Perbandingan PNN dan LVQ dalam Identifikasi Jenis Bercak pada Daun Cabai

Jaka Permadi

Jurusan Teknik Informatika, Politeknik Negeri Tanah Laut
Jl. A. Yani Km.06 Desa Panggung, Pelaihari
jakapermadi.88@gmail.com

Abstrak *– Learning Vector Quantization (LVQ) dan Probabilistic Neural Network (PNN) adalah dua buah metode dari Jaringan Syaraf Tiruan. Untuk beberapa kasus, dapat dikatakan LVQ dan PNN memiliki kemampuan yang setara dalam kasus klasifikasi, seperti identifikasi atau deteksi. Pada penelitian ini, dilakukan perbandingan kemampuan LVQ dengan PNN untuk identifikasi jenis bercak pada daun dengan memanfaatkan metode cross validation k-fold CV. Hasil dari penelitian yang dilakukan LVQ memiliki estimasi test error sebesar 0.4227 dengan akurasi validasi tertinggi sebesar 62.89%. Hasil yang dimiliki LVQ ini lebih baik daripada PNN dengan nilai estimasi test error sebesar 0.4495 dengan akurasi validasi tertinggi sebesar 59.79%. Dengan kata lain LVQ lebih baik daripada PNN dalam kasus identifikasi jenis bercak pada daun cabai berdasarkan citra daun tersebut.*

Kata Kunci: Identifikasi, k-fold CV, LVQ, PNN, Test error.

1. PENDAHULUAN

Penelitian tentang identifikasi jenis penyakit pada tanaaman cabai dengan menggunakan *Probabilistic Neural Networks* (PNN) berdasarkan pada citra daunnya telah dilakukan oleh Permadi (2016). *Training set*, *validation set* dan *testing set* pada penelitian tersebut ditentukan secara manual dengan perbandingan 70% : 20% : 10%. Kelas-kelasnya dibatasi menjadi 4 kelas, yaitu Bercak daun serkospora, Bercak karena bakteri, Bercak kelabu stemfilium dan Bukan bercak penyakit. Akurasi pengujian tertinggi dengan PNN sebesar 94,737%, dimana lebih tinggi dibandingkan JST *Backpropagation* dengan akurasi pengujian tertinggi sebesar 91,447% .

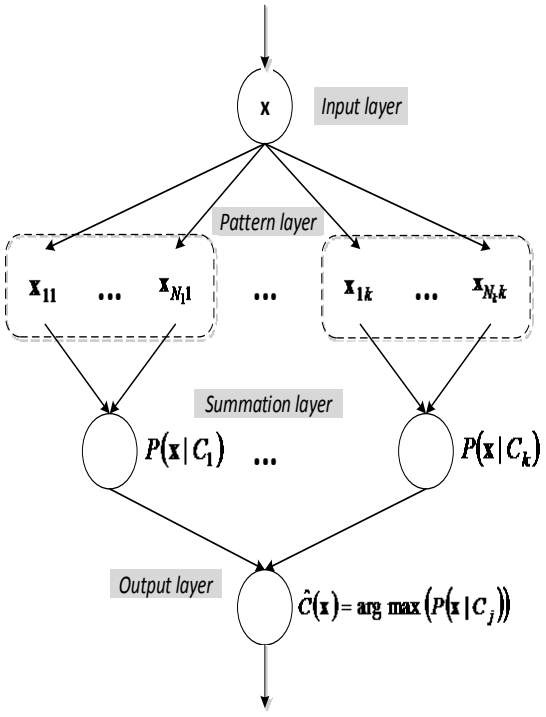
Learning Vector Quantization (LVQ) merupakan mesin pembelajaran dengan teknik pembelajaran terawasi, yang dikembangkan dari algoritma SOM (Haykin, 1999). Pada beberapa kasus klasifikasi, LVQ memiliki kemampuan yang sama dengan PNN. Pada penelitian yang dilakukan Liu dkk. (2013) tentang diagnosa kesalahan transformator daya, dimana akurasi pengujian LVQ dengan PNN sebesar 86,67%. Di beberapa kasus klasifikasi lainnya, LVQ memiliki kemampuan yang berbeda dengan PNN dalam mengenali kelas-kelas yang ada. Pada penelitian yang dilakukan George dkk. (2012) tentang klasifikasi tumor payudara, PNN menghasilkan akurasi pengujian yang lebih tinggi daripada LVQ. Sementara pada penelitian yang dilakukan Chen dkk. (2000) tentang klasifikasi sinyal bawah laut, PNN menghasilkan akurasi pengujian yang lebih rendah daripada LVQ.

Oleh karena latar belakang tersebut, pada penelitian ini dibandingkan kemampuan LVQ dan PNN dalam mengenali jenis penyakit cabai berdasarkan citra daunnya. Jumlah kelas yang digunakan dibatasi menjadi 3 kelas, yaitu Bercak daun serkospora, Bercak kelabu stemfilium dan Bercak karena bakteri. *Cross validation* digunakan untuk menghindari subjektifitas dalam penelitian yang dilakukan, dengan perbandingan *training set* dan *validation set* sebesar 80% : 20%.

