Konsolidasi Beban Kerja Kluster *Web server* Dinamis dengan Pendekatan *Backpropagation Neural Network*

Alan S Balantimuhe, Sholeh Hadi Pramono, Hadi Suyono

***Abstract*— The increasing demand for users of WWW applications has led to a commensurate increase in the use of cluster server resources. This study examines the provision of web server resources based on server workload parameters (load average CPU). Researchers conduct research on the cluster web server that serves Academic Student Information System of Universitas Brawijaya (SIAM-UB) application. Maximum use of server resources (peak load) occurs during the student registration period, more than 65000 students will access the SIAM server simultaneously. The number of *request*s served by the server in 1 day can reach 1.7 million *request*s. In this research, the server resources is predicted to determine the optimal CPU load average in web server cluster. The server workload predictions are classified into 3 classes, namely: Min (0-2), Medium (3-6), Maximum> 7. The backpropagation neural network (BNN) method, is then used to predict the server workload class based on the following parameters such as CPU usage, memory, network (throughput) and number of IP access to SIAM cluster server. The BNN architecture with the 32 inputs, 2 hidden layers including the number of neuron h1 512 and h2 32, 3 outputs, and learning rate of 0.0001, have produced the weights that can perform the classification with the precision level of 90%, the sensitivity level of 0.9, and the accuracy level of 93%.**

***Index Terms*—Classification, Backpropagation neural network, load average, cluster web server.**

***Abstrak*--- Meningkatnya permintaan pengguna aplikasi WWW telah menyebabkan peningkatan yang sepadan dalam penggunaan sumber daya server kluster. Penelitian ini mengkaji tentang penyediaan sumber daya *web server* berdasarkan parameter beban kerja server (*load average CPU*). Data yang digunakan adalah akses terhadap *web server* yang melayani applikasi Sistem Informasi Akademik Mahasiswa Universitas Brawijaya (SIAM-UB). Penggunaan sumber daya server secara maksimal (beban puncak) terjadi pada periode registrasi mahasiswa, yaitu lebih dari 65000 mahasiswa akan mengakses server SIAM secara bersamaan. Jumlah permintaan yang dilayani server dalam 1 hari dapat mencapai 1.7juta permintaan. Pada penelitian ini, penyediaan sumber daya server diprediksi untuk mendapatkan konsolidasi beban kerja *CPU* dalam kluster *web server* yang optimal. Prediksi beban kerja server diklasifikasikan menjadi 3 kelas, yaitu: Min (0-2), Medium (3-6), Maximum >7. Metode *backpropagation neural network* (BNN) digunakan untuk memprediksi kelas konsolidasi beban kerja server berdasarkan parameter input penggunaan *CPU*, *memory*, jaringan (throughput) dan jumlah IP akses. Arsitektur BNN dengan 32 input, 2 hidden layer dengan jumlah *neuoron* h1 512; h2 32, 3 output, dan *learning rate* 0.0001, menghasilkan bobot yang mampu melakukan klasifikasi dengan tingkat precision 90%, tingkat sensitivity 0.9, dan tingkat akurasi 93%.**

***Kata Kunci*— *Backpropagation neural network*, fungsi aktivasi, *confusion matrix*, *cluster web server*, *load balance*.**

I. PENDAHULUAN

Teknologi WWW berkembang dengan sangat cepat, memainkan peran yang penting dalam kehidupan sehari-hari. Tingkat pertumbuhan jumlah akses yang menggunakan teknologi ini juga mengalami peningkatan yang sangat serius dari waktu ke waktu [1]. Isu penting dengan pertumbuhan yang terus berkembang ini sangat berkaitan dengan kinerja dari server web yang harus menyediakan akses (sumber daya) yang dapat diandalkan, prediktif, dan efisien. Tingkat permintaan jumlah akses ke *web server* terjadi secara dinamis, pada rentang waktu tertentu jumlah akses dapat tinggi dan juga sebaliknya.

Untuk menangani masalah pertumbuhan jumlah akses yang tinggi, beberapa metode telah banyak digunakan seperti dengan menggunakan pendekatan teknik *load balancing* [2] yaitu dengan mekanisme penjadwalan *round-robin* dan *source hash*, dimana setiap *request* yang datang akan diteruskan ke *real server*. Penyediaan sumber daya *real server* dilakukan secara statis dengan mempertimbangkan kebutuhan beban puncak [3]. Namun demikian, kerugian dari penyediaan server secara statis pada kondisi tertentu dapat terjadi *over-provisioning* dari sumber daya server yang ada.

