PCA-RBPNN UNTUK KLASIFIKASI DATA MULTIVARIAT DENGAN ORTHOGONAL LEAST SQUARE (OLS)

Oni Soesanto
Program Studi Matematika
Universitas Lambung Mangkurat
Jl. Jend. A. Yani km 35, 8 Banjarbaru
Email: onis73@yahoo.com

ABSTRAK

Penelitian ini akan mengkaji tentang *PCA-RBPNN* (*Principal Component Analysis- Radial Basis Probabilistik Neural Network*) untuk klasifikasi data multivariat. *Analisis Komponen Utama (PCA)* telah banyak dikenal dalam statistika sebagai metode yang digunakan untuk mereduksi dimensi input pada data multivariat dengan meminimalkan kehilangan informasi. Dalam hal ini, PCA digunakan untuk mereduksi dimensi input pada jaringan syaraf RBPNN. Proses clustering dan inisialisasi center dilakukan dengan *Self-Organizing Map (SOM)*. Untuk penentuan bobot selama proses pembelajaran pada jaringan RBPNN, menggunakan algorithma *Orthogonal Least Square (OLS)*. Selanjutnya metode PCA-RBPNN digunakan untuk klasifikasi data multivariat. Akurasi klasifikasi PCA-RBPNN disimulasikan dan dibandingkan dengan model RBPNN biasa.

Kata Kunci: *RBPNN, PCA-RBPNN, SOM, Orthogonal Least Square (OLS)*

1. PENDAHULUAN

Radial Basis Function Neural Network (RBFNN) dikenal sebagai model neural network yang handal dan banyak digunakan pada masalah peramalan (forecasting) dan klasifikasi. Peramalan dengan neural network melibatkan beberapa proses yaitu clustering untuk karakteristik data time series, klasifikasi hubungan antara data time series dengan kriteria deskripsinya dan klasifikasi prediksi pada suatu data time series [9],[10]. Pengembangan model RBFNN telah banyak dilakukan dengan banyak cara seperti penggabungan, optimalisasi struktur jaringan, penetuan center [4], [7],[12].

Analisis Komponen Utama atau *Principal Component Analysis* (PCA) merupakan metode dalam statistika yang digunakan untuk mereduksi dimensi input dengan kehilangan informasi yang minimum. Beberapa peneliti telah menggabungkan PCA dengan neural network [1]. menggunakan model PCA neural pada preprocessing untuk pemisahan noise pada signal. PCA juga dapat dikombinasikan dengan neural network seperti RBFNN untuk mereduksi variabel input pada masalah peramalan [3], [6]. Pada masalah clustering dan klasifikasi RBFNN digabungkan dengan *Self-Orginizing Map* (SOM) dan *Probabilistic Neural Network* (PNN) disebut dengan *Radial Basis Probabilistic Neural Network* (RBPNN) dengan tujuan untuk meningkatkan kemampuannya [7], [8].

Proses training RBFNN menggunakan struktur khusus yang melibatkan dimensi tinggi pada hidden layer dan nonlinier input pada hidden layernya untuk mensimulasi sejumlah training data yang memerlukan akurasi. Dengan struktur RBFNN yang khusus tersebut maka seringkali menimbulkan permasalahan karena hidden layernya terlalu besar, sehingga diperlukan penyederhanaan jaringan [11]. Untuk itu beberapa metode seperti *Recursive Least Square, Orthogonal Least Square, Genetic Algorithm* digunakan untuk optimalisasi bobot pada RBFNN. Sedangkan untuk penentuan center banyak digunakan teknik clustering seperti *K-means Clustering* dan *Self-Orginizing Map (SOM)* neural [5].

Penelitian ini akan mengkaji model PCA-RBPNN untuk permasalahan klasifikasi data multivariat. Pada model ini, PCA digunakan untuk preprosessing RBPNN, sedangkan SOM digunakan untuk klustering dan inisialisasi center untuk RBPNN. Untuk optimalisasi bobot jaringan RBPNN digunakan *Orthogonal Least Square Algorithm* (OLSA). Selanjutnya model PCA-RBPNN ini akan dibandingkan dengan model RBPNN lain.

2. TINJAUAN PUSTAKA

2.1 METODE PCA-RBPNN

Reduksi Dimensi Input dengan PCA

Principal Componen Analysis (PCA) telah secara luas digunakan dalam mereduksi dimensi dari data multivariat, kompresi data, pengenalan pola dan analisis statistik. Algorithma PCA didesain untuk mengaproksimasi suatu pola pada ruang dimensi tinggi dengan sub ruang dimensi rendah yang merentang dengan *principal eigen vektor* dari data matriks kovarian. Dengan cara ini distribusi data dapat dinyatakan dan direkonstruksi kembali dengan principal eigen vektor dan eigen valuenya.