2. LANDASAN TEORI

2.1 *Probabilistic Neural Networks* (PNN)

PNN adalah salah satu metode dari jaringan syaraf tiruan yang keputusannya berdasarkan pada aturan Bayesian (Theodoridis & Koutroumbas, 2009). PNN terdiri atas 4 layer, yaitu *input layer*, *pattern layer*, *summation layer* dan *output layer*.



Gambar 1. Arsitektur PNN (Mao dkk., 2000)

 ISSN: 2460-173X

Input layer

Pada *input layer*, data-data yang akan dikenali kelasnya disimpan untuk kemudian akan didistribusikan ke dalam proses pada *pattern layer*. Vektor **x** merupakan vektor *input* yang dapat dijabarkan menjadi $[x_1 \;\; x_2 \;\; \cdots \;\; x_l]^T$ dengan l adalah jumlah atribut dari vektor **x** tersebut.

Pattern layer

Pattern layer berisi data-data yang telah dibagi ke dalam kelas-kelas C_j yang dinotasikan dengan $\mathbf{x}_{ij}$, dengan $j = 1,2,\ldots,k$ adalah banyaknya kelas dan $i = 1,2,\ldots,N_j$ adalah banyaknya data pada kelas j.

Summation layer

Setiap atribut pada **x** diukur kedekatannya dengan setiap atribut dari setiap data $\mathbf{x}_{ij}$ pada setiap kelas j. Diberikan fungsi *likelihood* yang merupakan *output* dari tiap neuron pada *pattern layer* seperti ditunjukkan pada persamaan (1).

$$\phi_{ij}(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{p}{2}}\cdot\sigma^p}\cdot\frac{1}{N_j}\cdot e^{\left(-\frac{(\mathbf{x}-\mathbf{x}_{ij})^T(\mathbf{x}-\mathbf{x}_{ij})}{2\sigma^2}\right)} \qquad (1)$$

Pada persamaan (1), p menyatakan besarnya dimensi dari vektor **x** dan σ adalah parameter penghalus. Probabilitas **x** terhadap kelas C_j, yang selanjutnya disebut sebagai *probability density function* (PDF) merupakan suatu nilai yang dihasilkan pada *summation layer*. Nilai PDF didapatkan dengan menambahkan semua nilai *likelihood* pada suatu kelas, seperti yang ditunjukkanpada persamaan (2) (Mao dkk., 2000).

$$PDF(\mathbf{x}|C_j) = \sum_{i=1}^{N_j}\phi_{ij}(\mathbf{x}) \qquad (2)$$

Output layer

Pada *output layer* diperoleh kelas dari data **x** yang diinputkan pada *input layer*. Estimasi kelas **x** dipilih berdasarkan nilai PDF terbesar. Jika estimasi kelas dari data **x** tersebut dinotasikan dengan $\hat{C}(\mathbf{x})$, maka persamaan yang menentukan estimasi kelas dari data **x** ditunjukkan pada persamaan (3) (Mao dkk., 2000).

$$\hat{C}(\mathbf{x}) = \arg\max_{1\le j\le k}\left(PDF(\mathbf{x}|C_j)\right) \qquad (3)$$

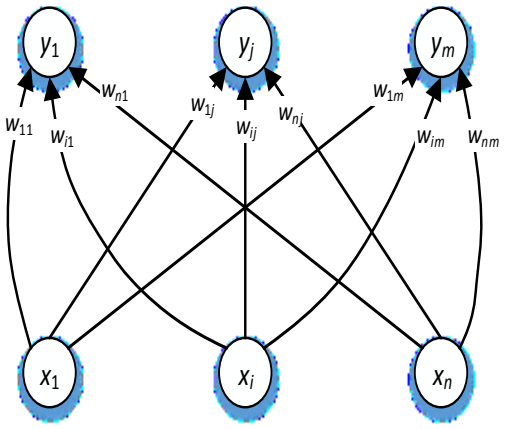
Pembelajaran pada PNN dikategorikan sebagai pembelajaran yang tidak terawasi (Mao dkk., 2000). Sehingga tidak diperlukan proses pelatihan yang berulang (iterasi) untuk memperbaiki parameter-parameter yang nantinya digunakan dalam mengenali kelas dari suatu data. Parameter yang dimaksud adalah *smoothing parameter* α. Algoritma pelatihan PNN diberikan pada Gambar 2.

1. Simpan training set ke dalam $\mathbf{x}_{ij}$ pada pattern layer.
2. Set nilai Smooth Parameter α_{min}, α_{max} dan β, dimana β adalah parameter penambah.
3. For $\alpha = \alpha_{min}$ to α_{max} step β
4. For $p = 1$ to $N_{trainingset}$
5. $\mathbf{x}$ = Training set$_p$
6. For each j
7. For each $\hat{\imath}$
8. Hitung $\phi_{ij}(\mathbf{x})$
9. Hitung $PDF(\mathbf{x}|C_j)$
10. Tentukan $\hat{C}(\mathbf{x})$
11. Hitung akurasi training
12. For $p = 1$ to $N_{validationset}$
13. $\mathbf{x}$ = Validation set$_p$
14. For each j
15. For each $\hat{\imath}$
16. Hitung $\phi_{ij}(\mathbf{x})$
17. Hitung $PDF(\mathbf{x}|C_j)$
18. Tentukan $\hat{C}(\mathbf{x})$
19. Hitung akurasi validasi
20. Cari α dimana akurasi training dan akurasi validasi tertinggi.
21. Simpan α dan *pattern units*.

