengenalan *side-and-dimension-based* FNLVQ lebih tinggi daripada *vector-based* FNLVQ pada semua skenario eksperimen.

Tabel 4.2 Peningkatan Tingkat Klasifikasi Side-and-Dimension-Based FNLVQ dari Vector-Based FNLVQ pada Skenario Optimal

Tujuan	Algoritma	Skenario	Data Teregistrasi	Data Tak Teregistrasi
Mencari Pengenalan Data Teregistrasi Tertinggi	*Vector-Based* FNLVQ	Fuzzifikasi *Testing*, 50%:50%, α 0.3	82%	11%
	Side-and-Dimension-Based FNLVQ	Tanpa Fuzzifikasi, 70%:30%, α 0.1	91%	74%
	Peningkatan Pengenalan		9%	63%
Mencari Keseimbangan Pengenalan Data Teregistasi dan Data Tak Teregistrasi	*Vector-Based* FNLVQ	Tanpa Fuzzifikasi, 50%:50%, α 0.5	61%	22%
	Side-and-Dimension-Based FNLVQ	Tanpa Fuzzifikasi, 50%:50%, α 0.5	80%	85%
	Peningkatan Pengenalan		19%	63%

Tabel 4.3 Perbandingan Tingkat Identfikasi dan Tingkat Klasifikasi *Vector-Based* FNLVQ dan *Side-and-Dimension-Based* FNLVQ

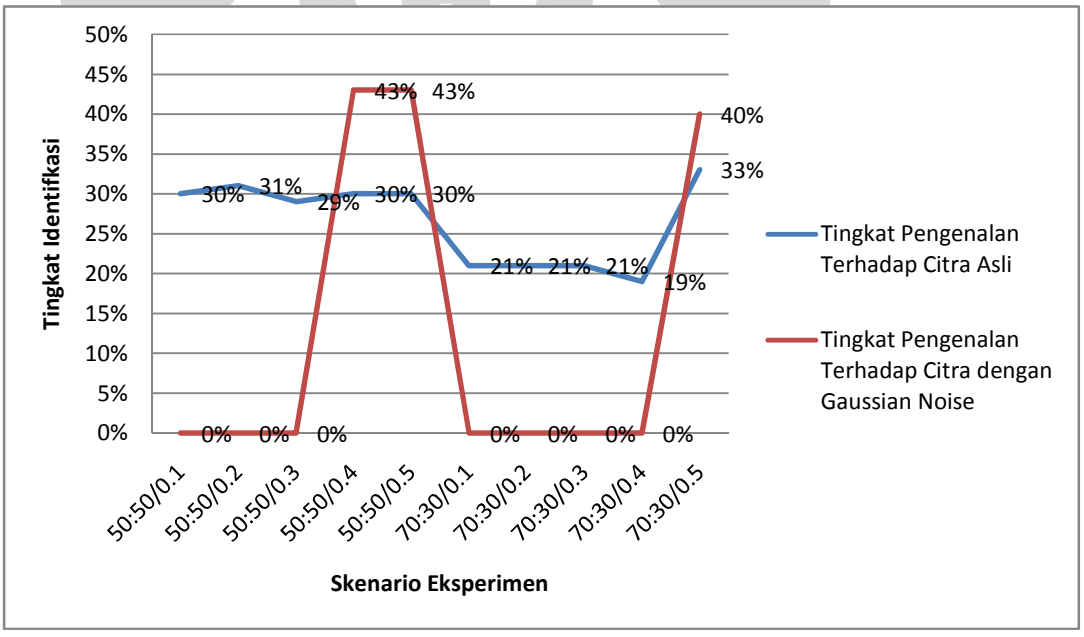
VARIABEL		TANPA FUZZIFIKASI						DENGAN FUZZIFIKASI PADA DATA *TESTING*						DENGAN FUZZIFIKASI DATA *TRAINING* DAN *TESTING*					
Training:Testing		*Vector-Based* FNLVQ		*Side-and-Dimension-Based* FNLVQ		Peningkatan		*Vector-Based* FNLVQ		*Side-and-Dimension-Based* FNLVQ		Peningkatan		*Vector-Based* FNLVQ		*Side-and-Dimension-Based* FNLVQ		Peningkatan	
	α	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*	*Identifikasi*	*Ratta-rata Klasifikasi*
5050	0.1	30%	50%	68%	81%	38%	31%	30%	50%	68%	81%	38%	31%	10%	50%	52%	81%	42%	31%
	0.2	31%	50%	68%	81%	37%	31%	32%	50%	68%	81%	36%	31%	12%	50%	52%	85%	40%	35%
	0.3	29%	50%	69%	82%	40%	32%	31%	50%	69%	81%	38%	31%	15%	50%	50%	78%	35%	28%
	0.4	30%	38%	69%	82%	39%	44%	13%	56%	68%	80%	55%	24%	12%	39%	61%	80%	49%	41%
	0.5	30%	38%	68%	83%	38%	45%	33%	56%	68%	83%	35%	27%	30%	39%	67%	86%	37%	47%
7030	0.1	21%	50%	82%	80%	61%	30%	30%	50%	9%	54%	-21%	4%	18%	50%	10%	50%	-8%	0%
	0.2	21%	50%	82%	80%	61%	30%	21%	50%	9%	54%	-12%	4%	2%	50%	10%	50%	8%	0%
	0.3	21%	50%	82%	80%	61%	30%	24%	50%	10%	50%	-14%	0%	0%	50%	10%	50%	10%	0%
	0.4	19%	50%	72%	81%	53%	31%	13%	50%	10%	50%	-3%	0%	8%	50%	10%	50%	2%	0%
	0.5	33%	31%	59%	83%	26%	52%	33%	48%	59%	83%	26%	35%	31%	36%	76%	81%	45%	45%
Rata-Rata Peningkatan						45%	36%	**Rata-Rata Peningkatan**				18%	19%	**Rata-Rata Peningkatan**				26%	23%