Proses awal PCA dikenal dengan feature extraction merupakan proses memetakan dari ruang dimensi tinggi ke ruang dimensi yang lebih rendah dengan kehilangan informasi yang minimum. Untuk ruang data sangat besar yang memiliki dimensi n, metode learning pada *neural network* (NN) secara efisien bisa digunakan untuk menyelesaikan persoalan pada PCA. Beberapa model NN dan algorithma learning telah banyak digunakan saat ini, baik model linier ataupun non linier, dengan algorithma learning yang supervised ataupun yang unsupervised [1].

Misalkan diberikan matriks data $X=[x_1, x_2,...,x_p]$ dengan matriks kovarian Σ yang memiliki nilai eigen $\lambda_1 \geq \lambda_2 \geq \lambda_3 ... \geq \lambda_p \geq 0$. Diberikan kombinasi linier:

$$\begin{aligned} Y_1 &= a_1'X = a_{11}x_1 + a_{12}x_2 + ... + a_{1p}x_p \\ Y_2 &= a_2'X = a_{21}x_1 + a_{22}x_2 + ... + a_{2p}x_p \\ &\ldots\ldots \\ Y_p &= a_p'X = a_{p1}x_1 + a_{p2}x_2 + ... + a_{pp}x_p \end{aligned} \quad (1)$$

dimana $Var(Y_i) = a_1' \Sigma a_i$, i = 1,2,...,P, $Cov(Y_i , Y_k) = a_1' \Sigma a_k$, i, k = 1,2,...,P

Komponen utama adalah kombinasi linear Y_1, Y_2, ..., Y_p yang tidak berkorelasi dengan varian terbesar. Komponen utama pertama adalah kombinasi linear dari $a_1'X$ dengan varian $Var(a_i'X)$ terbesar pada $a_1'a_1 = 1$ dan komponen utama kedua adalah kombinasi linear dari $a_1'X$ dengan varian $Var(a_2'X)$ terbesar pada $a_2'a_2 = 1$ dan $Cov(a_1'X, a_2'X)=0$. Untuk komponen utama ke-i adalah kombinasi linear dari $a_i'X$ dengan varian $Var(a_i'X)$ terbesar pada $a_i'a_i = 1$ dan $Cov(a_i'X,a_k'X)=0$, (k<i) [4].

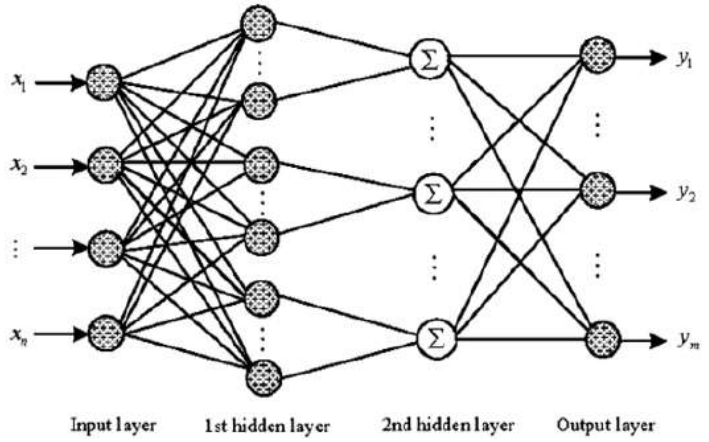
Proyeksi pada PCA adalah representasi himpunan data X ke dalam bentuk vektor eigen orthonormal dari matriks kovarian data X. Matriks kovarian merupakan korelasi antara variabel-variabel dalam himpunan data X. PCA merupakan proses mendapatkan vektor eigen orthonormal dari matriks kovarian sebagai basis untuk ditransformasi ke ruang data yang baru. Vektor eigen dapat dikatakan sebagai basis asli untuk multi dimensi data X. Nilai eigen terbesar dari matriks kovarian merupakan korelasi terkecil antar variabel dalam ruang data. Selanjutnya PCA akan mencari proyeksi variabel-variabel yang tidak berkorelasi.

2.2 RBPNN

Radial Basis Function Neural Network (RBFNN) merupakan jaringan *Multilayered Feedforward Neural Network* (MFNN) yang terdiri dari tiga lapisan, yaitu input, hidden

layer dan output. Jaringan ini merupakan pemetaan fungsi tak linier multidimensional yang tegantung pada jarak antara vektor input dan vektor center. RBFNN merupakan metode yang handal untuk permasalahan regresi dan klasifikasi. Untuk training data pada RBFNN menggunakan struktur khusus yang melibatkan dimensi tinggi pada hidden layer dan nonlinier input pada hidden layernya untuk mensimulasi sejumlah training data yang memerlukan akurasi [11].