Gambar 2. Algoritma pelatihan PNN

2.2 *Learning Vector Quantization* (LVQ)

LVQ adalah metode klasifikasi dimana setiap neuron outputnya merepresentasikan kelas-kelas tertentu. Selama proses pelatihan, unit-unit output diposisikan agar mendekati nilai target dengan mengatur bobot-bobotnya melalui pembelajaran terawasi (Fausett, 1994). Arsitektur dari LVQ dapat dilihat pada Gambar 3.



Gambar 3. Arsitektur LVQ (Fausett, 1994)

Algoritma pelatihan LVQ dapat dilihat pada Gambar 4.

1. Simpan training set di dalam $\mathbf{x}$.
2. $\boldsymbol{T}$ adalah kelas target dari $\mathbf{x}$.
3. Tentukan learning rate awal α.
4. Tentukan parameter pengurang β dan batas learning rate α_{min}.
5. Inisialisasi bobot awal $\mathbf{w}_j$ dengan nilai acak.
6. Do Until *StopCondition* = True
7. For each $\mathbf{x}$
8. $C_j = \arg\min_j \|\mathbf{x} - \mathbf{w}_j\|$
9. If $\boldsymbol{T} = C_j$, then
10. $\mathbf{w}_j = \mathbf{w}_j + \alpha(\mathbf{x} - \mathbf{w}_j)$
11. Else
12. $\mathbf{w}_j = \mathbf{w}_j - \alpha(\mathbf{x} - \mathbf{w}_j)$
13. $\alpha = \alpha - \beta$
14. If $\alpha \leq \alpha_{min}$
15. *StopCondition* = True
16. Simpan $\mathbf{w}_j$

Gambar 4. Algoritma pelatihan LVQ (Fausett, 1994)

2.3 *Cross Validation*

Pada metode klasifikasi jaringan syaraf tiruan, diperlukan tahapan pelatihan yang menggunakan *training set* untuk mendapatkan parameter-parameter yang menyebabkan *training error* menjadi kecil. Parameter-parameter tersebut diuji pada tahapan pengujian dengan menggunakan *testing set* dan kemudian dihitung nilai *test error* pada tahapan ini. Seringkali didapatkan kasus dimana hasil pengujian memiliki perbedaan yang cukup kontras dengan hasil pelatihan. Kekurangan yang cukup besar dari penggunaan *testing set* secara langsung untuk mendapatkan nilai *test error*, sejumlah teknik pun digunakan untuk mengestimasi nilai tersebut dengan memanfaatkan *training set*. Salah satunya adalah *cross validation* (James dkk., 2013).

2.3.1 Pendekatan *validation set*

Pendekatan *validation set* merupakan strategi yang sangat sederhana jika seseorang akan mengestimasi *test error* menggunakan data-data observasi. Data-data tersebut dibagi menjadi *training set* dan *validation set* secara acak. Model dilatih menggunakan *training set*, dan model yang terlatih tersebut digunakan untuk memprediksi responsnya terhadap data observasi pada *validation set*. Nilai *validation error* diambil sebagai estimasi untuk *test error* (James dkk., 2013).

2.3.2 *k-Fold cross validation*

Pendekatan *k-fold cross validation* (*k-fold* CV) membagi sekumpulan data observasi menjadi kelompok-kelompok *k* data (atau disebut *fold*). *Fold* pertama dianggap sebagai *validation set* dan sisanya adalah *training set*. Nilai *error* E_1 kemudian didapatkan. Prosedur ini dilakukan berulang kali sebanyak *k* iterasi. Setiap iterasi, *validation set* diambil dari *fold* yang berbeda. Sehingga tahapan ini akan menghasilkan sebanyak *k* estimasi *test error*, yaitu E_1, E_2, …, E_k. Persamaan (4) kemudian digunakan untuk mengestimasi nilai-nilai *test error* tersebut (James dkk., 2013).

$$\mathrm{CV} = \frac{1}{k}\sum_{i=1}^{k} E_i \qquad (4)$$

PNN dan LVQ adalah dua buah metode dengan nilai *output* pengenalannya kualitatif. Dengan demikian nilai estimasi *test error* dapat menggunakan persamaan (5) (James dkk., 2013)

$$E_i = \frac{1}{n}\sum_{p=1}^{n} I(y_p \neq \hat{y}_p) \qquad (5)$$

dengan y_p adalah nilai *output* pada data ke-*p* dan $\hat{y}_p$ adalah nilai target pada data ke-*p*. Nilai $I(y_p \neq \hat{y}_p)$ akan bernilai 1 jika $y_p \neq \hat{y}_p$ dan bernilai 0 jika sebaliknya. *Classifier* yang baik adalah *classifier* dengan *test error* terkecil (James dkk, 2013). Besarnya akurasi berlawanan dengan persamaan (5).

