ISSN: 0128-7680
Pertanika J. Sci. & Technol. 8(1): 41 - 54 (2000)

Analisis Pengawalan Pemberat Rangkaian Neural Perambatan-Balik untuk Pengecaman Aksara Jawi

Ramlan Mahmod dan Khairuddin Omar
Jabatan Sains Komputer
Fakulti Sains Komputer dan Teknologi Maklumat
Universiti Putra Malaysia
43400 Serdang Selangor

Jabatan Sains dan Pengurusan Sistem
Fakulti Teknologi dan Sains Maklumat
Universiti Kebangsaan Malaysia
43600 Bangi Selangor

Received 27 August 1996

ABSTRAK

Satu daripada beberapa faktor yang mempengaruhi keupayaan ***Rangkaian Neural*** (atau ringkasnya **RN**) dalam fasa pengecaman adalah nilai mula yang diberi kepada pemberat semasa fasa latihan. Rangkaian yang telah dilatih boleh terperangkap ke dalam *minimum tempatan* apabila nilai mula pemberat tidak ditentukan dengan betul. Kertas ini membentangkan analisis terhadap keupayaan rangkaian untuk mengecam aksara Jawi setelah ia dilatih menggunakan kaedah pengawalan pemberat yang berlainan. Tiga kaedah yang biasa digunakan ialah *sifar, rawak* dan *rawak Nguyen-Widrow.* Kertas ini membincangkan kesan ketiga-tiga kaedah ini terhadap keupayaan pengecaman rangkaian.

ABSTRACT

One of the factors that influences the recognition ability of a neural network is the initial values given to the weight vector during the training phase. The network may be trapped into a local minima if the initial weights are not chosen carefully. This paper presents an analysis of the ability of the network to recognise Jawi characters after it was trained using different methods of weight initialization. Three most common methods are zero, random and Nguyen-Widrow random. This paper presents the effect of these three methods on the ability of the network's recognition.

Keywords: Rangkaian neural, perambatan-balik, Rawak Nguyen-Widrow, pengecaman corak

PENGENALAN

Perambatan-balik adalah tatacara *kecerunan berasas-turunan* untuk mensetkan pemberat bagi satu **RN** multi-lapis rambatan ke hadapan yang juga dikenali sebagai *perseptron multi-lapis.* Bagaimanapun *perambatan-balik* hanyalah satu cara menganggarkan tatacara kecerunan turunan sebagaimana biasa ia dilaksanakan, Ruck *et al* (1992), iaitu, ralat kuasadua yang dikira itu diperuntukkan kepada permukaan ralat yang hanya didefinasikan oleh vektor latihan semasa dan bukan vektor latihan secara keseluruhan. Begitu juga tatacara ini sangat

bergantung kepada nilai awal pemberat yang diberikan, iaitu sama ada dengan memberikan nilai sifar kepada seluruh pemberat atau secara rawak dalam julat nilai tertentu. Kertas ini memaparkan satu analisis perbandingan pelbagai nilai pemberat awal seperti yang dinyatakan tadi termasuk menggunakan kaedah pemberat awal Nguyen-Widrow. Satu perbandingan kaedah nilai awal yang manakah yang dikatakan paling baik akan dipaparkan dengan menggunakan data sintasis. Di dalam kertas ini kami mengutarakan pencaman corak huruf-huruf tunggal Jawi sebagai aplikasi utama.

Bahagian berikut menjelaskan secara ringkas latarbelakang RN. Bahagian berikutnya menjelaskan implementasi bagi rangkaian ini dengan aplikasinya kepada aksara Jawi. Hasil simulasi dipaparkan di dalam bahagian selanjutnya. Akhir sekali hasil ujikaji ini disimpulkan di bahagian kesimpulan.

LATAR BELAKANG

Bahagian ini menjelaskan paradigma RN seperti senibina *perseptron multi-lapis*, *fungsi pengaktifan*, model *latihan perambatan-balik*, dan *algoritma*nya.