Model *Probabilistic Neural Network* (PNN) telah banyak digunakan pada berbagai bidang aplikasi sepeti clustering dan klasifikasi pada forecasting [10], dan klasifikasi pada image recognition [7]. PNN dikonstruksi berdasarkan teori probabilitas klasik seperti klasifikasi Bayesian dan estimator klasik fungsi kepadatan peluang untuk membentuk jaringan syaraf pada klasifikasi pola. Pada permasalahan klasifikasi, PNN melakukan training dengan menghitung jarak input pada lapisan pertama yang mengindikasikan kedekatan dengan vektor input training. Lapisan kedua merupakan kontribusi setiap klas input yang menghasilkan net output vektor probabilitas.



Gambar 1. Arsitektur RBPNN

Model RBPNN merupakan model jaringan yang menggabungkan beberapa keunggulan RBFNN dan PNN. Pada Gambar 1 adalah jaringan RBPNN yang terdiri dari empat layer, yaitu layer input, dua hidden layer tersembunyi dan layer output. Layer tersembunyi pertama merupakan layer dengan proses non linier, yang secara umum terdiri dari center tersembunyi yang ditentukan dari training input. Layer tersembunyi kedua merupakan penjumlahan dari output layer pertama, dan secara umum memiliki ukuran yang sama dengan layer output. Bobot antara layer tersembunyi pertama dan layer tersembunyi kedua, merupakan bobot konstan. Artinya, bobot diatur tetap sehingga tidak diperlukan learning. Lapisan terakhir dari RBPNN adalah lapisan output. Secara matematis, RBPNN dengan vektor input *x* akan menghasilkan nilai aktual untuk neuron output ke-i y_i^{α} yang dinyatakan sebagai persamaan berikut:

$$y_i^{\alpha} = \sum_{k=1}^{M} w_{ik} h_k(x) \text{ dimana } h_k(x) = \sum_{i=1}^{nk} \phi_i(x, c_{ki}) = \sum_{i=1}^{nk} \phi_i(\|x - c_{ki}\|_2), k = 1,2,..,M \qquad (2)$$

Sedangkan untuk $\phi_i(\bullet)$ adalah fungsi kernel yang umumnya fungsi kernel Gaussian

$$\phi_i(\|x - c_{ki}\|_2) = \exp(-\frac{\|x - c_{ki}\|_2^2}{\sigma_i^2}) \qquad (3)$$

Dimana σ_i adalah parameter fungsi kernel Gaussian. Metode tercepat dan sederhana untuk penentuan spread RBPNN yaitu dengan menggunakan pemilihan σ_i yang diberikan pada

$$\sigma = \frac{d_{maks}}{\sqrt{K}} \tag{4}$$

dengan d_{maks} adalah jarak Euclid maksimal dari sehimpunan pelatihan dan K adalah jumlah total sehimpunan pelatihan [2].

Metode penentuan center untuk RBFNN dapat juga dilakukan pada RBPNN. Salah satu metode yang digunakan untuk menentukan center RBFNN yaitu dengan menggunakan teknik clutering. Proses awal yang perlu dilakukan pada RBPNN adalah proses clustering untuk menentukan inisialisasi center RBFNN. Salah satu metode yang digunakan untuk menentukan center RBFNN yaitu dengan menggunakan teknik clustering.

2.3 Inisialisasi Center dengan Self-Organizing Map (SOM)

Klustering merupakan proses membagi sehimpunan objek menjadi sejumlah subset yang disebut klas atau kluster. Klas dibentuk oleh objek-objek yang memiliki sifat yang sama. *Self- Orginizing Map* (SOM) merupakan metode unsupervised learning yang banyak digunakan dalam *artificial neural network* (ANN). Secara umum SOM dianggap memiliki fungsi yang hampir sama dengan metode klustering dalam statistika seperti *k-mean*. Kemampuan SOM sedikit lebih dari k-mean dalam mengenali kluster yang secara statistik tidak mampu dikenali karena beberapa sifat data seperti tidak adanya korelasi dan tidak terdistribusi normal [6].

SOM merupakan pemetaan multi dimensi data kedalam satu atau dua dimensi. SOM terdiri dari dua layer neuron, layer input dan layer kompetitif [8]. Pada tahap training, secara random akan dipilih dari sampel vektor x(t) dari sekumpulan data training. Jarak antara x(t) dengan setiap vektor dihitung dan vektor unit pemenang terpilih c diperoleh dengan persamaan:

$$c = \arg\min_i \|x(t) - w_i\|, \ i \in 1,..., M \tag{5}$$

Sekumpulan titik disekitar unit pemenang dinotasikan dengan *Nc*. Fungsi kernel yang melingkupi unit pemenang c pada saat t dinotasikan dengan $h_{ic}(t)$, yaitu fungsi *nonincreasing* dari waktu dan jarak antara unit ke i dari unit pemenang c. Fungsi kernel unit pemenang c dapat dinyatakan dengan fungsi Gaussian:

$$h_{ic}(t) = \exp\left(\frac{\|r_i - r_c\|^2}{2\sigma(t)^2}\right), \ i \in N_c \tag{6}$$

dimana r_i adalah posisi unit ke i, r_c adalah posisi unit pemenang pada layer output dan $\sigma(t)$ adalah lebar (width) kernel pada saat t. Selanjutnya unit pemenang dan persekitarnnya akan dikeluarkan dari vektor input dan vektor bobot diupdate dengan persamaan:

$$w_i(t+1) = \begin{cases} w_i(t) + \alpha(t) h_{ic}(t)(x(t) - w_i(t)), \forall i \in N_c \\ w_i(t), \text{ lainnya} \end{cases} \tag{7}$$