3. METODE PENELITIAN

Tahapan yang dilakukan dalam penelitian ini adalah sebagai berikut :

1. Studi literatur
   Pada tahapan ini dilakukan pembacaan buku dan jurnal yang terkait dengan PNN, LVQ dan *cross validation*.
2. Pengumpulan data
   Data yang digunakan merupakan data-data hasil ekstraksi fitur dan pelabelan dari citra-citra daun cabai pada penelitian yang dilakukan oleh Permadi (2016). Dataset yang digunakan sebanyak 473 data, dengan rincian 216 data untuk kelas Bercak daun serkospora, 96 data untuk kelas Bercak kelabu stemfilium dan 161 data untuk kelas Bercak karena bakteri. Setiap data memiliki 13 atribut, yaitu nilai ASM_Max, ASM_Mean, Contrast_Max, Contrast_Mean, IDF_Max, IDF_Mean, Entropy_Max, Entropy_Mean, R_Mean, G_Mean, B_Mean, Selisih_EntropyGB dan *Circularity ratio*.
3. Desain penelitian dan implementasi
   Pada tahapan ini dibangun skema rancangan untuk proses *training* dan *validating* pada PNN dan LVQ dengan *k-fold* CV. Selanjutnya dilakukan pengimplementasian desain penelitian tersebut ke dalam baris program. Bahasa pemrograman yang digunakan adalah VB.Net, yang digunakan penulis untuk membantu mendapatkan estimasi *test error* (berupa tingkat akurasi) dari eksplorasi yang dilakukan.
4. Pembahasan
   Pada tahapan ini dilakukan pembahasan dari hasil yang didapatkan, yaitu tingkat akurasi dari proses *validating* menggunakan *validation set* pada eksplorasi yang dilakukan pada arsitektur LVQ dan PNN.

 ISSN: 2460-173X

5. Penarikan kesimpulan
   Pada tahapan ini dilakukan penarikan kesimpulan untuk menjawab permasalahan dan memenuhi tujuan penelitian yang telah dijabarkan pada bab pendahuluan.

4. HASIL DAN PEMBAHASAN

4.1 ANALISIS PERANCANGAN

4.1.1 Pembagian Dataset dengan *k-fold* CV

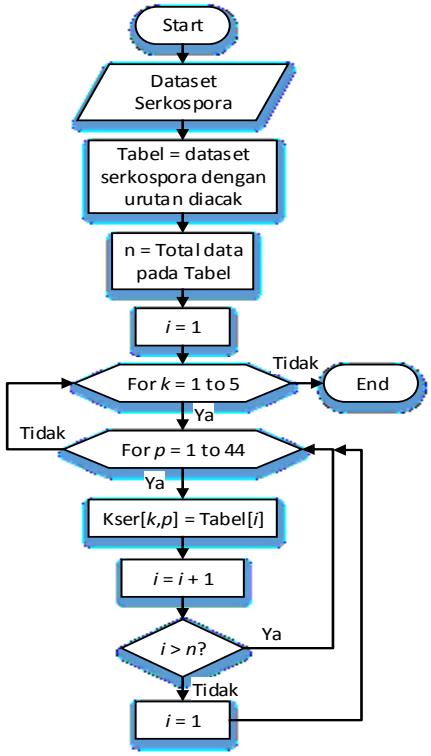
Pada penelitian ini, dibuat perbandingan jumlah *training set* dan *validation set* sebanyak 80% : 20%. Dengan demikian jumlah k = 5 untuk *k-fold* CV (Ada 5 *fold*). Teknik pembagiannya dilakukan pada setiap kelas. Sehingga jumlah *training set* dan *validation set* untuk setiap kelas diperlihatkan pada **Tabel 1**.

Tabel 1. Jumlah *training set* dan *validation set*

No	Kelas	*Training Set*	*Validation Set*	Total
1	Bercak daun serkospora	176	44	220
2	Bercak kelabu stemfilium	80	20	100
3	Bercak karena bakteri	132	33	165
Total		**388**	**97**	**485**

Terlihat bahwa banyaknya data yang digunakan setelah pembagian dengan *k-fold* CV berbeda dengan banyaknya data yang tersedia. Keadaan ini disebabkan karena hasil perhitungan dengan menggunakan perbandingan antara *training set* dan *validation set* dibulatkan ke atas. Setiap *fold* akan menyimpan data Bercak daun serkospora, Bercak kelabu stemfilium dan Bercak karena bakteri dengan perbandingan yang sama untuk setiap kelas. Dengan kata lain di setiap *fold* akan disiapkan tempat-tempat khusus untuk setiap kelas, katakanlah Kser[k] untuk kelas Bercak daun serkospora, Kstem[k] untuk kelas Bercak kelabu stemfilium dan Kbak[k] untuk kelas Bercak karena bakteri. Dataset sebelumnya disimpan dalam database, dipisah-pisahkan berdasarkan kelasnya dan diberikan nomor urut. Pengacakan data dilakukan dengan menampilkan dataset secara acak (tidak berdasar nomor urut) pada tabel dan kemudian memasukkan setiap data berdasarkan urutan indeks tabelnya. Karena banyaknya data yang dibutuhka untuk semua *fold* lebih besar daripada banyaknya data yang tersedia, maka untuk *fold* terakhir jika indeks tabelnya mencapai total data, maka indeks tabel dikembalikan ke indeks pertama.