4.2 Analisis Penurunan Tingkat Pengenalan terhadap Citra *Noisy*

Pada subbab sebelumnya dibahas mengenai analisis terhadap tingkat pengenalan pada citra wajah yang ideal dan tidak mengandung *noise*. Akan tetapi, suatu sistem pengenalan wajah yang optimal diharapkan mampu mengenali citra ketika citra tersebut tidak berada dalam kondisi optimal. Seperti yang sudah dijelaskan sebelumnya, pada eksperimen yang menggunakan *noise,* tidak diterapkan fuzzifikasi pada data. Pada subbab ini akan dibahas mengenai penurunan tingkat pengenalan citra *noisy* dibandingkan dengan pengenalan citra asli.

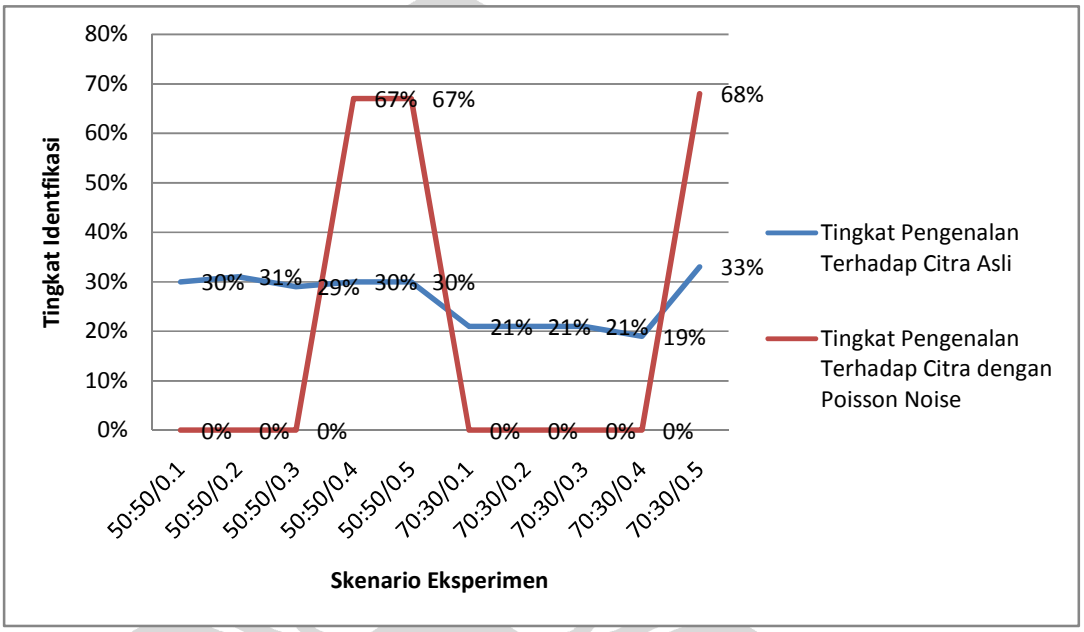
4.2.1 Tingkat Identifikasi terhadap Citra *Noisy*

Pada penerapan algoritma pengenalan *vector-based* FNLVQ, perbandingan tingkat identifikasi jaringan terhadap citra wajah murni dan citra wajah dengan *gaussian noise* adalah seperti pada Gambar 4.1. Tingkat identifikasi terhadap citra dengan *gaussian noise* menurun dengan rata-rata 14% serta maksimum mencapai 31%. Disini, tingkat



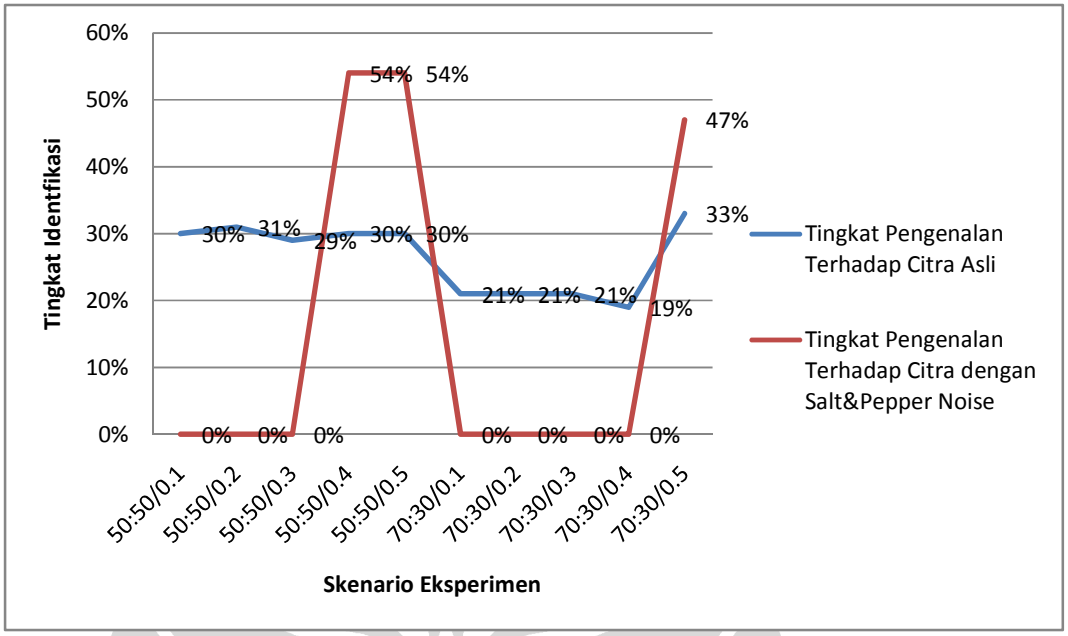
Gambar 4.1 Perbandingan Tingkat Identifkasi *Vector-Based* FNLVQ Terhadap Citra Asli dan Citra dengan *Gaussian Noise*