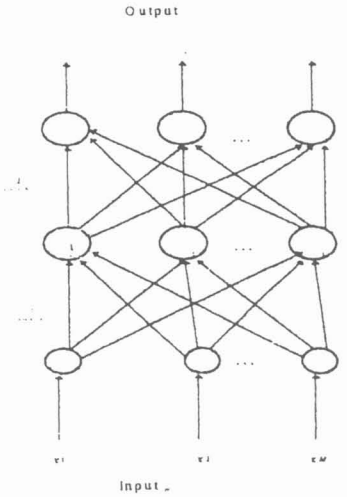
Senibina Perseptron Multi-Lapis

Senibina *perseptron multi-lapis* adalah lanjutan daripada *perseptron satu-lapis*, yang dikaji oleh Rosenblatt pada pertengahan 1950'an. *Perseptron multi-lapis* bermakna terdapat lebih daripada satu lapisan nod-nod yang terhubung sepenuhnya di antara lapisan-lapisan itu. Rajah 1 menunjukkan struktur *perseptron multi-lapis*.

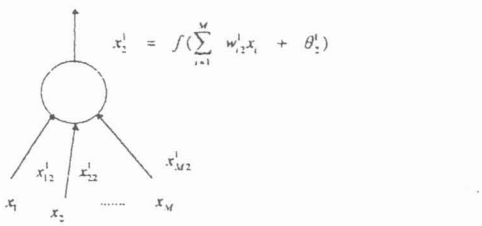
Setiap nod dalam Rajah 1a digambarkan oleh bulatan berlorek di dalam Rajah 1b, yang melaksanakan perjumlahan input berpemberat dan seterusnya digunakan oleh *fungsi pengaktifan tak-linear* dalam Rajah 1c untuk mendapatkan nilai output untuk setiap nod pada lapisan yang seterusnya. Rangkaian tadi menunjukkan terdapat satu lapisan tersembunyi, iaitu satu lapisan yang bukan lapisan input atau lapisan output. Rangkaian itu mempunyai M unit nod input, H nod pada lapisan tersembunyi, dan N nod output. Lapisan input tidak dikira sebagai satu lapisan, iaitu hanya lapisan yang menggunakan *fungsi pengaktifan tak-linear* sahaja dikira sebagai satu lapisan. Ini bermakna lapisan pertama bermula di lapisan tersembunyi. Senibina RN jenis *perseptron multi-lapis* akan digunakan dalam kertas ini.

Fungsi Pengaktifan

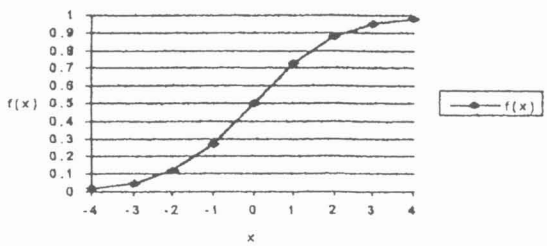
Pelbagai *fungsi pengaktifan* boleh digunakan. Fungsi yang sering digunakan ialah *fungsi Sigmoid* (lengkok bentuk-S), (Fauset 1994); dan (Ramlan dan Khairuddin 1996). Fungsi ini tak-linear dan boleh dicari terbitannya. Dua bentuk fungsi tersebut ialah *fungsi logistik* dan *fungsi tangen hiperbola* yang mempunyai sifat-sifat yang sama di antara keduanya. Kedua-duanya mempunyai kebaikan terutama didalam kegunaan RN yang dilatih oleh *perambatan-balik*, kerana terdapat hubungan yang mudah di antara nilai fungsi pada satu titik dan nilai terbitannya (boleh mendapatkan terbitannya) pada titik tadi boleh mengurangkan beban pengiraan ketika latihan. Fungsi ini juga bersifat global,



Rajah 1a. Senibina rangkaian multi-lapis



Rajah 1b.Butiran terperinci tentang nod



Rajah 1c. Fungsi pengaktifan sigmoid perpaduan

iaitu kesan nilai fungsi pada satu titik dan nilai terbitannya adalah terhadap julat -∞ hingga +∞ dan sangat berguna untuk tujuan pengkelasan yang fungsi outputnya bernilai 0 atau 1 atau -1. Berikut diberikan fungsi Sigmoid Binari yang sering digunakan:

$$f(x) = \frac{1}{1+\exp(-\sigma x)}$$

dan terbitannya diberikan oleh

$$f(x) = \sigma(x)[1- f(x)]$$

Fungsi sigmoid yang julatnya adalah dari 0 hingga 1 sering digunakan untuk RN jika output dikehendaki adalah berbentuk binari atau berada dalam julat di antara 0 dan 1. Sebaliknya jika julat berada di antara –1 dengan 1 adalah lebih baik ditukar kepada bentuk bipolar dan kemudian gunakan fungsi sigmoid bipolar, Fausett(1994) seperti berikut:

$$g(x) = 2f(x)-1 = \frac{2}{2+\exp(-\sigma x)} - 1$$

$$= \frac{1-\exp(-\sigma x)}{1+\exp(-\sigma x)}$$

dan terbitannya diberikan oleh:

$$g'(x) = \frac{\sigma}{2}[1+ g(x)][1 - g(x)]$$

Fungsi pengaktifan yang digunakan didalam kertas ini adalah dari jenis bipolar.