Perancangan pengisian *fold* untuk kelas Bercak daun serkospora dapat dilihat pada Gambar 5.



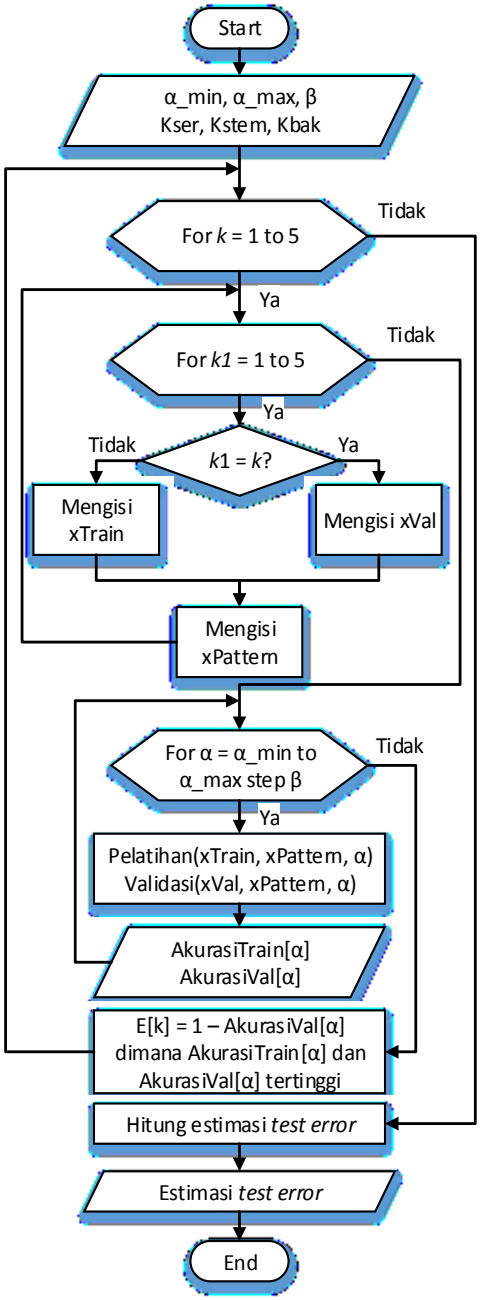
Gambar 5. *Flowchart* pengisian data Bercak daun serkospora untuk setiap *fold*

Flowchart pengisian data bercak lainnya untuk setiap *fold* serupa dengan *flowchart* pengisian data Bercak daun serkospora. Perbedaannya terletak pada dataset yang digunakan dan variabel penyimpannya.

4.1.2 Perancangan Proses Menentukan Estimasi *Test Error* pada PNN

Proses pelatihan dan validasi pada PNN dilakukan pada *input layer*, *pattern layer*, *summation layer* dan *output layer*. Pada *input layer*, disiapkan variabel xTrain[p] untuk menyimpan *training set* dan xVal[p] untuk menyimpan *validation set* dengan p adalah indeks data. Disiapkan juga variabel xPattern[k, i] untuk menyimpan *pattern units*, dengan k adalah indeks kelas dan i adalah indeks data pada *pattern unit*. Pada penelitian ini, kelas Bercak daun serkospora memiliki indeks k = 1, kelas Bercak kelabu stemfilium memiliki indeks k = 2, dan Bercak karena bakteri memiliki indeks k = 3. Pada *summation layer* dilakukan perhitungan *likelihood* dari *input units* terhadap *pattern units*, yang selanjutnya pada *output layer* didapatkan nilai *output* yang berupa kelas dari *input unit* tersebut. Semua proses pelatihan dan validasi ini dilakukan dengan nilai *smoothing parameter* α yang sebelumnya ditentukan dari *range* α_{min} sampai α_{max} dengan penambahan β. Estimasi *test error* diambil dari nilai *error* terkecil dari proses validasi pada α tertentu, dimana akurasi pelatihan dan validasinya tertinggi pada parameter tersebut. Perancangan tahapan penentuan estimasi *test error* dengan PNN dapat dilihat pada Gambar 6.