2.4 Penentuan Bobot dengan Orthogonal Least Square (OLS)

Secara umum, komputasi bobot output pada jaringan feedforward berdasarkan fungsi biaya error atau mengacu pada fungsi targetnya, yaitu fungsi pada struktur jaringan dan bobot. Untuk RBPNN, error pada fungsi costnya didefinisikan sebagai berikut [5]:

$$J(W) = \|Y - HW\|_F^2 \tag{8}$$

dimana: $\|\bullet\|_F^2$ adalah norm Frobenius, Y matriks output, H matriks output pada hidden layer, W matriks bobot antara hidden layer dan layer output.

Dengan menggunakan dekomposisi orthogonal, matriks H pada persamaan (8) merupakan matriks berukuran *mxn* dengan vektor kolom bebas linier, Q adalah matriks orthogonal berukuran NxN yang memenuhi $QxQ^T=Q^TxQ=I$, R adalah matriks segitiga atas berukuran MxM

Karena Q matriks orthogonal maka diperoleh:

$$Y = Q\begin{bmatrix}\hat{Y}\\ \tilde{Y}\end{bmatrix} \tag{9}$$

Norm-F dikonversi berdasarkan transformasi orthogonal, diperoleh fungsi cost sebagai berikut

$$J(W) = \left\|\hat{Y} - HW\right\|_F^2 + \left\|\tilde{Y}\right\|_F^2 \tag{10}$$

Selanjutnya dengan meminimasi fungsi error J(W) diperoleh bobot $W = R^{-1}\hat{Y}$.

3. HASIL DAN PEMBAHASAN

3.1 Data

Data untuk pengujian model PCA-RBPNN diambil dari PT Jasa Tirta I Jawa Timur yang berpusat di Malang yaitu data harian Ketersediaan Air di Bendungan Sutami tahun 2007-2009. Ketersediaan air di bendungan Sutami dipengaruhi oleh beberapa faktor antara lain inflow, curah hujan dan debit aliran yang menuju aliran bendungan. Dalam paper ini data pengukuran yang diambil terdiri dari 8 parameter pengukuran yaitu inflow bendungan Sutami, debit aliran (Gadang, Tawangrajeni) dan curah hujan (Sutami, Sengguruh, Poncokusumo, Tangkil, Wagir).

Pada paper ini, untuk tahap training digunakan data tahun 2007 dan 2008 sebanyak 5848 titik data, sedangkan untuk tahap testing digunakan data tahun 2009 sebanyak 2920 titik data.

3.2 Reduksi Variabel Input dengan PCA

Dalam kajian ini, PCA digunakan sebagai proses awal untuk mengetahui korelasi setiap variabel pengukuran . Tujuan PCA adalah mereduksi dimensi variabel input menjadi dimensi yang lebih kecil dengan kehilangan infornasi minimum, dimana setiap komponen utama PC yang terbentuk tidak berkorelasi satu sama lain. Berdasarkan data pengukuran yang diambil dari tahun 2007-2009, terdapat 8 parameter pengukuran misalkan x_1, x_2, ..., x_8 yang masing-masing merepresentasikan data pengukuran untuk inflow Sutami, debit sungaiTawangrajeni, debit sungai Gadang, curah hujan Poncokusumo, curah hujan Sutami, curah hujan Wagir, curah hujan Sengguruh dan curah hujan Tangkil.

Setiap kelompok data memiliki komponen utama, tetapi proses PCA akan bekerja secara baik jika kelompok data tersebut berdistribusi Gaussian. Untuk data berdimensi tinggi diasumsikan berdistribusi Gaussian. Untuk itu pada PCA selalu menggunakan bentuk deviasi mean karena setiap data telah dikurangi dengan meannya atau sering disebut dengan *zero-mean*. Distribusi probabilitas zero mean yang secara penuh menggambarkan variannya merupakan distribusi Gaussian. Dengan menghitung nilai mean dan standar deviasi masing-masing parameter maka dapat ditentukan matriks varian-kovariannya.