Metode lainnya adalah menggunakan teknik konsolidasi dinamis dengan melakukan pemodelan vektor multidimensi (m-D). Teknik ini menjalankan algoritme dalam rentang waktu tertentu untuk

Manuskrip diterima pada tanggal 20 Desember 2017.

Alan S Balantimuhe, adalah mahasiswa Program magister Teknik Elektro, Minat Sistem Komunikasi dan Informatika, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya, Malang, Indonesia (e-mail: alan@ub.ac.id).

Sholeh Hadi Pramono, adalah dosen Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya, Malang, Indonesia (e-mail: sholehpramono@ub.ac.id).

Hadi Suyono, adalah dosen Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya, Malang, Indonesia (email: hadis@ub.ac.id).

melakukan pengecekan terhadap pemanfaatan sumber daya server, apabila ada server yang tidak digunakan maka segera dinon-aktifkan dan begitu juga sebaliknya [4]. Namun demikian pendekatan ini mengalami keterlambatan respon perubahan beban kerja sehingga layanan yang diberikan kepada pengguna tidak maksimal.

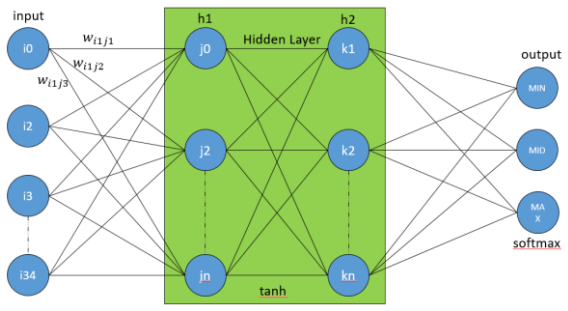
Untuk meresponse perubahan beban kerja yang ada selanjutnya diperlukan migrasi *virtual machine* (VM) ke *physical machine* (PM) lainnya. Proses migrasi VM ke PM dibutuhkan lebih banyak *bandwidth* sehingga *bandwidth* pengguna pada aplikasi yang tersedia akan lebih rendah [5]. Fakta-fakta ini meningkatkan kebutuhan akan teknik prediksi yang dapat digunakan untuk memperkirakan beban kerja yang akan datang, sehingga penyediaan sumber daya server lebih optimal dan efisien. Namun demikian, prediksi beban kerja akan dapat sangat rumit karena keragaman dan permintaan pengguna yang datang bersifat stokastik. Permintaan yang datang dan penggunaan sumber daya server (*CPU*, *memory*, *bandwidth*, dll) menghasilkan pola-pola yang dinamis dan bervariasi (stokastik) [6].

Penelitian ini mencoba untuk melakukan kajian terhadap prediksi klasifikasi beban kerja *web server* dengan memperhatikan parameter sumber daya server (*CPU*, *memory*, *bandwidth*, *load average*) dan jumlah permintaan. Metode untuk melakukan prediksi menggunakan klasifikasi, dengan pendekatan *Artificial Neural Network* (ANN) *Backpropagation*.

Klasifikasi menggunakan ANN *Backpropagation* dengan parameter output adalah *load average CPU* dan parameter input adalah jumlah permintaan, penggunaan *CPU*, *memory*, dan *bandwidth*. Nilai total *load average CPU* dibuat menjadi 3 kelas, yaitu: minimum, medium, dan maximum. Kajian dilakukan pada *web server* yang melayani applikasi SIAM UB, pada periode registrasi mahasiswa (KRS), lebih dari 65000 mahasiswa akan mengakses *web server* SIAM secara bersamaan. Jumlah permintaan yang dilayani server dalam 1 hari dapat mencapai 1.7 juta *request* [8].

II. *Artificial Neural Network* (*ANN*)

ANN adalah elemen pemrosesan yang saling berhubungan dan berfungsi untuk memecahkan masalah-masalah tertentu. ANN terdiri dari 3 lapisan (*layer)*, yaitu: *input layer*, *hidden layer*, dan *output layer* (Gambar 1). Lapisan-lapisan ini saling terhubung dengan sejumlah simpul dan pada *hidden layer* terdapat fungsi aktivasi [7].



Gambar 1. Arsitektur ANN

A. *Fungsi Aktivasi*

Fungsi aktivasi yang sering digunakan adalah fungsi linear, fungsi *threshold*, fungsi *Log-sigmoid* dan fungsi *tanh-sigmoid*.