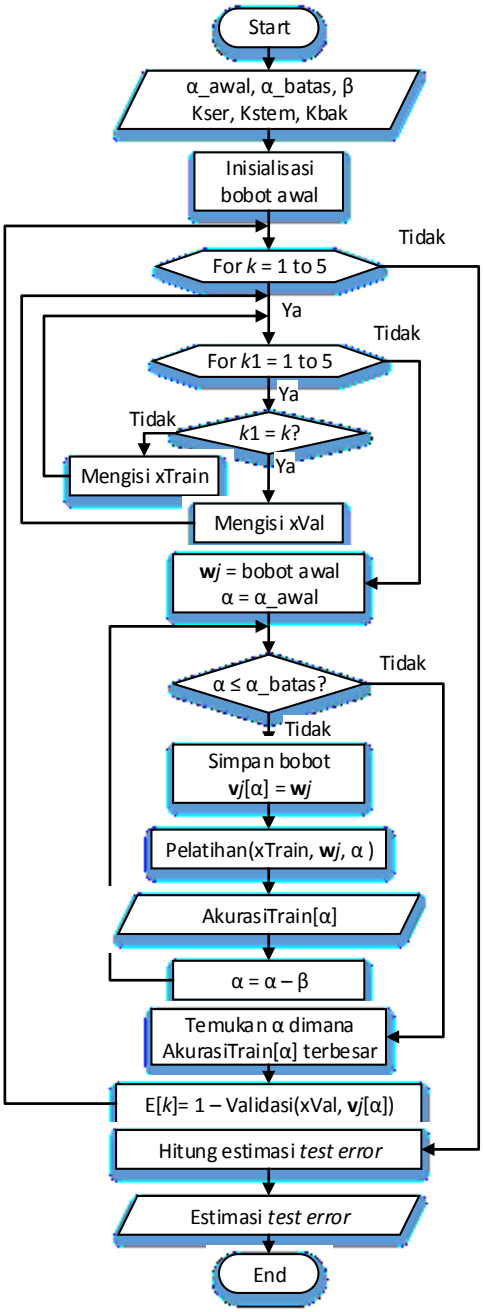
ISSN: 2460-173X



Gambar 6. *Flowchart* estimasi *test error* dari pelatihan dan validasi PNN

4.1.3 Perancangan Proses Menentukan Estimasi *Test Error* pada LVQ

Proses pelatihan dan validasi pada LVQ dilakukan dengan inisialisasi bobot awal secara acak dan kemudian menyimpan xTrain[p] dan xVal[p] di setiap *fold*. Pelatihan dilakukan dengan menggunakan *learning rate* α yang di setiap iterasi semakin berkurang nilainya sebesar β. Proses iterasi akan berakhir ketika nilai α sudah lebih kecil atau sama dengan α_batas, yaitu suatu batasan yang ditetapkan dari awal. Aturan indeks kelas yang digunakan sama dengan yang diterapkan pada PNN. Estimasi *test error* di setiap *fold* didapatkan dengan menghitung akurasi validasi, dimana nilai-nilai *output*nya diperoleh menggunakan bobot-bobot hasil pelatihan pada iterasi tertentu, dimana pada iterasi tersebut akurasi pelatihannya tertinggi. Perancangan tahapan penentuan estimasi *test error* dengan LVQ dapat dilihat pada Gambar 7.



Gambar 7. *Flowchart* estimasi *test error* dari pelatihan dan validasi LVQ

4.2 Estimasi *Test Error* pada PNN

Pada PNN, nilai estimasi *test error* memanfaatkan akurasi validasi terbesar di setiap *k-fold*. Nilai akurasi yang dimaksud adalah ketepatan untuk seluruh data, baik pada *training set* maupun *validation set* (bukan akurasi setiap kelas). **Tabel 2** menggambarkan nilai akurasi dan estimasi *test error* setelah melakukan pelatihan dan validasi menggunakan PNN dengan α_min = 0.001, α_max = 0.07 dan β = 0.0001.

Tabel 2. Hasil penggunaan metode PNN

K-fold	Akurasi terbesar			*E*(*k*)
	Smooth Parameter (α)	Pelatihan	Validasi	(1-AkurasiVal)
1	0.0155	100%	55.67%	0.4433
2	0.0117	100%	48.45%	0.5155
3	0.0143	100%	54.64%	0.4536
4	0.0151	100%	56.7%	0.433
5	0.0162	100%	59.79%	0.4021
		Estimasi test error		**0.4495**

Pada k = 1, akurasi pelatihan dan validasi tertinggi berada ketika nilai α = 0.0155 dengan akurasi validasi sebesar 55.67%, atau sebesar 0.5567. Dengan demikian, berdasarkan persamaan (5), nilai $E(1) = 0.4433$. Pada k = 2, akurasi pelatihan dan validasi tertinggi berada ketika nilai $\alpha = 0.0117$ dengan akurasi validasi sebesar 48.45% atau $E(2) = 0.5155$. Pada k = 3, akurasi pelatihan dan validasi tertinggi berada ketika $\alpha = 0.0143$ dengan akurasi validasi sebesar 54.64% atau $E(3) = 0.4536$. Pada k = 4, akurasi pelatihan dan validasi tertinggi berada pada $\alpha = 0.0151$ dengan akurasi validasi sebesar 56.7% atau $E(4) = 0.433$. Pada k = 5, akurasi pelatihan dan validasi tertinggi berada pada $\alpha = 0.0162$ dengan akurasi validasi sebesar 59.79% atau $E(5) = 0.4021$. Berdasarkan persamaan (4) diperoleh nilai estimasi *test error* dengan menggunakan PNN sebesar 0.4495.