Setelah mendapatkan matriks kovarian C maka akan dilakukan diagonalisasi matriks kovariannya. Untuk proses diagonalisasi, PCA mengasumsikan semua vektor basis p_1,..,p_m adalah orthonormal, atau dengan kata lain P adalah matriks orthonormal. Pada

PCA, matriks orthonormal P digunakan untuk menyamakan rotasi pensejajaran basis dengan variansi maksimalnya. Secara sederhana proses tersebut dilakukan dengan normalisasi nilai eigen terbesar dalam ruang dimensi m sepanjang nilai variansi pada matriks kovariannya. Vektor yang dihasilkan kemudian disimpan sebagai p_i. Proses dilanjutkan sebanyak m hingga didapatkan keseluruhan p_i dimana i=1,..,m. Hasil p_i yang telah terurut merupakan komponen utama dari X, dapat dilihat pada Tabel 1.

Tabel 1. Nilai PC untuk masing-masing parameter, nilai eigen dan variabilitas.

	PC 1	**PC 2**	**PC 3**	**PC 4**	**PC 5**	**PC 6**	**PC 7**	**PC 8**
Inflow Sutami (x_1)	-0.0834	**-0.6577**	-0.6278	0.4062	0.0308	-0.013	0.0035	0.0148
D Tawangrajeni (x_2)	**-0.991**	0.1337	-0.0069	0.0022	0.0017	0.0043	0.0031	0.0026
D Gadang (x_3)	-0.1014	-0.7129	0.6833	-0.118	-0.0134	-0.0145	0.003	-0.0178
CH Poncokusumo (x_4)	-0.0152	-0.086	-0.1686	-0.4275	0.2917	-0.4014	-0.7216	-0.1192
CH Sutami (x_5)	-0.0059	-0.0597	-0.0824	-0.2385	-0.2233	0.0817	-0.1213	0.9282
CH Wagir (x_6)	-0.0077	-0.0873	-0.1277	-0.3405	0.4008	0.8306	-0.0435	-0.0868
CH Sengguruh (x_7)	-0.0093	-0.0808	-0.1608	-0.4347	0.4271	-0.3729	0.6698	0.0919
CH Tangkil (x_8)	-0.0181	-0.1271	-0.2482	-0.5259	-0.7217	0.0529	0.1182	-0.3282
Nilai Eigen	3.6875	1.1898	0.8409	0.7588	0.5583	0.2538	0.3948	0.3161
Variabilitas (%)	81.5826	13.4744	2.4831	1.2282	0.5103	0.3462	0.1979	0.1772

Dari Tabel 1, untuk PC ke-1 dan ke-2 dengan nilai eigen 3.6875 dan 1.1898 masing-masing mewakili 81.5826% dan 13.4744% dari seluruh variabilitas, secara kumulatif kedua komponen utama menyatakan 95.0571%. dari total variabilitas. Hal ini berarti apabila delapan variabel asli (x_1, x_2, ..., x_8) direduksi menjadi 2 variabel, maka kedua variabel pengganti tersebut dapat menjelaskan 95.0571% dari total variabilitas delapan variabel asli.

Selanjutnya kedua variabel pengganti (misalkan y_i ,i=1,2) didefinisikan sebagai pengganti kedelapan variabel asli (x_i ,i=1,...,8) . Berdasarkan komponen utama yang bersesuaian dengan yi maka sampel data pada setiap variabel asli xi diproyeksikan ke variabel pengganti, dapat dinyatakan sebagai berikut:

$$Y = \begin{bmatrix} -0.0834 & -0.9910 & -0.1014 & -0.0152 & -0.0059 & -0.0077 & -0.0181 & -0.0093 \\ -0.6577 & 0.1337 & -0.7129 & -0.0860 & -0.0597 & -0.0873 & -0.1271 & -0.0808 \end{bmatrix} X$$

dimana $Y=[y_1,y_2]^T$ dan $X=[x_1, x_2, ...,x_8]^T$

Hasil proyeksi (misalkan y_{ij}, i=1,2, j=1,..,N dengan N=jumlah sampel data) selanjutnya digunakan sebagai input pada inisialisasi center dan clustering dengan SOM dan klasifikasi pada RBPNN. Dengan demikian proses PCA memberikan keuntungan yaitu mereduksi variabel input yang secara langsung akan menyederhanakan struktur jaringan RBPNN.

3.3 Klasifikasi pada RBPNN

Pada proses SOM secara random urutan center awal dipilih dari vektor input y_{ij} sebagai inisialisasi centernya. Dengan sejumlah N sampel data maka terdapat N permutasi urutan center awal. Jumlah cluster ditentukan berdasarkan jumlah komponen utama yang dihasilkan pada PCA. Proses akhir SOM akan menghasilkan center (c_{ij}, i,j=1,2) dan cluster (T_i, i=1,..,N) dari keseluruhan data sampel. Center c_{ij} dan cluster T_i selanjutnya digunakan sebagai inisialisasi center dan target RBPNN untuk proses klasifikasi y_{ij}.