identifikasi terhadap data teregistrasi maksimal yang mampu dicapai oleh jaringan *vector-based* FNLVQ ini adalah sebesar 43%. Tingkat identfikasi ini bahkan lebih tinggi dibandingkan tingkat pengenalan yang mampu dicapai terhadap data yang murni.Tren perubahan tingkat identfikasi *vector-based* FNLVQ pada Gambar 4.1 memperlihatkan tingkat identfikasi terhadap data murni secara rata-rata lebih tinggi daripada tingkat identfikasi data dengan *noise.* Walau demikian, tingkat identfikasi tertinggi yang dicapai dalam identfikasi data dengan *noise* bisa lebih tinggi daripada pada identfikasi data murni.



Gambar 4.2 Perbandingan Tingkat Identifikasi *Vector-Based* FNLVQ Terhadap Citra Asli dan Citra dengan *Poisson Noise*

Selanjutnya untuk eksperimen dengan data yang mengandung *poisson noise* (Gambar 4.2), sama halnya dengan data yang mengandung *gaussian noise,*rata-rata tingkat identfikasi juga mengalami penurunan. Pada banyak skenario jaringan tidak mampu mengidentifkasi data sama sekali (tingkat pengenalan 0%). Perbedaan dari pengujian terhadap citra dengan *gaussian noise* adalah bahwa tingkat identfikasi tertinggi yang mampu dicapai data dengan *poisson noise* bisa mencapai 67%.

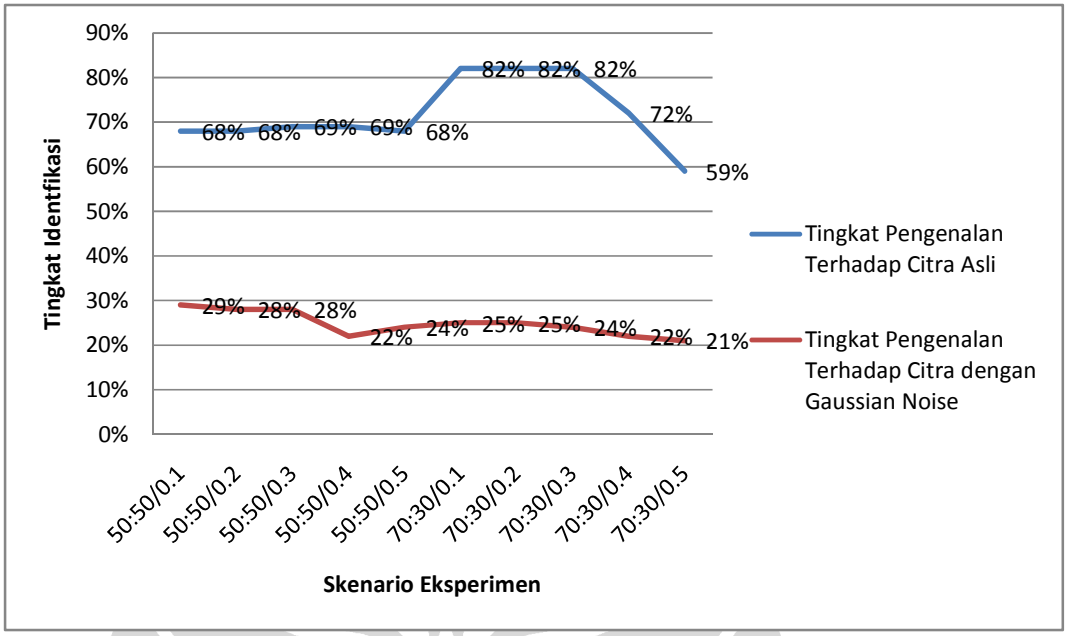


Gambar 4.3 Perbandingan Tingkat Identfikasi *Vector-Based* FNLVQ Terhadap Citra Asli dan Citra dengan *Salt&Pepper Noise*

Pada grafik tingkat identfikasi *vector-based* FNLVQ dapat dilihat bahwa sama halnya dengan *gaussian noise,* tingkat identfikasi yang mampu dicapai untuk identfikasi citra dengan *poisson noise* juga lebih tinggi (67%) daripada tingkat identfikasi maksimum terhadap citra asli (33%). Akan tetapi, secara keseluruhan, tingkat identfikasi terhadap citra wajah murni lebih tinggi dan stabil.

Eksperimen yang terakhir adalah eksperimen dengan data yang mengandung *salt&pepper noise* (Gambar 4.3). Pada eksperimen ini, sama halnya dengan data yang mengandung *gaussian noise,* rata-rata tingkat identfikasi juga mengalami penurunan yang bisa mencapai tingkat identfikasi 0%.

Pada grafik dapat dilihat bahwa kecenderungan tingkat identfikasi masih sama seperti pada *noise* yang lainnya. Tingkat identfikasi secara umum lebih baik terhadap citra asli, tetapi pengenalan citra dengan *salt&pepper noise* bisa mencapai 54%, lebih tinggi daripada tingkat pengenalan maksimum terhadap citra asli.