4.3 Estimasi *Test Error* pada LVQ

Pada LVQ, dilakukan eksplorasi pada nilai *learning rate* α dengan memvariasikan nilai-nilai α_{awal}, α_{batas} dan parameter pengurang β. Ada lima eksplorasi yang dilakukan. Setiap eksplorasi dilakukan *k-fold* CV dengan k = 5. Di setiap k dilakukan proses pelatihan dengan menyediakan bobot awal $\mathbf{w}_j$ yang bernilai *random*. Selama pelatihan dilakukan proses *update* bobot dan pengurangan nilai *learning rate* sebesar β. Validasi dilakukan dengan menggunakan bobot yang dihasilkan dari proses pelatihan pada iterasi tertentu, dimana pada iterasi tersebut akurasi pelatihan tertinggi.

Eksplorasi pertama dilakukan dengan nilai $\alpha_{awal} = 0.003$, $\alpha_{batas} = 0.0001$ dan $\beta = 0.00001$. Hasil pelatihan dan validasi pada eksplorasi pertama ditampilkan pada **Tabel 3**.

Tabel 3. Hasil penggunaan LVQ, dengan $\boldsymbol{\alpha_{awal} = 0.003}$, $\boldsymbol{\alpha_{batas} = 0.0001}$ dan $\boldsymbol{\beta = 0.00001}$

K-fold	Akurasi terbesar		*E*(*k*)
	Pelatihan	Validasi	(1-AkurasiVal)
1	59.02%	58.76%	0.4124
2	60.31%	56.7%	0.433
3	60.05%	51.55%	0.4845
4	60.57%	62.89%	0.3711
5	56.7%	57.73%	0.4227
	Estimasi test error		**0.4248**

Eksplorasi kedua dilakukan dengan nilai $\alpha_{awal} = 0.003$, $\alpha_{batas} = 0.001$ dan $\beta = 0.00001$. Hasil pelatihan dan validasi pada eksplorasi pertama ditampilkan pada **Tabel 4**.

Tabel 4. Hasil penggunaan LVQ, dengan $\boldsymbol{\alpha_{awal} = 0.003}$, $\boldsymbol{\alpha_{batas} = 0.001}$ dan $\boldsymbol{\beta = 0.00001}$

K-fold	Akurasi terbesar		*E*(*k*)
	Pelatihan	Validasi	(1-AkurasiVal)
1	59.02%	58.76%	0.4126
2	60.05%	56.7%	0.433
3	59.28%	51.55%	0.4845
4	60.57%	62.89%	0.3711
5	56.44%	57.73%	0.4227
	Estimasi test error		**0.4248**

Eksplorasi ketiga dilakukan dengan nilai $\alpha_{awal} = 0.003$, $\alpha_{batas} = 0.00001$ dan $\beta = 0.00001$. Hasil pelatihan dan validasi pada eksplorasi pertama ditampilkan pada **Tabel 5**.

Tabel 5. Hasil penggunaan LVQ, dengan $\boldsymbol{\alpha_{awal} = 0.003}$, $\boldsymbol{\alpha_{batas} = 0.00001}$ dan $\boldsymbol{\beta = 0.00001}$

K-fold	Akurasi terbesar		*E*(*k*)
	Pelatihan	Validasi	(1-AkurasiVal)
1	60.05%	58.76%	0.4126
2	60.31%	56.7%	0.433
3	60.31%	51.55%	0.4845
4	60.57%	62.89%	0.3711
5	57.73%	58.76%	0.4124
	Estimasi test error		**0.4227**

Eksplorasi keempat dilakukan dengan nilai $\alpha_{awal} = 0.004$, $\alpha_{batas} = 0.00001$ dan $\beta = 0.00001$. Hasil pelatihan dan validasi pada eksplorasi pertama ditampilkan pada **Tabel 6**.

Tabel 6. Hasil penggunaan LVQ, dengan $\boldsymbol{\alpha_{awal} = 0.004}$, $\boldsymbol{\alpha_{batas} = 0.00001}$ dan $\boldsymbol{\beta = 0.00001}$

K-fold	Akurasi terbesar		*E*(*k*)
	Pelatihan	Validasi	(1-AkurasiVal)
1	59.79%	58.76%	0.4126
2	60.05%	56.7%	0.433
3	61.34%	51.55%	0.4845
4	60.57%	62.89%	0.3711
5	57.22%	58.76%	0.4124
	Estimasi test error		**0.4227**

Eksplorasi kelima dilakukan dengan nilai $\alpha_{awal} = 0.004$, $\alpha_{batas} = 0.000001$ dan $\beta = 0.00001$. Hasil pelatihan dan validasi pada eksplorasi pertama ditampilkan pada **Tabel 7**.

Tabel 7. Hasil penggunaan LVQ, dengan $\boldsymbol{\alpha_{awal} = 0.004}$, $\boldsymbol{\alpha_{batas} = 0.000001}$ dan $\boldsymbol{\beta = 0.00001}$

K-fold	Akurasi terbesar		*E*(*k*)
	Pelatihan	Validasi	(1-AkurasiVal)
1	60.05%	58.76%	0.4126
2	60.31%	56.7%	0.433
3	61.34%	51.55%	0.4845
4	60.82%	62.89%	0.3711
5	58.25%	58.76%	0.4124
	Estimasi test error		**0.4227**

4.4 Perbandingan Estimasi *Test Error* PNN dan LVQ

Nilai estimasi *test error* yang didapatkan dari penggunaan PNN dan LVQ dibandingkan, dan ditentukan metode yang terbaik dengan dilihat dari nilai estimasi yang terkecil. Nilai-nilai estimasi *test error* yang didapatkan dari setiap metode dan eksplorasi diperlihatkan pada **Tabel 8**.