1) Fungsi threshold

Fungsi *threshold* (persamaan 2.1): Ini hanya menghitung 1 dan 0. Jenis fungsi aktivasi ini disebut juga model McCulloch-Pitts. Model McCulloch-Pitts adalah neuron buatan yang sangat sederhana.

$$y = \begin{cases} +1, jika\ s \geq 0 \\ -1, jika\ s \leq 0 \end{cases} \quad (2.1)$$

2) Fungsi log-sigmoid

Fungsi aktivasi *sigmoid* logistik (persamaan 2.2): Ketika menggunakan fungsi aktivasi *sigmoid*, neuron buatan akan terlihat lebih seperti alami. Nilai output yang diberikan dalam rentang [0,1]

$$y = \frac{1}{1 + e^{-s}} \quad (2.2)$$

3) Fungsi tanh-sigmoid

Fungsi *tanh* (persamaan 2.3) dapat memberikan output bernilai negatif. Nilai output yang diberikan pada pada rentang [−1, 1]

$$y = \frac{e^{s} - s^{-s}}{e^{s} + e^{-s}} \quad (2.3)$$

4) Fungsi softmax

Fungsi aktivasi *softmax* (persamaan 2.4) biasanya digunakan pada lapisan jaringan terakhir, mengubah nilai *arbitrary real* menjadi *posterior probability* kelas c_k dalam rentang (0; 1)

$$p(c_k|x) = \frac{e^{a_k}}{\sum_{i=1}^{m} e^{a_i}} \quad (2.4)$$

III. Confusion Matrix

Confusion matrix adalah sebuah tabel yang menyatakan jumlah data uji yang benar diklasifikasikan dan jumlah data uji yang salah diklasifikasikan. Confusion matrix akan digunakan sebagai alat untuk evaluasi atau mengukur kinerja dari sistem atau metode yang digunakan dalam penelitian ini adalah ANN Backpropagation. Contoh: confusion matrix untuk kelas biner (*minimum load avg.*) ditunjukkan pada Tabel I, dengan keterangan sebagai berikut:

- Kondisi *Positive* (P), yaitu jumlah kondisi positive dalam data.
- Kondisi *Negative* (N), yaitu jumlah kondisi negative dalam data.
- *True Positive* (TP), yaitu jumlah *load average CPU* dari kelas *Min. Load Avg* yang benar dan diklasifikasikan sebagai kelas *Min. Load Avg*.
- *True Negative* (TN), yaitu jumlah *load average CPU* dari kelas *Non Min. Load Avg* yang benar diklasifikasikan sebagai kelas *Non Min. Load Avg*.
- *False Positive* (FP), yaitu jumlah *load average CPU* dari kelas *Non Min. Load Avg* yang salah diklasifikasikan sebagai kelas *Min. Load Avg*.
- *False Negative* (FN), yaitu jumlah *load average CPU server* dari kelas *Min. Load Avg* yang salah diklasifikasikan sebagai kelas *Non. Min. Load Avg*.

TABEL I
CONFUSION MATRIX KELAS BINER

n=180		Kelas Prediksi (Predicted) Min. *Load Avg.* (P)	non-Min. *Load Avg.* (N)
Kelas Sebenarnya (Real)	Min. *Load Avg.* (P)	TP	FN
	non-Min *Load Avg.* (N)	FP	TN

Parameter-parameter yang akan dievaluasi atau diukur adalah sebagai berikut:

1. *Positive Predictive Value* (*PPV*) atau *precision* adalah tingkat ketepatan sistem atau metode melakukan klasifikasi sesuai dengan nilai klasifikasi sebenarnya. Untuk menghitung *precision* menggunakan persamaan 3.1.

$$PPV = \frac{\sum TP}{\sum Predicted\ Kondisi\ Positive} \times 100\% \quad (3.1)$$

2. *True Positive Rate (TPR)* atau *recall* adalah tingkat keberhasilan sistem untuk selalu melakukan klasifikasi dengan benar. Untuk menghitung *sensitivity* menggunakan persamaan 3.2

$$TPR = \frac{\sum TP}{\sum Real\ Kondisi\ Positive} \quad (3.2)$$

3. *Accuracy* adalah tingkat kedekatan antara nilai prediksi dengan nilai sebenarnya. Untuk menghitung *Accuracy* (ACC) dengan menggunakan persamaan 3.3.