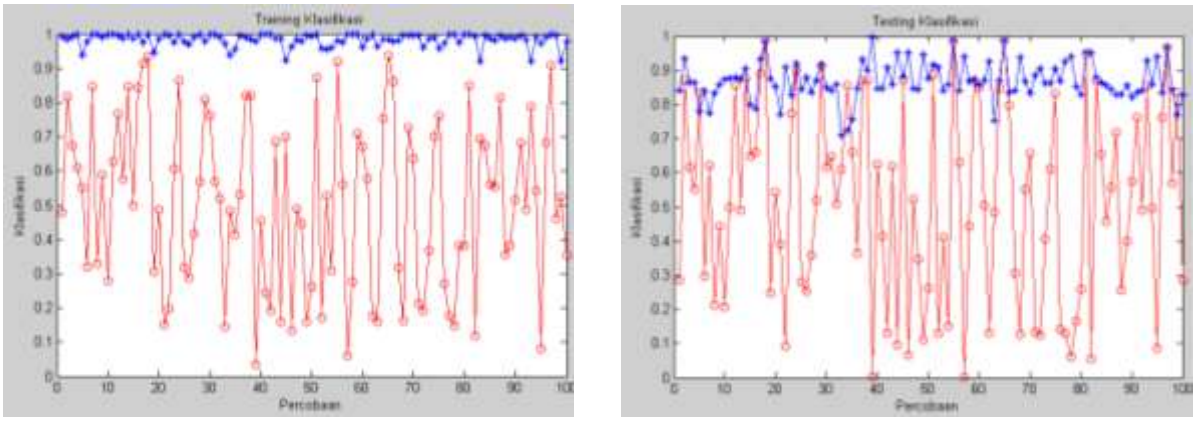
Untuk menguji model digunakan data debit bendungan Sutami tahun 2007-2009 dengan data training sebanyak 5848 titik dat (tahun 2007-2008) dan data testing sebanyak 2920 titik data (tahun 2009). Proses pengujian dilakukan sebanyak 100 kali dengan iterasi SOM sebanyak 10 epoch x 1096 iterasi dengan rata-rata error SOM sebesar 11.8198 %.

Tabel 2. Perbandingan kemampuan klasifikasi

Percobaan	RBPNN				PCA-RBPNN			
	Training	CPU time (dt)	Testing	CPU time (dt)	Training	CPU time (dt)	Testing	CPU time (dt)
1	0.478796	0.125000	0.284932	0.187500	0.993160	0.140625	0.841313	0.031250
2	0.816689	0.031250	0.871233	0.031250	0.983584	0.078125	0.932969	0.015625
3	0.673051	0.015625	0.616438	0.015625	0.993160	0.062500	0.863201	0.015625
4	0.608755	0.031250	0.550685	0.031250	0.998632	0.062500	0.864569	0.031250
5	0.549932	0.015625	0.835616	0.031250	0.935705	0.046875	0.778386	0.015625
6	0.320109	0.015625	0.295890	0.031250	0.976744	0.031250	0.839945	0.015625
7	0.767442	0.015625	0.854795	0.015625	0.995896	0.031250	0.878249	0.015625
8	0.574555	0.015625	0.490411	0.015625	0.986320	0.015625	0.871409	0.015625
9	0.845417	0.015625	0.876712	0.015625	0.998632	0.031250	0.905609	0.015625
10	0.497948	0.015625	0.646575	0.015625	0.986320	0.031250	0.800274	0.015625
.....	.....	.....	.....	.....	.....	.....	.....	.....
98	0.459644	0.031250	0.569863	0.046875	0.995896	0.046875	0.842681	0.015625
99	0.525308	0.031250	0.835616	0.046875	0.919289	0.062500	0.771546	0.015625
100	0.352941	0.015625	0.284932	0.046875	0.976744	0.078125	0.830369	0.015625
Mean	**0.505253**	**0.022344**	**0.513589**	**0.033125**	**0.982818**	**0.052969**	**0.869097**	**0.012344**
(%)	**50.525310**	**2.234375**	**51.358905**	**3.312500**	**98.281798**	**5.296875**	**86.909705**	**1.234375**