Model Latihan Perambatan-balik

Perambatan-balik adalah kaedah kecerunan menurun untuk melatih pemberat dalam *Perseptron multi-lapis* dan juga dikenali sebagai petua *delta teritlakkan,* Ruck et al(1992). Latihan bagi rangkaian ini melibatkan tiga peringkat*: menghantar corak isyarat input latihan kehadapan, pengiraan dan merambat balik ralat yang dikira,* dan *kemaskini semua pemberat,* Fausett(1994).

Untuk sebarang masalah yang diberi, terdapat satu set vektor latihan katalah X, dalam keadaan untuk setiap $x \in X$, maka terdapat vektor yang dikehendaki d D, dengan D adalah set output yang bersekutu dengan vektor latihan dalam X.

Katalah ralat semasa E_p didefinasikan oleh:

$$E_p = \frac{1}{2}\left(d_p - z_p\right)^T \left(d_p - z_p\right)$$

$$= \frac{1}{2}\sum_{k=1}^{N}\left(d_{k,p} - z_{k,p}\right)^2 \qquad (1)$$

dengan $d_{k,p}$ adalah komponen ke k daripada p vektor output yang dikehendaki d_p, dan $z_{k,p}$ adalah komponen ke k daripada vektor output z_p sebenar apabila p latihan x_p menjadi input kepada p*erseptron multi-lapis.*

Katakan jumlah ralat E_T didefinasikan seperti berikut:

$$E_T = \sum_{p=1}^{P} E_p$$

dengan ***P*** adalah *kekardinalan* bagi ***X***. Ambil ingatan bahawa E_T adalah fungsi bagi kesemua set latihan dan juga pemberat dalam rangkaian. *Petua perambatan-balik* didefinasikan sebagai berikut:

$$\Delta w(t) = -\eta \frac{\partial E_p}{\partial w} + \alpha \Delta w(w-1) \qquad (2)$$

dengan η, adalah *kadar pembelajaran*, iaitu suatu nilai positif yang kecil, α adalah faktor *momentum* yang juga mempunyai nilai positif yang kecil, dan w mewakili pemberat. Di dalam persamaan di atas, $\Delta w(t)$ adalah perubahan pemberat yang dikira pada masa *t*. Apabila sebutan *momentum* digunakan ($\alpha \neq 0$), petua latihan ini disebut *kaedah momentum*. Apabila nilai α dalam (2) sama dengan sifar, ia disebut *perambatan-balik sertamerta* kerana ia mengira kecerunan berdasarkan vektor latihan tunggal. Terdapat variasi lain iaitu *perambatan-balik berkelompok*, yang mengira perubahan pemberat menggunakan kecerunan berdasarkan kepada *ralat total* E_T. Untuk masalah yang melibatkan pemaparan keputusan, maka kaedah *perambatan-balik serta-merta* digunakan.

Pilihan Pengawalan Pemberat dan Pincang

Pilihan untuk mengawalkan pemberat adalah sangat penting dan memberi kesan kepada rangkaian sama ada telah mencapai tahap *ralat minimum yang sejagat* dan, jika sekalipun telah mencapai tahap tersebut adakah ia menumpu dengan cepat. Persoalan ini akan cuba digarab dengan memilih beberapa kaedah mengawalkan nilai pemberat. Bahagian berikut akan membicarakan kaedah-kaedah tersebut.

Pengawalan Pemberat Sifar

Dalam kaedah ini semua pemberat diawalkan dengan nilai sifar, ini termasuklah nilai awal pemberat *pincang*.