$$ACC = \frac{\sum TP + \sum TN}{\sum Total\ Data} \times 100\% \quad (3.3)$$

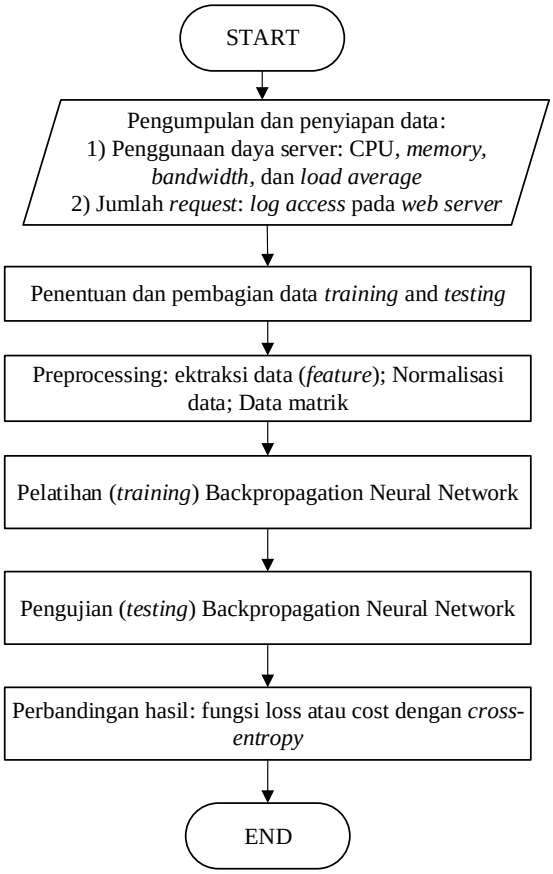
IV. METODOLOGI PENELITIAN

Metodologi penelitian yang dilakukan pada penelitian ini ditunjukkan pada Gambar 2, dengan proses diberikan sebagai berikut:

A. Pengambilan Data

Data-data yang digunakan dalam penyusunan penelitian ini adalah sebagai berikut:

1. Data primer adalah data yang didapatkan dari *web server* SIAM Universitas Brawijaya selama bulan agustus 2017 dan bulan januari 2018 dengan interval 2 jam [8]. Kegunaan dari masing-masing data ini sebagai berikut:
   a. Data log *web server* digunakan untuk mengukur jumlah *request* yang dilayani oleh kluster *web server*.
   b. Data SNMP adalah hasil pengukuran penggunaan sumber daya *CPU*, *Memory* dan trafik jaringan.
2. Data sekunder adalah data yang diperoleh dari hasil studi literature [7], [9]. Data sekunder yang digunakan dalam penelitian ini adalah:
   a. Fungsi aktivasi pada hidden layer adalah tanh.
   b. Fungsi aktivasi pada output layer adalah softmax.
   c. Fungsi Loss atau Cost menggunakan Cross-entropy.
   d. Memperbarui parameter bobot dan bias pada saat proses backpropagation menggunakan mini-batch gradient descent.



Gambar 2. Metodologi Penelitian

B. Preprocessing

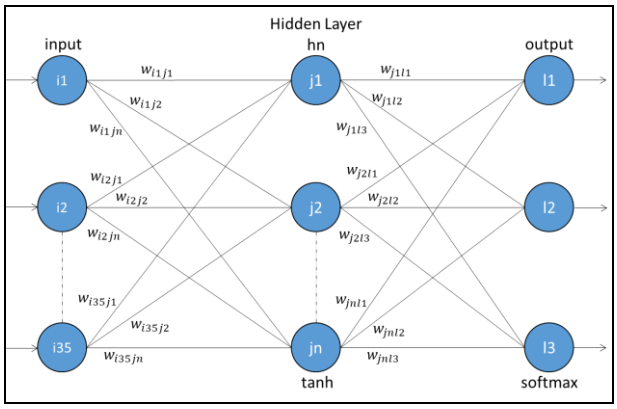
Tahapan pada preprocessing data, yang secara terurut adalah sebagai berikut:

1) Eksktraksi data dengan memecah seluruh bagian data mentah berdasarkan tanda baca (karakter selain alfabet) menjadi vektor string dan dikelompokkan berdasarkan ciri (feature).
2) Melakukan normalisasi data (feature scalling) untuk membuat data numeric pada dataset memiliki rentang nilai (scale) yang sama sehingga tidak ada satu variabel data yang mendominasi variabel data lainnya. Rentang nilai yang dihasilkan dari proses normalisasi ini adalah 0 – 1, kecuali untuk nilai jumlah IP akses tetap menggunakan nilai awal.
3) Data yang dihasilkan dari normalisasi (feature scalling) dikonversi menjadi data matriks yang akan digunakan untuk pelatihan backpropagation neural network. Data yang dihasilkan ini kemudian dibagi menjadi data latih dan data validasi, yaitu 70% data latih dan 30% data validasi.