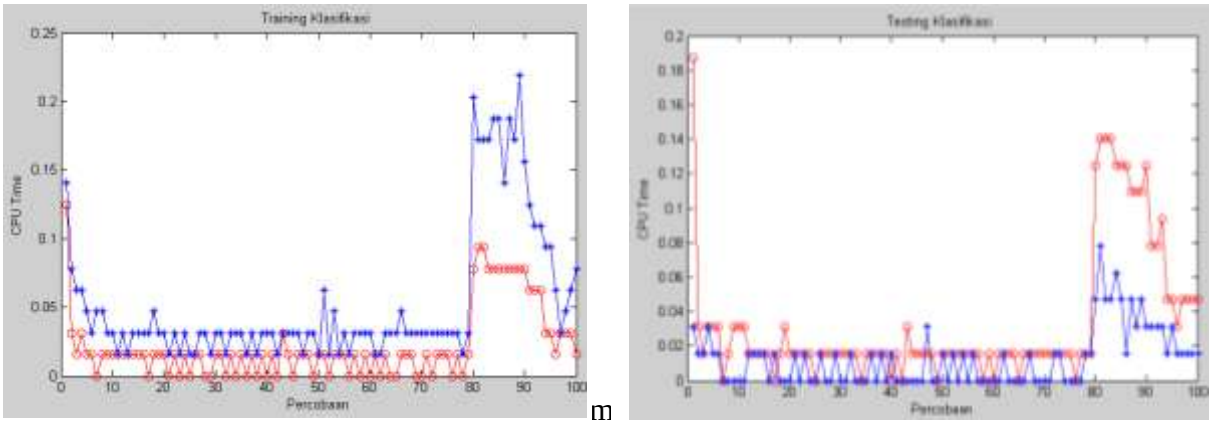
Dari Tabel 2. dapat diketahui untuk proses training RBPNN kemampuan klasifikasi rata-rata sebesar 50.5253% dengan CPU time rata-rata 0.02344 detik, sedangkan untuk training PCA-RBPNN kemampuan klasifikasi rata-rata sebesar 98.2818% dengan CPU time rata-rata 0.052969 detik. Pada proses testing RBPNN kemampuan klasifikasi rata-rata sebesar 51.3589% dengan CPU time rata-rata 0.03313 detik, sedangkan untuk testing PCA-RBPNN kemampuan klasifikasinya sebesar 89.9097% dengan CPU time rata-rata 0.01234 detik.

Hal ini berarti kemampuan klasifikasi pada proses training dan testing PCA-RBPNN jauh lebih baik dari RBPNN. Dari CPU time rata-rata pada proses training, RBPNN sedikit lebih baik dari RBPNN, namun pada proses training PCA-RBPNN masih lebih baik, hal ini disebabkan karena kemampuan klasifikasinya jauh lebih besar dibandingkan dengan RBPNN. Hasil simulasi kemampuan klasifikasi dan CPU time untuk model PCA-RBPNN dapat dilihat pada Gambar 2. dan Gambar 3.



(a) Training (b) Testing

Gambar 2. Perbandingan kemampuan klasifikasi antara RBPNN (^{0}) dan PCA-RBPNN (*)



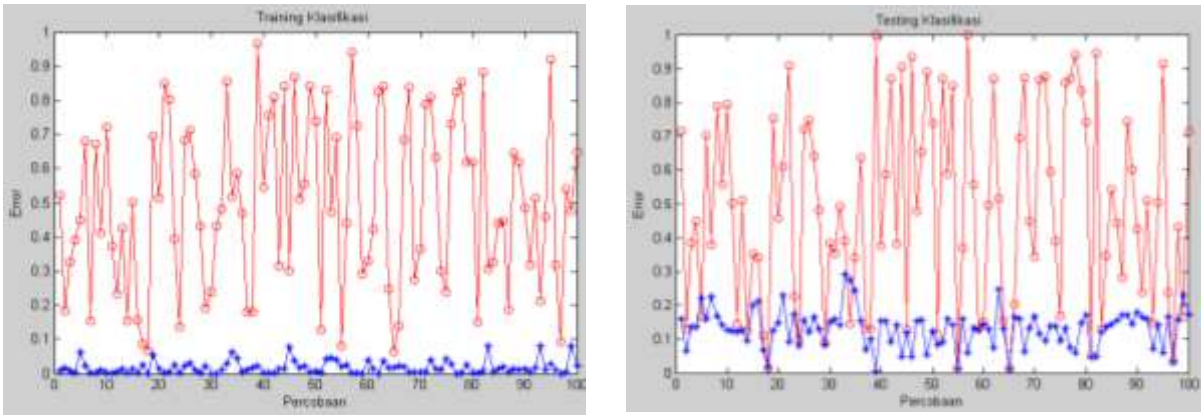
(b) Training (a) Testing

Gambar 3. Perbandingan CPU time antara RBPNN (^{o}) dan PCA-RBPNN (*)

RBPNN dengan rata-rata error klasifikasi masing-masing pada proses training sebesar 1.7182% untuk PCA-RBPNN sedangkan RBPNN sebesar 49.4747%, sedangkan pada proses training rata-rata error klasifikasi untuk PCA-RBPNN sebesar 13.0903% dan RBPNN sebesar 48.6411%. Hasil simulasi error klasifikasi pada proses training dan testing untuk model PCA-RBPNN dapat dilihat pada Gambar 4.