Pengawalan Pemberat Rawak

Kaedah ini sangat bergantung kepada *fungsi pengaktifan* yang digunakan. *Fungsi sigmoid* yang digunakan didalam rangkaian ini mempunyai batas atas dan batas bawah. Dengan alasan ini, adalah sangat penting supaya tidak memilih suatu nilai yang akan menghasilkan nilai *fungsi pengaktifan*nya bersamaan dengan sifar. Ini termasuklah nilai terbitannya. Begitupun nilai awalnya juga tidak boleh terlalu besar, atau isyarat input awal kepada unit tersembunyi atau unit output hendaklah berada di dalam julat yang membolehkan nilai terbitan *fungsi sigmoid* itu dengan suatu nilai yang kecil. Dengan perkataan lain, jika nilai awal pemberat terlalu kecil, input bersih kepada unit tersembunyi atau

unit output akan menghampiri sifar. Nilai yang dicadangkan ialah suatu nilai rawak yang berada di antara julat -0.5 dan 0.5 ataupun nilai julat yang berpatutan. Pelbagai *fungsi rawak* boleh digunakan untuk menjana nombor-nombor dalam julat -0.5 dan 0.5. Dalam kajian ini, kami tidak membuat analisis terhadap setiap fungsi rawak secara terperinci sebelum membuat pilihan. Kami hanya mengambil satu daripada fungsi rawak yang kerap digunakan dan rumus fungsi tersebut adalah seperti di bawah:

rawak = (float) ((rand()%5000)/10000.00).

yang rand() adalah adalah fungsi pratakrif dalam bahasa pengaturcaraan C. Ia berfungsi menjanakan nombor rawak berada di antara -5000 dan 5000 sahaja.

Pengawalan Pemberat Rawak Nguyen-Windrow

Kaedah ini boleh digunakan untuk mengawalkan nilai pemberat dan *pincang*nya dan dikatakan boleh menumpu cepat berbanding dengan pengawalan rawak biasa, Fausett(1994). Dalam kaedah ini sedikit ubahsuai terhadap perambatan ralat dan ubahsuai ini berasaskan analisa geometri iaitu melihat tindakbalas di antara neuron di lapisan tersembunyi dengan satu unit input; dan kemudian dikembangkan kepada beberapa unit input. Pemberat untuk lapisan di antara unit-unit tersembunyi dengan unit-unit output diawalkan dengan nilai yang berada di antara -0.5 dan 0.5. Manakala pemberat untuk lapisan di antara unit-unit tersembunyi dengan unit-unit input diubahsuaikan supaya dapat memperbaiki kebolehan proses pembelajarannya. Pembolehubah-pembolehubah dan rumus yang digunakan ialah η adalah bilangan unit input, manakala p *adalah* bilangan unit tersembunyi, dan β adalah faktor skala:

$$\beta = 0.7(p)^{1/n}$$

Tatacara untuk kaedah ini adalah seperti berikut:

Langkah 0: Untuk setiap unit tersembunyi ($j = 1, 2, ..., p$):

Langkah 1: Awalkan Vektor Pemberat (VP) nya dengan (dari unit input):

$v_{i,j}$ (*lama*) = nombor rawak di antara -0.5 dan 0.5 (atau dalam julat tertentu selain daripada sifar).

Langkah 2: Kira $\left\| v_j(lama) \right\|$

Langkah 3: Awalkan semula dengan:

$$v_{i,j} = \frac{\beta v_{i,j}(lama)}{\left\| v_j(lama) \right\|}$$

dan setkan pincang v_{0j} = nombor rawak di antara $-\beta$ dan β.

Langkah 4: Uji syarat berhenti.