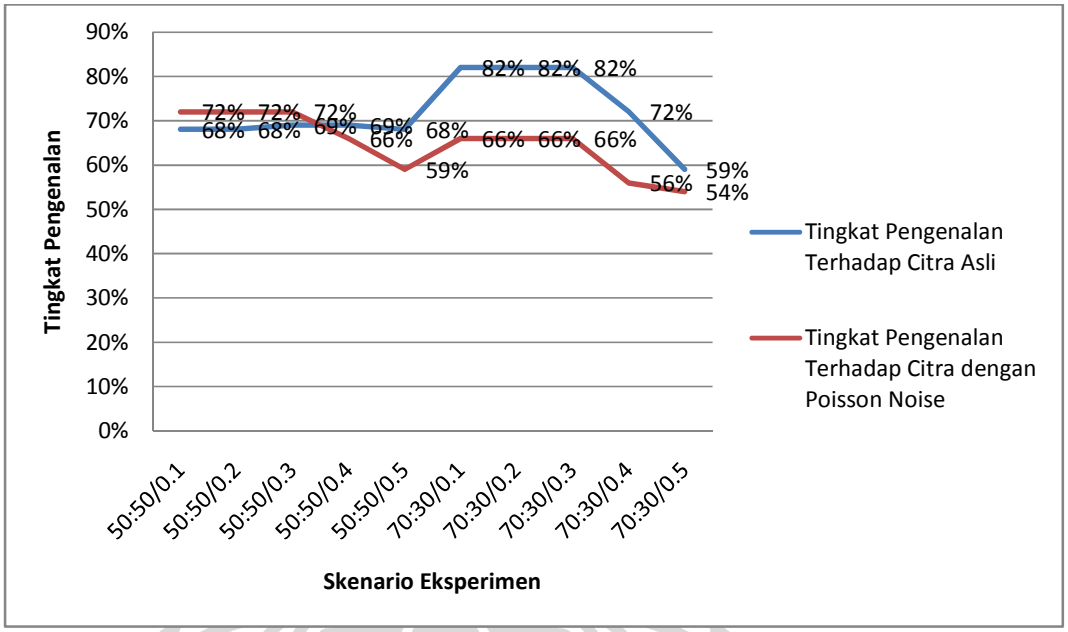
Gambar 4.4 Perbandingan Tingkat Identfikasi *Side-and-Dimension Based* FNLVQ Terhadap Citra Asli dan Citra dengan *Gaussian Noise*

Selanjutnya untuk algoritma pembelajaran *side-and-dimension-based* FNLVQ yang diujikan terhadap citra dengan *noise* yang sama. Pada algoritma *side-and-dimension* FNLVQ ini, terdapat kecenderungan yang lebih teratur daripada algoritma *vector-based* FNLVQ.

Pertama-tama untuk data citra yang mengandung *gaussian noise*. Tingkat identfikasi untuk citra dengan *noise* pada umumnya menurun untuk semua skenario. Penurunan terjadi terutama pada data teregistrasi, dari identfikasi yang berada pada kisaran 60-80%, menjadi sebesar 21-29% yang mengakibatkan penurunan sebanyak 38-58% (Gambar 4.4).

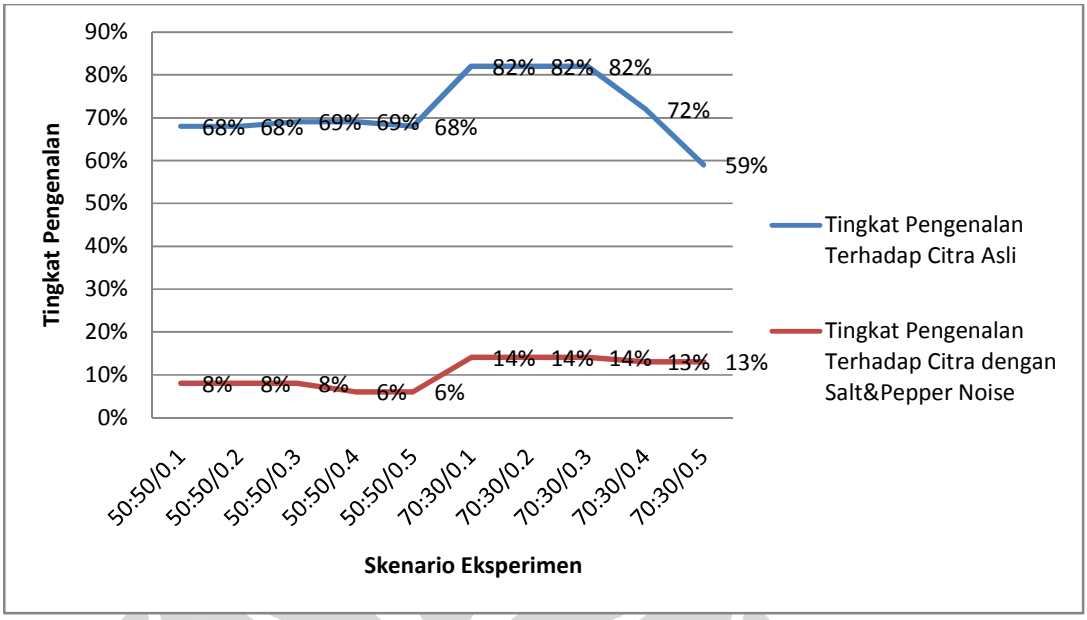
Berikutnya, untuk identfikasi citra yang mengandung *poisson noise,* penurunan identfikasi juga kurang lebih sama, walaupun besarnya identfikasi tidak sebesar penurunan pada identfikasi citra dengan *gaussian noise.* Penurunan yang terjadi untuk data teregistrasi paling besar hanya sebesar 16%, bahkan pada beberapa skenario terdapat peningkatan tingkat identfikasi sebesar 3-4% (Gambar 4.5). Terlihat bahwa pada umumnya tingkat identfikasi dengan algoritma *side-and-dimension-based* FNLVQ

untuk citra dengan *poisson noise* ini menurun. Walau demikian, penurunan yang terjadi tidak sebesar pada citra dengan *noise* jenis lainnya, bahkan pada beberapa skenario terdapat tingkat identfikasi yang justru meningkat ketika dilakukan pengujian dengan citra yang mengandung *poisson noise* sebesar 72%, yang lebih besar daripada tingkat pengenalan yang dicapai ketika memproses citra yang murni pada skenario yang sama (68%), tetapi tidak lebih besar daripada tingkat identfikasi citra asli yang paling tinggi (82%).