C. Pelatihan Backpropagation Neural Network

Neural Network (NN) terdiri dari elemen-elemen neuron yang beroperasi secara pararel yang saling

terkoneksi dengan nilai bobot tertentu (gambar 3). Nilai-nilai bobot koneksi antar neuron-neuron ini akan disesuaikan atau dilatih untuk melakukan fungsi tertentu dalam hal ini untuk melakukan klasifikasi beban kerja sebuah kluster server.



Gambar 3. Struktur Neural Network

Algoritma backpropagation yang dilakukan adalah sebagai berikut:

a. Menentukan jumlah neuron dan hidden layer.
b. Menentukan jumlah epoch.
c. Menentukan learning rate.
d. Untuk proses training, dilakukan langkah e – g.
e. Fase propagasi maju (forward):
   i. Jumlahkan semua sinyal input (i_n) dan bobot (w_{ij}) yang masuk ke setiap *hidden unit* (J_n) pada *hidden layer* (h_n), menggunakan persamaan 4.1

$$h\_in_n = b_{j0} + \sum_{i=1}^{n} i_i w_{ij} \tag{4.1}$$

   ii. Hitung keluaran setiap *hidden unit* (J_n) pada *hidden layer* (h_n) menggunakan fungsi aktivasi *tanh*. menggunakan persamaan 4.2

$$h\_out_n = f(h_{in_n}) = \frac{e^s - e^{-s}}{e^s + e^{-s}} \tag{4.2}$$

   iii. Jumlahkan semua sinyal yang masuk ke *output layer* pada setiap unit (l_n), dengan menggunakan persamaan 4.3

$$o\_in_n = b_{l0} + \sum_{i=1}^{n} h\_out_n w_{jl} \tag{4.3}$$

   iv. Hitung keluaran dari setiap *output* unit (l_n) di *output layer*, menggunakan fungsi aktivasi *softmax*, menggunakan persamaan 4.4

$$o\_out_n = \frac{e^{o\_in_n}}{\sum_{a=1}^{n} e^{o\_in_a}} \tag{4.4}$$

   v. Hitung *error* menggunakan fungsi *loss cross-entropy* di setiap *epoch*, menggunakan persamaan 4.5.

$$error = -\left(\frac{1}{n}\right)\left(\sum_{i=1}^{3} y_i \times \log(o\_out_i)\right) + \left((1-y_i)\right) \times \log\left((1-o\_out_i)\right) \tag{4.5}$$

f. Fase propagasi mundur (*backward*):
   i. Hitung faktor *error* pada *output layer* dengan melakukan penurunan partial persamaan 4.5 terhadap o_out_n, menggunakan persamaan 4.6

$$\frac{\partial E_n}{\partial o\_out_n} = \frac{\partial\left(-1 \times \left((y_n \times \log(o\_out_n) + (1-y_n) \times \log(1-o\_out_n)\right)\right)}{\partial o\_out_n} \tag{4.6}$$

   ii. Hitung perubahan bobotnya, menggunakan persamaan 4.7

$$\Delta w_{jl} = \alpha \partial_k z_j \tag{4.7}$$

   iii. Hitung penjumlahan *error*-nya, menggunakan persamaan 4.8

$$\partial\_net_j = \sum_{k=1}^{m} \partial_k w_{kj} \tag{4.8}$$

   iv. Hitung faktor *error* pada *hidden layer*, menggunakan persamaan 4.9

$$\partial_j = \partial\_net_j z_j (1 - z_j) \tag{4.9}$$

   v. Hitung perubahan bobot, menggunakan persamaan 4.10

$$\Delta v_{ji} = \alpha \partial_j i_i \tag{4.10}$$

g. Perubahan bobot
   i. Ubah bobot yang menuju *output layer*, menggunakan persamaan 4.11

$$w_{kj}(t+1) = w_{kj} + \Delta w_{kj} \tag{4.11}$$

   ii. Ubah bobot yang menuju *hidden layer*, menggunakan persamaan 4.12

$$v_{ji}(t+1) = v_{ji}(t) + \Delta v_{ji} \tag{4.12}$$

D. Pengujian

Pada tahap ini dilakukan uji coba untuk mendapatkan struktur NN yang optimal dengan mengkur nilai error (loss) terkecil, tingkat accuracy (%) tertinggi dan waktu training (s). Tahapan dan parameter-parameter yang dilakukan pada proses pengujian ini adalah sebagai berikut:

1) Menentukan jumlah hidden layer, ujicoba pertama menggunakan 1 layer dan ujicoba ke-dua menggunakan 2 layer. Parameter jumlah unit neuron yang diuji adalah 8, 16, 32, 64, 128, 256, dan 512. Hasil (jumlah neuron) dari ujicoba pertama akan digunakan sebagai referensi untuk ujicoba ke-dua.
2) Menentukan nilai learning rate, sebagai berikut 0.00001, 0.0001, 0.001, dan 0.01.
3) Melakukan ujicoba akurasi klasifikasi menggunakan data latih, data uji secara bersamaan dan parameter-parameter yang sudah ditentukan.
4) Setiap proses latih (epoch) langsung dilakukan uji menggunakan data uji, proses ini tidak mempengaruhi nilai bobot.

E. Evaluasi Hasil

Confusion matrix akan digunakan sebagai alat untuk evaluasi atau mengukur kinerja dari sistem atau metode yang digunakan dalam penelitian ini adalah *ANN Backpropagation.* Parameter yang akan diukur adalah tingkat *precision*, tingkat *sensitivity*, dan tingkat *accuracy* dengan menggunakan persamaan 3.1, 3.2 dan 3.3.

V. HASIL DAN PEMBAHASAN

A. Proses Pembelajaran

Proses pembelajaran dilakukan dengan menggunakan script learn.py, dibuat dengan menggunakan bahasa pemrograman python yang telah menyediakan fungsi-fungsi pembelajaran dan pengujian pada ANN dengan algoritma backpropagation. Proses pembelajaran dilakukan untuk mencari konfigurasi terbaik dengan cara sebagai berikut:

1) Menentukan jumlah hidden layer, ujicoba pertama menggunakan 1 layer dan ujicoba ke-dua menggunakan 2 layer.
2) Menentukan jumlah unit neuron pada hidden layer. Parameter jumlah unit yang diuji adalah 8, 16, 32, 64, 128, 256, 512.
3) Menentukan nilai learning rate, parameter learning rate sebagai berikut 0.00001, 0.0001, 0.001, 0.01, 0.1, dan 1
4) Melakukan ujicoba akurasi klasifikasi menggunakan data latih, data uji secara bersamaan dan parameter-parameter yang sudah ditentukan.

Data sumber daya server server dan jumlah IP akses (*request*) yang digunakan pada proses pembelajaran diambil pada bulan agustus 2017 dan Januari 2018.

B. Pengujian

Pengujian dilakukan melalui dua tahap yaitu, pengujian terhadap data yang dilatih/training dan pengujian data baru yang belum pernah dilatih/training. Selama proses pembelajaran berlangsung, error dari tiap pola dapat ditampilkan, error dari tiap pola yang diajarkan pada jaringan yang disimpan dari iterasi 1 sampai dengan 500. Penyebab lamanya proses pembelajaran pada struktur jaringan yang digunakan adalah pola yang diajarkan sangat banyak, sehingga pengulangan dari satu iterasi ke iterasi dipengaruhi oleh banyaknya pola dan struktur data pada pola tersebut.

Pada proses pembelajaran, jaringan melakukan proses backward atau arus balik dengan cara mengubah nilai bobot jaringan, sebelum target error terpenuhi, pembelajaran akan terus dilanjutkan sampai tercapai target error. Semakin kecil target error yang ditetapkan semakin akurat jaringan mengenali dan menghitung bentuk pola baru yang diujikan kepadanya.

TABEL II
OUTPUT ANN DENGAN 1 HIDDEN LAYER

No	Jumlah neuron H1	Jumlah Iterasi	Loss (*error*)	*Accuracy* (%)	Waktu *Training*(s)
1	8	500	1.002	34.64	45
2	16	500	0.9968	33.52	38
3	32	500	0.8770	48.04	37
4	64	500	0.8210	76.54	36
5	128	500	0.7970	83.24	38
6	256	500	0.7528	83.80	42
7	512	500	0.7485	88.83	42

1) Analisis Hasil Pelatihan dengan 1 Hidden layer

Hasil pelatihan data sumber daya server dan jumlah IP akses (*request*), dengan menggunakan struktur jaringan 3 layer terdiri dari 32 unit input, 1 *hidden layer* dan 3 unit output dengan 500 *epoch*, seperti tabel 2. Dengan menggunakan struktur jaringan *1 hidden layer* didapat nilai *loss (error)*, yaitu: avg. loss=0.8564, min. loss=0.7485, max. loss=1.002.