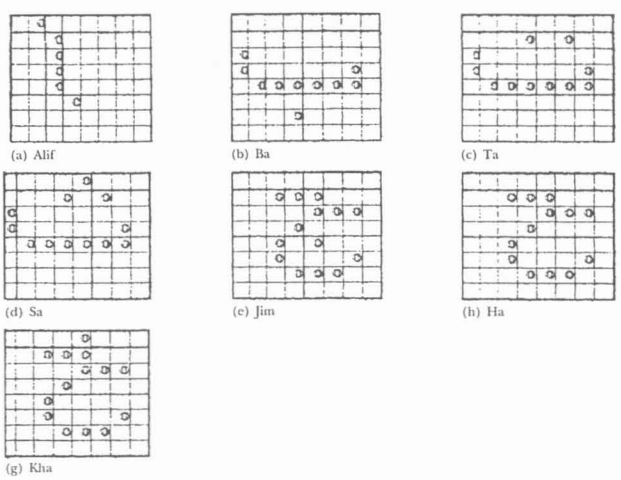
IMPLEMENTASI

Pengkodan Data

Corak input dan tindak balas atau outputnya yang digunakan ketika latihan hendaklah diwakilkan dalam bentuk yang tertentu dan sesuai untuk diproses di dalam RN. *Perwakilan dwikutub* dikatakan mempunyai ciri-ciri yang sangat baik jika dibandingkan dengan *perwakilan perduaan*, (Fausett 1994). Oleh itu, corak input dan corak output yang berbentuk *perwakilan dwikutub* ini telah digunakan di dalam semua latihan yang dilakukan. Pertimbangkan huruf-huruf Jawi tunggal seperti yang diberikan di dalam Rajah 2.

Sebanyak 7 corak input telah digunakan didalam *algoritma latihan*. Kita akan menggunakan rangkaian ini untuk mengkelaskan setiap *vektor input* itu sama ada dipunyai atau tidak dipunyai oleh setiap 7 kelas kategori. Rangkaian akan mempunyai 7 unit output untuk mengkelaskan setiap unit input. Vektor input mempunyai saiz corak pixel 8x8, atau 64-tutupan. Setiap corak '.' diwakilkan dengan nilai 1 manakala corak ' ' pula diwakilkan dengan –1, seperti yang digambarkan di dalam Rajah 2a.

Menukarkan corak dua dimensi kepada satu vektor input dilakukan dengan mudah, iaitu dengan hanya mencantumkan baris-baris, iaitu baris pertama akan diikuti baris kedua dan seterusnya, sebagai contoh corak, huruf 'Alif' akan menjadi



Rajah 2. Contoh huruf Jawi

-1 1 -1 -1 -1 -1 -1 -1,
-1 -1 1 -1 -1 -1 -1 -1,
-1 -1 1 -1 -1 -1 -1 -1,
-1 -1 1 -1 -1 -1 -1 -1,
-1 -1 1 -1 -1 -1 -1 -1,
-1 -1 1 -1 -1 -1 -1 -1,
-1 -1 -1 -1 -1 -1 -1 -1,
-1 -1 -1 -1 -1 -1 -1 -1.

Rajah 2a. Perwakilan huruf 'Alif' dalam bentuk perwakilan dwikutub

-1 1 -1 -1 -1 -1 -1 -1, -1 -1 1 -1 -1 -1 -1 -1, -1 -1 1 -1 -1 -1 -1 -1, -1 -1 1 -1 -1 -1 -1 -1, -1 -1 1 -1 -1 -1 -1 -1, -1 -1 1 -1 -1 -1 -1 -1, -1 -1 -1 -1 -1 -1 -1 -1, -1 -1 -1 -1 -1 -1 -1 -1.

Tindakbalas yang betul bagi corak pertama diwakilkan dengan nilai 1, manakala tindakbalas yang tidak betul bagi corak yang lain dinyatakan dengan -1. Ini bermakna bagi satu corak, misalnya huruf 'Alif', vektor output boleh diwakilkan dengan (1 -1 -1 -1 -1 -1 -1).

Algoritma

Algoritma yang akan dipaparkan di dalam kertas ini dibahagikan kepada dua bahagian iaitu *Algoritma Latihan* dan *Algoritma Aplikasi* seperti yang diutarakan oleh Fausett(1994) di bahagian berikut:

ALGORITMA LATIHAN

Secara amnya algoritma ini dibahagikan kepada 3 bahagian utama iaitu menghantar isyarat input bersih kepada lapisan yang lebih atas dalam langkah kehadapan, mengira ralat dalam perambatan-balik, dan akhir sekali kemas kini pemberat dan pincangkannya. Algoritma latihan diberikan oleh tatacara berikut:

Langkah 0. Awalkan Pemberat (mengikut keterangan 2.5.1 hingga 2.5.4).

Langkah 1. Selagi syarat berhenti masih palsu, lakukan langkah 2-9.

Langkah 2. Untuk setiap pasangan latihan, lakukan langkah 3-8.
Langkah kehadapan:

Langkah 3. Setiap unit input (X_i, i = *1, 2, ..., n*) menerima isyarat input x_i dan menghantarkan isyarat tersebut kepada semua unit di layar yang di atasnya (unit-unit tersembunyi).