Gambar 4.5 Perbandingan Tingkat Identfikasi *Side-and-Dimension-Based* FNLVQ Terhadap Citra Asli dan Citra dengan *Poisson Noise*

Eksperimen terakhir dilakukan terhadap citra yang mengandung *salt&pepper noise* (Gambar 4.6). Penurunan yang terjadi akibat adanya *noise salt&pepper* ini adalah yang terbesar dibandingkan dengan penurunan karena *noise* yang lain. Hal ini dapat dilihat pada. Penurunan yang terjadi pada pengenalan data teregistrasi yang mengandung *salt&pepper noise* sangat besar dibandingkan *noise* lainnya, yaitu sebesar 46-68% penurunan. Tingkat pengenalan data yang mengandung noise hanya sebesar 8-14%.



Gambar 4.6 Perbandingan Tingkat Identfikasi *Side-and-Dimension-Based* FNLVQ Terhadap Citra Asli dan Citra dengan *Salt&Pepper Noise*

Perbandingan tingkat identifikasi antara *vector-based* FNLVQ dan *side-and-dimension-based* FNLVQ terhadap citra dengan *noise* adalah seperti pada Tabel 4.4. Skenario yang diambil disini adalah skenario dengan hasil paling baik untuk setiap *noise*. Dapat dilihat bahwa tingkat identifkasi yang bisa dicapai oleh *side-and-dimension* FNLVQ menurun untuk identifikasi citra dengan *gaussian noise* dan citra dengan *salt&pepper noise,* sedangkan tingkat identifikasi meningkat untuk citra dengan *poisson noise.*

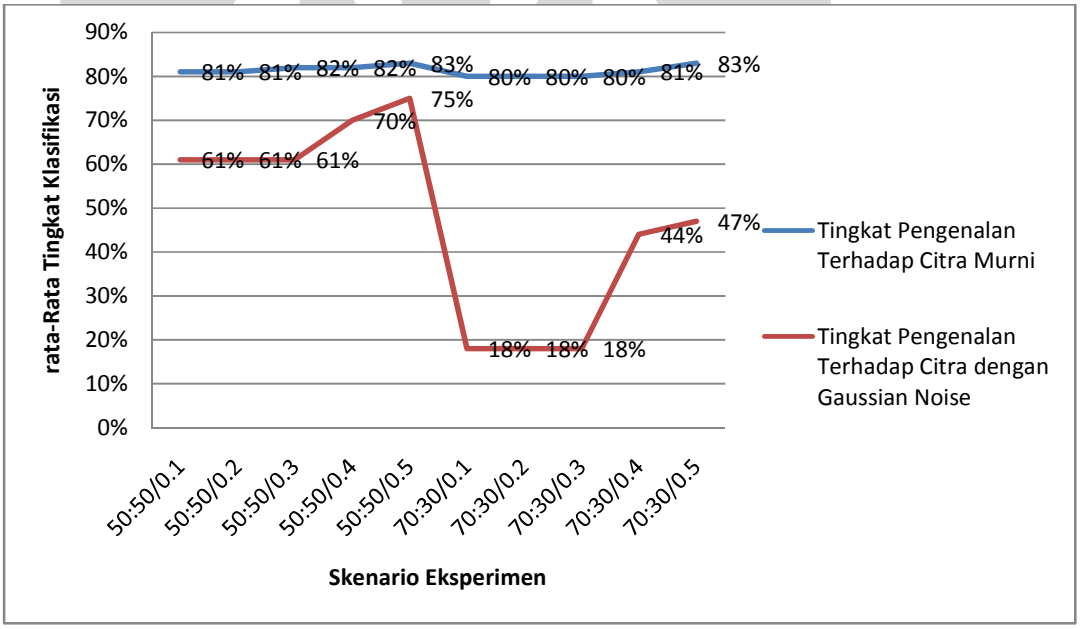
Tabel 4.4 Perubahan Tingkat Identifikasi Terhadap Citra *Noisy*

Algoritma	*Gaussian Noise*	*Poisson Noise*	*Salt&Pepper Noise*
Vector-Based FNLVQ	43%	68%	54%
Side-and-Dimension-Based FNLVQ	28%	72%	14%
Perubahan Tingkat Identifikasi.	**-15%**	**4%**	**-40%**

4.2.2 Tingkat Klasifikasi terhadap Citra *Noisy*

Pada eksperimen pertama terhadap algoritma *vector-based* FNLVQ terhadap citra yang mengandung *noise*, tingkat klasifikasi data teregistrasi dan data tidak teregistrasi tidak pernah mencapai suatu keadaan yang seimbang. Ketika tingkat klasifikasi data teregistrasi mencapai suatu angka tertentu, tingkat klasifikasi data tidak teregistrasi 0%, dan sebaliknya. Hal ini menunjukkan bahwa dalam hal klasifikasi citra yang mengandung *noise,* jaringan *vector-based* FNLVQ tidak mampu membedakan antara data teregistrasi dan data tidak teregistrasi.