Tabel 3. Perbandingan error klasifikasi

Percobaan	RBPNN		PCA-RBPNN	
	Training	Testing	Training	Testing
1	0.521204	0.715068	0.006840	0.158687
2	0.183311	0.128767	0.016416	0.067031
3	0.326949	0.383562	0.006840	0.136799
4	0.391245	0.449315	0.001368	0.135431
5	0.450068	0.164384	0.064295	0.221614
6	0.679891	0.704110	0.023256	0.160055
7	0.154583	0.378082	0.002736	0.225718
8	0.671683	0.789041	0.001368	0.168263
9	0.411765	0.556164	0.009576	0.140903
10	0.719562	0.794521	0.001368	0.127223
.....	.....	.....	.....	.....
98	0.540356	0.430137	0.004104	0.157319
99	0.474692	0.164384	0.080711	0.228454
100	0.647059	0.715068	0.023256	0.169631
Mean	**0.494747**	**0.486411**	**0.017182**	**0.130903**
(%)	**49.474690**	**48.641095**	**1.718202**	**13.090295**



(a) Training (b) Testing

Gambar 4. Perbandingan error klasifikasi antara RBPNN (^{o}) dan PCA-RBPNN (*)

Dengan demikian dapat dikatakan secara umum bahwa kemampuan klasifikasi model PCA-RBPNN jauh lebih akurat dibandingkan dengan model RBPNN. Hal ini disebabkan pada model PCA-RBPNN bobot jaringan dioptimalkan dengan dekomposisi orthogonal sehingga bobot jaringan lebih cepat konvergen.

4. KESIMPULAN

Pada model PCA-RBPNN, *Principal Component Analysis* (PCA) tidak hanya mereduksi dimensi input pada RBPNN namun juga memberikan keuntungan dengan orthogonlisasi data asli agar supaya struktur jaringan RBPNN lebih sederhana. Hasil percobaan dengan data multivariat, yaitu data Ketersediaan Air di Bendungan Sutami tahun 2007-2009 dengan delapan parameter pengukuran dihasilkan dua komponen utama dengan variabilitas 95.0571%.

Hal ini juga berakibat proses clustering , inisialisasi center dengan SOM dan penentuan bobot center pada layer hidden dengan *Orthogonal Least Square* akan lebih cepat. Hasil percobaan memberikan hasil rata-rata akurasi klasifikasi model PCA-RBPNN sebesar 98.2818% untuk training 86.909705% untuk testing dengan CPU time 5.2969 detik. Dari hasil simulasi dapat disimpulkan bahwa model PCA-RBPNN memiliki akurasi dan kecepatan yang lebih baik dibandingkan dengan model RBPNN.

5. DAFTAR PUSTAKA

[1] Diamantaras, K.I. dan Papadimitrious, T. 2009. Applying PCA neural models for blind separation signals', *Neurocomputing* 73, 3–9.

[2] Hasanuddin dan Irawan, I.M. 2009. The study of Sensitivity of Radial Basis Probabilistic Neural Network, *International Conference on Mathematics, Statistics and their Application*, 344-349

[3] Hynar, M., Burda, M. dan Sarmanova, J. 2005. Unsupervisied Clustering with Growing Self-Orginizing Neural Network - A Comparation with Non-Neural Approach, *Computer Science*, 58–68.

[4] Li, Z.X. dan Guang, H.G. 2007. Forecasting Approach for Short-term Traffic Flow based on Principal Component Analysis and Combined Neural Network, *System Engineering* 27, 167–171.

[5] Lu, W.Z., Wang, W.J., Wang, X.K., Yan, S.Y. dan Lam, J.C. 2004. Potensial Assesment of A Neural Network Model with PCA/RBF Approach Forecasting Pollutant Trends in Mong Kok Urban Air, Hong Kong', *Environmental Research* 96, 79–87.

[6] Huang, D.S. dan Zao, W.B. 2005. Determining the Center of radial Basis Probabilistic Neural Network by Recursive Orthogonal Least Square Algorithms, *Applied Mathematics and Computation* 162, 461–473.

[7] Huang, D.S. dan Du, J.X. 2008. A Constructive Structure Optimization Methodology for Radial Basis Probabilistic neural Network, *IEEE Transactions on Neural Network* 19, 2009-2115.

[8] Mu, T. dan Nandi, A.K. 2007. Breast Cancer Detection from FNA using SVM with Different Parameter Tuning System and SOM-RBF Classifier, *Journal of Franklin Institute* 344, 258-311.

[9] Ranaweera, D.K., Hubele, N.e. dan Papalexopoulos, A.D. 1995. Application of Radial Basis Function Neural Network Model for Short-Term Load Forecasting, *IEE Proc-Gener. Transm. Distrib*. 142, 45-50.

[10] Thomassey, S. dan Happiette, M. 2007. A Neural Clustering and Classification System for Sales Forecasting of New Apparel Item, *Applied Soft Computing* 7, 1177–1187.
[11] Wang, X.Z., Li, C.G., Yeung, D.S., Song, SJ dan Feng, H.M. 2008. A Definition of Partial Derivative of Random Functions and Its Application to RBFNN Sensitivit Anaysis, *Neurocomputing*, 71, 1551–1526.
[12] Yousef, R. dan Hindi, K. 2005. Locating Center Points for Radial Basis Function Network Using Instance Reduction Techniques, *Engineering and Technology* 4, 213-216.