Tabel 8. Perbandingan nilai estimasi *test error*

No	Metode	Parameter pelatihan	Estimasi *test error*
1	PNN	$\alpha = [0.001, 0.07]$	0.4495
2	LVQ	$\alpha = [0.0001, 0.003]$	0.4248
3	LVQ	$\alpha = [0.001, 0.003]$	0.4248
4	LVQ	$\alpha = [0.00001, 0.003]$	0.4227
5	LVQ	$\alpha = [0.00001, 0.004]$	0.4227
6	LVQ	$\alpha = [0.000001, 0.003]$	0.4227

Dilihat dari **Tabel 8**, metode LVQ menghasilkan estimasi *test error* yang lebih kecil jika dibandingkan dengan PNN. LVQ dengan lima buah eksplorasi yang dilakukan menghasilkan nilai estimasi *test error* tertinggi sebesar 0.4248 dan nilai estimasi *test error* terendah sebesar 0.4227. Nilai estimasi *test error* tertinggi dari LVQ ternyata masih lebih rendah dibandingkan dengan nilai estimasi *test error* dari PNN sebesar 0.4495. Dengan demikian dapat disimpulkan bahwa metode LVQ lebih baik daripada metode PNN dalam identifikasi jenis bercak pada daun cabai berdasarkan citra daun tersebut.

5. KESIMPULAN

Kesimpulan yang didapatkan pada penelitian ini adalah:

a. Besarnya akurasi validasi dari masing-masing metode dengan menggunakan *k-fold* CV yang tertinggi sebesar 62.89%. Besarnya akurasi ini lebih kecil daripada penelitian yang dilakukan sebelumnya. Hal ini diakibatkan oleh penggunaan *k-fold* CV yang membentuk dataset tiap *fold* secara *random* dari keseluruhan dataset.
b. Nilai estimasi *test error* terendah yang dihasilkan oleh LVQ adalah sebesar 0.4227 dengan akurasi validasi tertinggi sebesar 62.89%. Sementara yang dihasilkan oleh PNN adalah sebesar 0.4495 dengan akurasi validasi tertinggi sebesar 59.79%. Karena nilai estimasi *test error* LVQ lebih rendah dibandingkan dengan PNN, maka dapat disimpulkan bahwa LVQ lebih baik digunakan pada kasus identifikasi bercak pada daun cabai berdasarkan citranya dibandingkan dengan PNN.

DAFTAR PUSTAKA

Chen, C.-H., Lee, J.-D. & Lin, M.-C. 2000. Classification of Underwater Signals Using Neural Networks. *Tamkang Journal of Science and Engineering*. Vol 3, No1 : 31-48.

Fausett, L. 1994. *Fundamentals of Neural Networks : Architectures, Algorithms, and Applications*. New Jersey : Prentice-Hall.

George, Y.M., Elbagoury, B.M., Zayed, H.H. & Roushdy, M.I. 2012. Breast Fine Needle Tumor Classification using Neural Networks. *IJCSI*. Vol 9, Issue 5, No 2 : 247-256.

Haykin, S. 1999. *Neural Networks a Comprehensive Foundation Second Edition*. Singapore : Pearson Education.

James, G., Witten, D., Hastie, T. & Tibshirani, R. 2013. *An Introduction to Statistical Learning with Applications in R*. New York : Springer.

Liu, J., Zheng, K., Zhang, H. & Peng, D. 2013. A Comparative Research on Power Transformer Fault Diagnosis Based on Several Artificial Neural Networks. *Journal of Computational Information Systems*. Vol 9, No 18 : 7501-7508.

Mao, K.Z., Tan, K.-C. & Ser, W. 2000, Probabilistic Neural-Network Structure Determination for Pattern Classification. *IEEE Transactions on Neural Networks*, Vol 11, No 4 : 1009–1016.

Permadi, J. 2016. Identifikasi Penyakit Cabai Berdasarkan Gejala Bercak Daun dan Penampakan Conidia Menggunakan *Probabilistic Neural Network*. *Tesis*. Yogyakarta : Ilmu Komputer FMIPA UGM.

Theodoridis, S. & Koutroumbas, K. 2009. *Pattern Recognition Fourth Edition*. Burlington : Academic Press.

Biodata Penulis

Jaka Permadi, mendapatkan gelar S.Si. dari Universitas Tanjungpura Pontianak dalam bidang matematika pada tahun 2011 dan gelar M.Cs. dari Universitas Gadjah Mada Yogyakarta dalam bidang Ilmu Komputer pada tahun 2016. Pekerjaan penulis saat ini adalah sebagai dosen di Program Studi Teknik Informatika, Politeknik Negeri Tanah Laut. Ketertarikan penelitian pada bidang kecerdasan buatan, pengolahan citra, data mining, jaringan syaraf tiruan dan keamanan data.