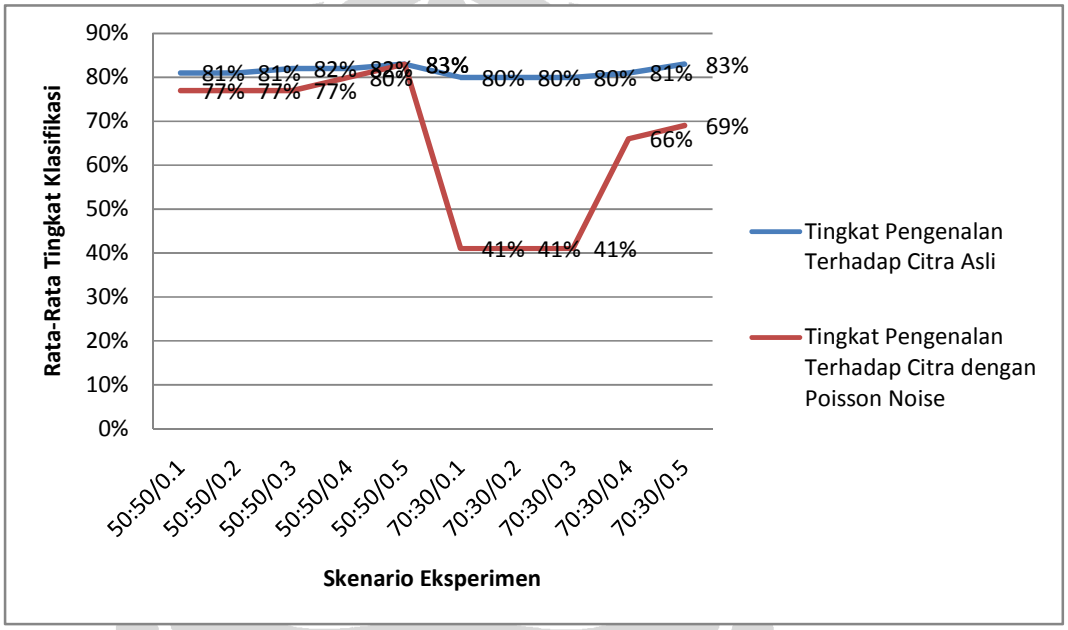
Eksperimen selanjutnya adalah eksperimen dengan menggunakan algoritma *side-and-dimension* FNLVQ. Untuk citra yang mengandung *gaussian noise,* penurunan yang terjadi cukup signifikan pada beberapa skenario(Gambar 4.7). Tingkat klasifikasi tertinggi terhadap citra dengan *gaussian* noise adalah sebesar 75%, yang masih lebih rendah daripada tingkat klasifikasi rata-rata untuk citra asli yang terendah, yaitu sebesar 71%. Tingkat klasifikasi yang terjadi fluktuatif, tetapi terdapat skenario yang masih bisa stabil pada kisaran 61%-75%.Walaupun tingkat klasifikasi citra dengan *gaussian noise*



Gambar 4.7 Perbandingan Rata-Rata Tingkat Klasifikasi *Side-and-Dimension-Based* FNLVQ terhadap Citra Asli dan Citra dengan *Gaussian Noise*

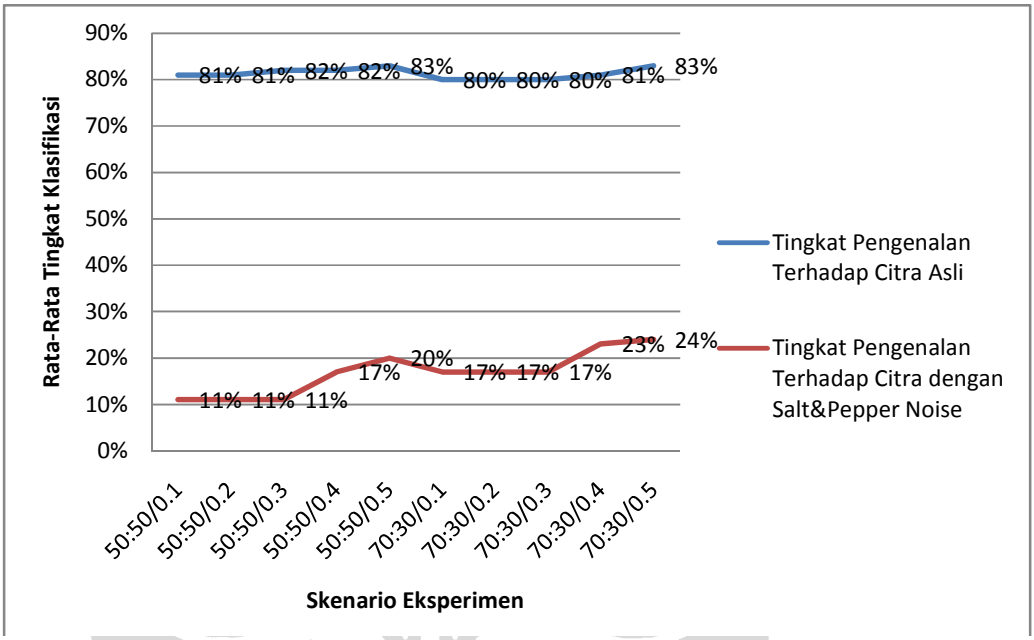
ini pada skenario tertentu mengalami penurunan yang besar, akan tetapi pada keadaan optimal rata-rata tingkat klasifikasi masih bisa mencapai angka yang cukup tinggi

Untuk citra dengan *poisson noise* (Gambar 4.8) rata-rata tingkat klasifikasi yang dicapai tidak mengalami penurunan yang berarti. Rata-rata tingkat klasifikasi terhadap citra asli berada pada kisaran 80%-83%, sedangkan rata-rata klasifikasi terhadap citra dengan *poisson noise* berada pada kisaran 41%-83%. Citra dengan *poisson noise* masih bisa mencapai keadaan maksimal tingkat klasifikasi rata-rata yang cukup tinggi, yaitu sebesar 83%.