Langkah 4. Setiap unit tersembunyi (Z_j, $j = 1, 2, \ldots, p$) menjumlahkan isyarat input berpemberat,

$$z\_in_j = v_{0,j} + \sum_{i=1}^{n} x_i v_{i,j},$$

menggunakan fungsi pengaktifannya untuk mendapatkan nilai isyarat output,

$$z_j = f\left(z\_in_j\right)$$

dan menghantarkan isyarat itu kepada semua unit yang berada di layar di atasnya (unit-unit output).

Langkah 5. Setiap unit output (Y_k, $k = 1, 2, \ldots, m$) menjumlahkan isyarat input berpemberat,

$$y\_in_k = W_{0,k} + \sum_{j=1}^{p} z_i w_{j,k},$$

menggunakan fungsi pengaktifannya untuk mendapatkan nilai isyarat output,

$$y_k = f\left(y\_in_k\right)$$

RALAT PERAMBATAN BALIK

Langkah 6. Setiap unit output (Y_k, $k = 1, 2, \ldots, m$) menerima satu corak sasaran berpasangan dengan corak latihan input, mengira ralat,

$$\delta_k = \left(t_k - y_k\right) f'\left(y\_in_k\right),$$

mengira pembetulan pemberat (digunakan untuk kemaskini pemberat W_j, k kemudian),

$$\Delta w_{j,k} = \alpha \delta_k z_j,$$

mengira pembetulan pincang (digunakan untuk kemaskini nilai pincang W_0, k kemudian),

$$\Delta w_{0,k} = \alpha \delta_k,$$

dan menghantar δ_k ke unit di layar di bawahnya.

Langkah 7. Setiap unit tersembunyi ($Z_j = 1, 2, \ldots, p$) menjumlahkan input delta tadi (dari unit-unit di layar yang di atasnya),

$$\delta\_in_j = \sum_{k=1}^{m} \delta\_w_{j,k},$$

didarabkan dengan terbitan fungsi pengaktifannya untuk mengira ralat,

$$\delta_j = \delta\_in_j f'\left(z\_in_j\right),$$

mengira ralat pemberat (digunakan untuk kemaskini $v_{i,j}$ kemudian),

$$\Delta v_{i,j} = \alpha \delta_j x_i,$$

dan mengira pembetulan pincangnya (digunakan untuk kemaskini $v_{0,j}$ kemudian),

$$\Delta v_{0,j} = \alpha \delta_j.$$

KEMASKINI PEMBERAT DAN PINCANGNYA

Langkah 8. Setiap unit output (Y_k = *1, 2, ..., m*) mengemaskini pincang dan pemberatnya (j = *0, 1, 2, ..., p*):

$$w_{j,k}(baru) = w_{j,k}(lama) + \Delta w_{j,k}.$$

Setiap unit tersembunyi (Z_j = *1, 2, ..., p*) mengemaskini pincang dan pemberatnya (i = *0, 1, 2, ..., n*):

$$v_{i,j}(baru) = v_{i,j}(lama) + \Delta v_{i,j}.$$

Langkah 9. Uji syarat berhenti.

Algoritma Aplikasi

Algoritma aplikasi diberikan oleh tatacara berikut:

Langkah 0. Awalkan Pemberat (diperoleh daripada Algoritma Latihan).

Langkah 1. Untuk setiap vektor input, lakukan langkah 2-4.

Langkah 2. Setiap unit input (X_i, i = *1, 2, ..., n*) menerima isyarat input dan menghantarkan isyarat tersebut kepada semua unit di layar yang di atasnya (unit-unit tersembunyi).

Langkah 3. Setiap unit tersembunyi (Z_j = *1, 2, ..., p*) menjumlahkan isyarat input berpemberat,

$$z\_in_j = v_{0,j} + \sum_{i=1}^{n} x_i v_{i,j},$$

menggunakan fungsi pengaktifannya untuk mendapatkan nilai isyarat output,

$$z_j = f(z\_in_j),$$

dan menghantarkan isyarat itu kepada semua unit yang berada di layar di atasnya (unit-unit output).