2) Analisis Hasil Pelatihan dengan 2 Hidden Layer

Hasil pelatihan data sumber daya server dan jumlah IP akses (*request*), dengan menggunakan struktur jaringan 5 layer terdiri dari 32 unit input, 2 hidden layer dan 3 unit output dengan 500 iterasi (epoch), seperti tabel 3. Dengan menggunakan struktur jaringan 2 hidden layer didapat nilai loss semakin kecil, yaitu: avg. loss=0.6655, min. loss=0.6537, dan max. loss=0.6733.

TABEL III
OUTPUT ANN DENGAN 2 HIDDEN LAYER

No.	H1	H2	Iterasi	Loss	*Acc.* (%)	Waktu *training* (s)
1	512	16	500	0.6724	90.50	62
2	512	32	500	0.6537	90.50	59
3	512	64	500	0.6733	90.50	49
4	512	128	500	0.6638	90.50	66
5	512	256	500	0.6667	90.50	71
6	512	512	500	0.6631	90.50	143

3) Analisis Hasil Pelatihan Berdasarkan Learning Rate

Hasil pengujian menunjukkan bahwa nilai *learning rate* awal mencapai nilai loss paling optimal. Dalam penelitian ini semakin besar nilai *learning rate*, nilai loss yang dicapai semakin besar. Jaringan optimal saat nilai *learning rate* awal 0.00001 yaitu kemampuan jaringan dalam mengenali data baru adalah yang paling tinggi (90.50%) dengan nilai loss paling kecil (0.6537).

a) Learning rate 0.0001

Hasil pelatihan data sumber daya server dan jumlah IP akses (*request*), dengan menggunakan struktur jaringan 5 layer terdiri dari 32 unit input, 2 *hidden layer* dan 3 unit output dengan 500 iterasi (*epoch*) dan *learning rate* 0.0001, seperti tabel 4.

TABEL IV
OUTPUT ANN DENGAN *LEARNING RATE* 0.0001

No.	H1	H2	Iterasi	Loss	*Accuracy* (%)	Waktu *training* (s)
1	512	32	500	0.7509	90.50	42
2	512	64	500	0.7726	90.50	43
3	512	128	500	0.8040	90.50	52
4	512	256	500	0.8336	90.50	65
5	512	512	500	0.8518	90.50	123

b) Learning rate 0.000001

Hasil pelatihan data sumber daya server dan jumlah IP akses (*request*), dengan menggunakan struktur jaringan 5 layer terdiri dari 32 unit input, 2 *hidden layer* dan 3

unit output dengan 500 iterasi (*epoch*) dan *learning rate* 0.000001, seperti pada tabel 5.

TABEL V
OUTPUT ANN DENGAN *LEARNING RATE* 0.000001

No.	H1	H2	Jumlah Iterasi	Loss	*Accuracy* (%)	Waktu *training* (s)
1	512	32	500	0.9919	59.78	43
2	512	64	500	0.9393	61.45	47
3	512	128	500	0.9773	62.01	54
4	512	256	500	0.9731	60.34	72
5	512	512	500	0.9388	58.10	129

c) Learning rate 0.001

Hasil pelatihan data sumber daya server dan jumlah IP akses (*request*), dengan menggunakan struktur jaringan 5 layer terdiri dari 32 unit input, 2 *hidden layer* dan 3 unit output dengan 500 iterasi (*epoch*) dan *learning rate* 0.001, seperti pada tabel 6.

TABEL VI
OUTPUT ANN DENGAN LEARNING RATE 0.001

No.	H1	H2	Iterasi	Loss	*Accuracy* (%)	Waktu *training* (s)
1	512	32	500	1.137	90.50	43
2	512	64	500	1.173	90.50	47
3	512	128	500	1.245	90.50	54
4	512	256	500	1.276	90.50	72
5	512	512	500	1.310	90.50	129

C. *Evaluasi Kinerja System Pengelompokkan*

Confusion matrix digunakan sebagai alat untuk melakukan evaluasi atau mengukur kinerja dari sistem atau metode klasifikasi yang digunakan dalam penelitian ini (ANN Backpropagation). Parameter-parameter yang digunakan untuk mengukur kinerja dari sistem adalah PPV atau precision, TPR atau sensitivity, Accuracy (ACC). Bobot yang dihasilkan dari arsitektur ANN yang optimal akan digunakan untuk membentuk tabel confusion matrix 3-kelas (tabel 7), dari tabel tersebut akan dibentuk menjadi tabel confusion matrix biner (tabel 7). Kemudian dilakukan proses perhitungan PPV atau precision, TPR atau sensitivity dan Accuracy (ACC) untuk 180 data uji menggunakan persamaan 3.1, 3.2, dan 3.3.