Gambar 4.8 Perbandingan Rata-Rata Tingkat Klasifikasi *Side-and-Dimension-Based* FNLVQ terhadap Citra Asli dan Citra dengan *Poisson* Noise

Percobaan terakhir adalah pada citra dengan *salt&pepper noise*. Citra yang mengandung *noise* jenis ini mengalami penurunan yang paling besar dibandingkan dengan jenis *noise* lainnya. Tingkat klasifikasi yang dicapai hanya berada pada kisaran 11%-24% saja, jauh sekali dari tingkat pengenalan citra asli yang mencapai kisaran 80%-83% (Gambar 4.9). Jika ditinjau skenario eksperimen optimal, tingkat klasifikasi yang bisa dicapai adalah seperti pada Tabel 4.5.



Gambar 4.9 Perbandingan Rata-Rata Tingkat Klasifikasi *Side-and-Dimension-Based* FNLVQ terhadap Citra Asli dan Citra dengan *Salt&Pepper Noise*

Tingkat klasifikasi pada umumnya cenderung mengalami peningkatan, kecuali untuk pengenalan citra teregistrasi yang mengandung *gaussian noise.* Walaupun *side-and-dimension-based* FNLVQ tidak menghasilkan tingkat identifikasi yang lebih baik daripada vector based FNLVQ untuk kasus *gaussian noise* dan *salt&pepper noise,* tetapi dalam tingkat klasifikasi, side and dimension FNLVQ mampu mencapai keseimbangan.

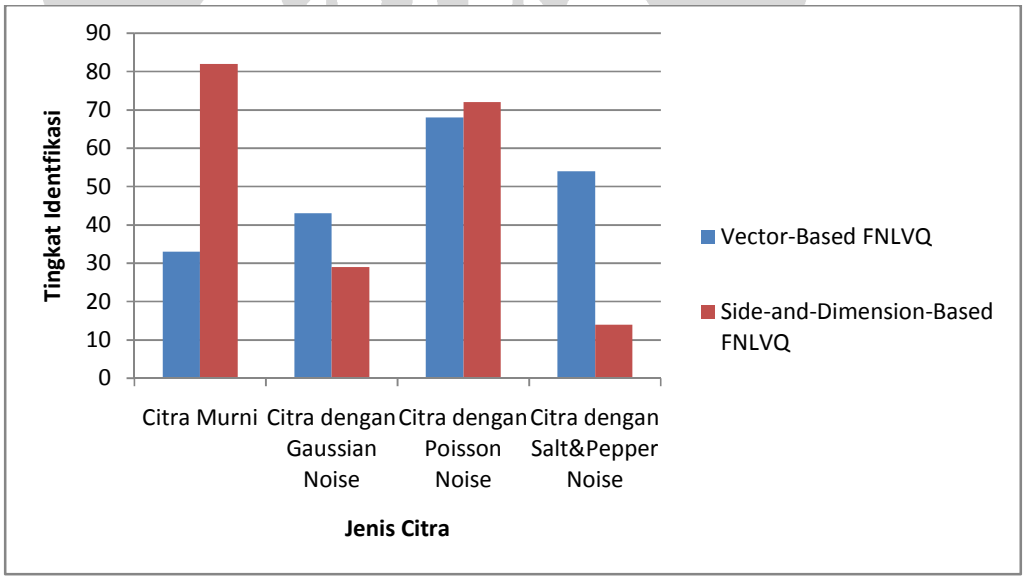
4.3 Perbandingan Tingkat Identfikasi dan Tingkat Klasifikasi

Sebagai analisis akhir pada perbandingan antara algoritma *vector-based* FNLVQ dan *side-and-dimension-based* FNLVQ, tingkat identfikasi adalah seperti pada Gambar 4.10. Tingkat identfikasi *side-and-dimension-based* FNLVQ terlihat meningkat secara signifikan untuk identifikasi citra asli. Di sisi lain untuk identfikasi citra yang mengandung *noise, vector based* FNLVQ masih lebih baik untuk jenis *noise gaussian* dan *salt&pepper.* Untuk jenis *noise poisson,* tingkat identfikasi *side-and-dimension-based* FNLVQ lebih tinggi sedikit dibandingkan *vector-based* FNLVQ.