Langkah 4. Setiap unit output (Y_k, $k = 1, 2, ..., m$) menjumlahkan isyarat input berpemberat,

$$y\_in_k = w_{0,k} + \sum_{j=1}^{p} z_j w_{j,k},$$

menggunakan fungsi pengaktifannya untuk mendapatkan nilai isyarat output,

$$y_k = f(y\_in_k)$$

Hasil-Perbandingan Prestasi

Senibina rangkaian ini telah diimplemenkan pada mikrokumputer 486 serasi IBM dan dilarikan menggunakan bahasa C++. Tiga ujikaji telah dijalankan berdasarkan kepada bilangan unit tersembunyi (UT). Bilangan UT yang digunakan ialah (i) 4, (ii) 8, dan (iii) 12 untuk diaplikasikan kepada beberapa huruf Jawi. Huruf jawi yang dipilih *ialah Alif, Ba, Ta, Sa, Jim, Ha, dan Kho,* ini menjadikan bilangan pasangan corak input output ialah sebanyak 7. Kadar latihan yang digunakan untuk kesemua latihan ialah bernilai 0.2. Faktor momentum tidak dipertimbangkan dalam kajian ini. Latihan diteruskan sehingga mencapai 50,000 kitar dan *jumlah ralat kuasa-dua* untuk kesemua latihan diuji supaya menghampiri sifar. Berikut adalah keterangan hasil perbandingan di antara tiga kaedah nilai awal pemberat.

PEMBERAT SIFAR

Berpandukan kepada Rajah 3, latihan dengan menggunakan nilai awal pemberat sifar tidak dapat mengecam corak yang dipertimbangkan tadi. Untuk kes ini walaupun latihan telah mencecah 50,000 kitaran, senibina rangkaian masih belum dapat mengenali corak tadi. Untuk kes 4-UT, *jumlah kuasa-dua ralat* bernilai 21.00 dari permulaan latihan sehinggalah ke kitaran 50,000, manakala untuk 8-UT ralat kuasa dua berkurangan (iaitu bernilai 6.0 sehinggalah ke kitaran 26,000 dan bertambah sebanyak 1.0 menjadi 7.0 pada kitaran 27,000) tetapi masih gagal mengecam corak tadi. Bagi 12-UT pula penumpuan bertambah buruk dan nilai kuasa dua ralat menjadi 24.0 dari mula latihan sehinggalah tamat latihan. Fenomena ini menunjukkan ia terperangkap di dalam minimum tempatan.

PEMBERAT RAWAK

Berpandukan kepada Rajah 4, latihan menggunakan nilai awal rawak juga didapati tidak dapat mengecam corak yang dipertimbangkan tadi. Bagi ketiga-tiga jenis UT, secara relatifnya sangat perlahan untuk menumpu. Bagi 4-UT nilai ralat kuasa-duanya bermula dengan 18.0 dan meningkat sedikit pada kitar 6,000 dan beransur-ansur menyusut sehinggalah kepada kitaran ke 37,000 dan kemudian sesudah itu meningkat semula dan kembali menyusut pada

kepada nilai awal pemberat dan telah dibuktikan melalui contoh-contoh yang telah dipaparkan didalam kertas ini.

Hasil eksperimen menunjukkan pengawalan pemberat menggunakan pendekatan Nguyen-Widrow untuk melatih *perseptron multi-lapis* berjaya mempercepatkan penumpuan dan mencapai *ralat minimum* sifar. Ini sekaligus menghasilkan keupayaan pengecaman corak ke takat 100 %. Namun demikian bilangan nod-nod yang perlu didalam senibina RN hendaklah dipilih dengan teliti agar ketepatan pengecaman yang baik ini dapat diperoleh.

RUJUKAN

FAUSETT L. 1994. *Fundamental of Neural Networks: Architectures, Algorithms, & Applications.* New Jersey: Prentice-Hall.

RAMLAN BIN MAHMOD dan KHAIRUDDIN BIN OMAR, 1996. Rangkaian Neural: Satu Tinjauan Ringkas. Laporan Teknik Jabatan Sains Komputer, Universiti Pertanian. ***SAK/TR-010/96.***

RUCK D. W., S. K. ROGERS, M. KABRISKY, P. S. MAYBECK, M. E. OXLEY, 1992. Comparative analysis of backpropagation & the extended kalman filter for training multilayer perceptrons. *IEEE Transactions on Pattern Analysis & Machine Intelligence.* 14(6): 686-691.

TSOI, AH CHUNG. 1994. Constructive Algorithms. A Course on Artificial Neural Networks. Jointly Organised by MIMOS & Computer Centre, University of Malaya, 4-8 July 1994.