TABEL VII
CONFUSION MATRIX 3 KELAS *LOAD AVG CPU*

n=180		Predicted Class		
		Min.	Med.	Max
Real Class	Min.	54	3	3
	Med.	3	54	3
	Max.	3	3	54

TABEL VIII
CONFUSION MATRIX BINER *LOAD AVG CPU*

n=180		Predicted Class		
		Min. (P)	non-Min. (N)	
Real Class	Min. (P)	54 (TP)	6 (FN)	60
	non-Min. (N)	6 (FP)	114 (TN)	120
		60	120	

Nilai *PPV* atau *precision* ditentukan menggunakan persamaan 3.1, adalah sebagai berikut:

$$PPV = \frac{54}{60} \times 100\% \qquad PPV = 90\%$$

Nilai *TPR* atau *sensitivity* ditentukan menggunakan persamaan 3.2, adalah sebagai berikut:

$$TPR = \frac{54}{60} \qquad TPR = 0.9$$

Nilai *Accuracy* (ACC) ditentukan menggunakan persamaan 3.3, adalah sebagai berikut:

$$ACC = \frac{54 + 114}{180} \times 100\% \qquad ACC = 93\%$$

VI. KESIMPULAN

Prediksi kelas konsolidasi beban kerja *CPU* dalam kluster *web server* untuk penyediaan sumber daya server yang optimal menjadi fokus diskusi pada makalah ini. Prediksi konsolidasi beban kerja kluster server diklasifikasikan menjadi 3 kelas, yaitu: Min (0-2), Medium (3-6), Maximum (>7). Metode *artificial neural network backpropagation* digunakan untuk memprediksi kelas konsolidasi beban kerja server berdasarkan parameter input penggunaan *CPU*, *memory*, jaringan (*throughput*) dan jumlah IP akses. Disamping itu pengujian juga dilakukan dengan input 1 dan 2 hidden layer, serta pengujian dengan *learning rate* yang berbeda yaitu 0.0001, 0.000001, dan 0.001.

Berdasarkan hasil pengujian yang telah dilakukan, arsitektur ANN *backpropagation* dengan 32 input, 2 *hidden layer* dengan jumlah neuron h1 512; h2 32, 3 output, dan *learning rate* 0.00001 menghasilkan prediksi (klasifikasi) konsolidasi beban kerja kluster server terbaik dengan tingkat precision 90%, tingkat sensitivitas 0.9, dan tingkat akurasi 93%.

VII. DAFTAR PUSTAKA

[1] X. Wang, A. Abraham, K. A. Smith, Intelligent web traffic mining and analysisl, Journal of Network and Computer Applications.

[2] Setyawan, Raden Arief. "Analisis Implementasi Load Balancing dengan Metode Source Hash Scheduling pada Protocol SSL." Jurnal *EECCIS*, 2014: Vol. 8, No. 2.

[3] T. C. Ferreto, M. A. Netto, R. N. Calheiros, and C. A. De Rose. "Server consolidation with migration control for virtualized *data centers*." *Future Gener. Comput. Syst.*, 2011: vol. 27, no. 8, pp. 1027–1034.

[4] G. Lovasz, F. Niedermeier, and H. de Meer. "Performance tradeoffs of energy-aware *virtual machine* consolidation." Kluster *Comput.*, 2013: vol. 16, no. 3, pp. 481–496.

[5] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari. "Server workload analysis for power minimization using consolidation." *Proc. USENIX Annu. Tech. Conf.*, 2009: pp. 28–28.

[6] Z. Gong, X. Gu, and J. Wilkes. "PRESS: Predictive elastic resource scaling for cloud systems,." *Proc. IEEE Int. CNSM*, 2010: pp. 9–16.

[7] Ida Wahyuni, Nakhel Rifqi Adam, Wayan Firdaus Mahmudy, Atiek Iriany. Modeling Backpropagation Neural Network for Rainfall Prediction in Tengger East Java. SIET, 2017.

[8] UPT TIK Universitas Brawijaya. Tersedia di: < http://175.45.184.129/awstats/awstats.pl?config=siam-64bit> [Diakses 24 Maret 2018]

[9] S. Kosasi, K. Barat. Penerapan Metode Jaringan Saraf Tiruan Backpropagation untuk Memprediksi Nilai Ujian Sekolah. J. Teknol., vol 7, no. 1, pp. 20-28, 2014.