Tabel 4.5 Perubahan Tingkat Klasifikasi Terhadap Citra Noisy

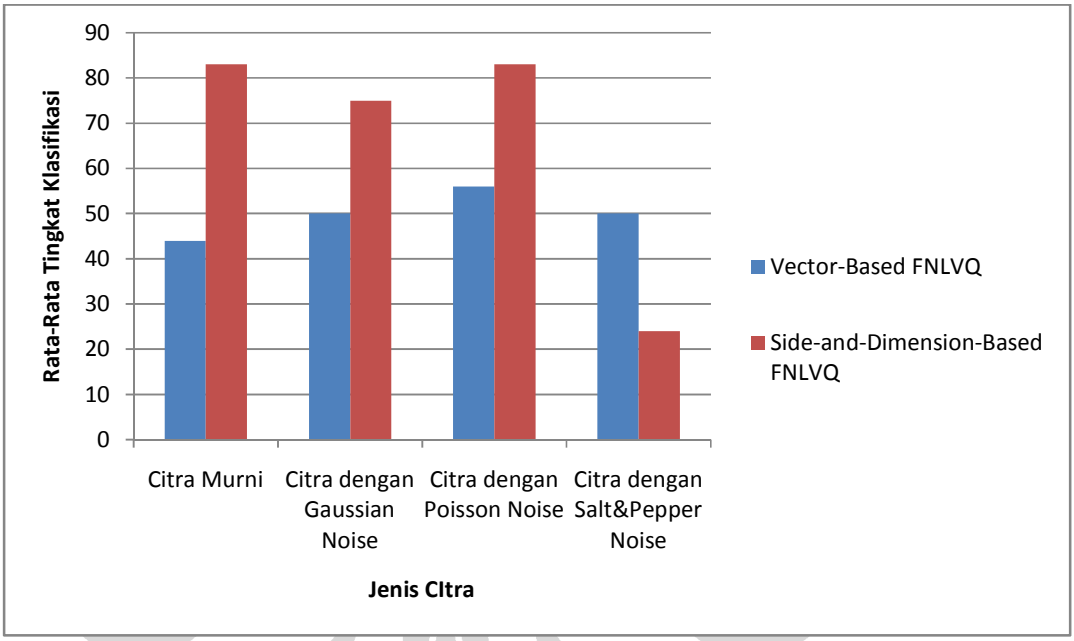
Algoritma	*Gaussian Noise*		*Poisson Noise*		*Salt&Pepper Noise*	
	Data Teregistrasi	Data Tak Teregistrasi	Data Teregistrasi	Data Tak Teregistrasi	Data Teregistrasi	Data Tak Teregistrasi
Vector-Based FNLVQ	71%	0%	74%	0%	62%	0%
Side-and-Dimension-Based FNLVQ	68%	72%	92%	74%	35%	61%
Perubahan	**-3%**	**72%**	**18%**	**74%**	**27%**	**61%**

Berikutnya untuk perbandingan rata-rata tingkat klasifikasi antara kedua algoritma adalah seperti pada Gambar 4.11. Tingkat klasifikasi rata-rata yang dicapai oleh *side-*



Gambar 4.10 Perbandingan Tingkat Identifikasi *Vector-Based* FNLVQ dan *Side-and-Dimension-Based* FNLVQ

and-dimension-based FNLVQ lebih tinggi daripada tingkat klasifikasi rata-rata *vector-based* FNLVQ pada klasifikasi citra asli, citra dengan *gaussian noise,* serta citra dengan *poisson noise,* sedangkan untuk citra dengan *salt&pepper noise,* tingkat klasifikasi rata-rata lebih tinggi dicapai dengan menggunakan algoritma *vector-based* FNLVQ.



Gambar 4.11 Perbandingan Rata-Rata Tingkat Klasifikasi Data Teregistrasi dan Data Tidak Teregistrasi *Vector-Based* FNLVQ dan *Side-and-Dimension-Based* FNLVQ

Pembelajaran *side-and-dimension-based* FNLVQ adalah pembelajaran yang lebih teliti dan tepat, karena pembelajaran berlangsung pada satuan yang lebih kecil. Vektor pewakil setiap kelas dilatih dan disesuaikan sedemikian hingga bisa cukup mewakili kelas tersebut. Pembelajaran yang terjadi menyesuaikan vektor pewakil per dimensi agar mencakup semua nilai yang mungkin untuk masing-masing dimensi. Akibatnya tingkat identfikasi juga lebih tinggi, dapat dilihat dari peningkatan tingkat identifikasi dari *vector-based* FNLVQ hingga lebih dari 2 kali lipat.

Tingkat klasifikasi yang bisa dicapai oleh *side-and-dimension-based* FNLVQ terhadap juga tentunya lebih baik lagi, karena dengan pembelajaran yang lebih teliti ini, dimensi-dimensi vektor pewakil disesuaikan dengan tepat dengan data kelas tersebut, sehingga mampu mengklasifikasikan data teregistrasi dan data tidak teregistrasi dengan baik.

Dalam hal identifikasi citra *noisy,* dimensi-dimensi pada vektor pewakil perlu dilatih sehingga bisa mengenali rentang nilai yang lebih lebar untuk setiap dimensinya. Jika pembelajaran yang terjadi sedetil pembelajaran pada *side-and-dimension-based* FNLVQ, setiap dimensi dilatih agar mencakup nilai dengan tepat, tanpa memberikan toleransi rentang nilai yang lebih lebar. Sebaliknya, *vector-based* FNLVQ melakukan pembelajaran pada satuan yang lebih besar, yaitu satuan vektor. Dengan demikian, pembelajaran yang terjadi lebih banyak, sehingga dimensi-dimensi akan bisa mencakup rentang nilai yang lebih lebar. Oleh karena itu, *vector based* FNLVQ mampu mencapai tingkat identfikasi yang lebih tinggi.

Selain memberi rentang toleransi nilai lebih lebar untuk kepentingan identifikasi citra *noisy,* pembelajaran *vector-based* FNLVQ ini membawa dampak buruk pada tingkat klasifikasi yang dicapai. Pembelajaran yang banyak ini membuat rentang nilai setiap dimensi sedemkian lebar untuk mengantisipasi nilai-nilai pada citra *noisy,* akibatnya jaringan tidak lagi mampu membedakan citra teregistrasi dan citra tidak teregistrasi. Hal ini disebabkan oleh lebarnya dimensi vektor pewakil yang mencakup pula nilai-nilai yang tidak mewakili kelas tersebut, bahkan mencakup nilai pada citra *outlier.*

Rata-rata tingkat klasifikasi *vector-based* FNLVQ terhadap citra *noisy* sangat buruk, karena jika citra teregistrasi bisa dikenali, citra tidak teregistrasi tidak bisa dikenali, dan sebaliknya. Rata-rata tingkat klasifikasi yang terlihat sebesar 50%-55% untuk *vector-based* FNLVQ tidak menunjukkan bahwa jaringan bisa mengklasifikasikan citra *noisy* dengan baik, karena itu hanya berasal dari pengenalan citra teregistrasi saja, atau sebaliknya. Jaringan yang demikian tentunya tidak bermanfaat untuk melakukan klasifikasi.