Pertanika J. Sci. & Technol. 8(2): 241-252 (2000)
ISSN: 0128-7680

Rangkaian Neural Genetik Aplikasi dalam Pengecaman Aksara Jawi

Ramlan Mahmod, Khairuddin Omar and Md. Nasir Sulaiman
Jabatan Sains Komputer
Fakulti Sains dan Pengajian Alam Sekitar
Universiti Putra Malaysia
43400 Serdang Selangor Darul Ehsan, Malaysia.

Received: 27 August 1996

ABSTRACT

The basic objective of Genetic Algorithm (GA) is looking at the original evolution process in the form of software version. It is commonly used for optimization problem. In this process a population can expand, cloned and die within seconds. These changes occur continuously. Now, the concept of GA has been extended to Neural Networks (NN). This paper discusses the concept or the evolution process which was used in the NN.

ABSTRAK

Objektif asas bagi *Algoritma Genetik* (atau ringkasnya AG) ialah melihat proses evolusi asli dalam bentuk satu versi perisian. Ia sering digunakan untuk masalah pengoptimuman. Dalam proses ini suatu populasi boleh berkembang biak, ditot atau diklon, dan mati dalam beberapa saat. Perubahan ini berlaku secara berterusan. Kini, AG telah dikembangkan konsepnya ke dalam *Rangkaian Neural* (atau ringkasnya **RN**). Kertas ini membicarakan konsep atau proses evolusi yang digunakan didalam **RN**.

Kata kunci: vektor pemberat, nilai keupayaan, pemberat sinaptik, pincang, matriks pemarkahan, nilai keupayaan ternormalkan, pengeluaran-semula, menyilang, mutasi, generasi

PENGENALAN

Terdapat banyak takrif yang boleh diberikan kepada *Algoritma Genetik* (**AG**). Satu daripada takrif yang sering digunakan ialah satu teknik carian di ruang berdimensi luas, (Rogers, 1991) dan sering digunakan dalam masalah yang melibatkan pengoptimuman. Lihat juga Ramlan dan Azim (1995).

Teknik ini diilhamkan daripada proses pengevolusian DNA. Antara proses yang berlaku ialah ahli satu set rentetan perduaan berlumba-lumba untuk bersaing mendapatkan tempat dalam satu set rentetan yang baru. Penyatuan semula dilakukan dengan memilih dua ahli yang paling berjaya dalam populasi untuk dijadikan generasi datuk-nenek. Rentetan baru dicipta dengan menyambat (splicing) gen datuk-nenek masing-masing. Akhir sekali, satu rentetan baru akan diperoleh untuk menggantikan set rentetan yang lama, dan mana-mana rentetan lama yang tidak terpakai dibuang. Kini **AG** telah dikembangkan

konsepnya di dalam **RN**, (Tsoi 1994). Algoritma bagi **AG** seperti yang diutarakan oleh Tsoi (1994) diberikan oleh Rajah 1.

1. Mengawalkan nilai pemberat VP dengan nilai rawak.
2. Menilai VP sama ada sesuai dengan set latihan atau tidak.
3. Carian – Ulang sebanayak Gc kali.
   (a) Pilih VP yang boleh terus berguna.
   (b) Membentuk generasi baru VP dengan menggunakan operator-operator genetik seperti silang, atau mutasi, atau klon ke atas VP yang dipilih dalam (a).
   (c) Menilai VP yang dipilih itu sama ada sesuai dengan set latihan atau tidak.
4. Pembersihan – ulang sebanyak Gp kali.
   (a) Pilih VP yang boleh terus berguna.
   (b) Membentuk generasi baru VP untuk kegunaan generasi berikut dengan menggunakan klon sahaja ke atas VP yang dipilih dalam (a)
   (c) Menilai VP yang dipilih itu sama ada sesuai dengan set latihan atau tidak.

Akhirnya kita akan peroleh NI kelas VP, yang memberikan bilangan neuron untuk lapisan semasa.

5. Pilih satu perwakilan dari setiap kelas VP.
6. Bentukkan pula satu set latihan untuk lapisan yang berikut.
7. Ulang semula keseluruhan algoritma di atas dengan menggunakan set latihan yang terbentuk daripada langkah 6, jika terdapat lebih daripada satu kelas dalam langkah 5 (iaitu set latihan tidak dapat dikelaskan dengan hanya satu neuron) dan memerlukan satu lapisan lagi.

Rajah 1. Algoritma Genetik

Selain daripada memperoleh pemberat, algoritma yang diutarakan oleh Tsoi ini dikatakan mampu menentukan saiz suatu **RN**. Saiz yang dimaksudkan di sini ialah bilangan neuron yang ada pada lapisan tersembunyi. Lihat Khairuddin dan Ramlan (1996). Bahagian berikut memaparkan beberapa konsep yang digunakan oleh Tsoi untuk menghubungkan **AG** dan **RN**.

Bahagian berikut akan menjelaskan tatacara **AG** secara terperiinci. Ia diikuti dengan struktur seni bina **RN** Genetik. Seterusnya eksperimen bagi seni bina ini dijelaskan di bahagian berikutnya. Akhir sekali kesimpulan hasil eksperimen dipaparkan dan dibincang secara ringkas.

ALGORITMA GENETIK

AG melibatkan operasi ke atas satu set pemberat secara individu yang dinyatakan sebagai *Vektor Pemberat* (**VP**). **VP** akan mengandungi semua maklumat yang diperlukan untuk mendefinisikan satu neuron, iaitu pemberat untuk setiap input kepada neuron dan juga ambangnya. Satu neuron dengan I input dan satu ambang input akan didefinisikan dengan I+1 pemberat. Setiap pemberat disimpan dalam bentuk nombor nyata titik-tetap B bit, atau satu integer B-bit. Oleh itu **VP** didefinisikan sebagai (I+1)B bit. Lihat Tsoi (1994).

VP boleh dinyatakan secara matematik

$$y = f\left(\sum_{i=0}^{I} w_i \quad x_i\right) \qquad (1)$$

di mana y adalah output bagi neuron, w_i, $i = 0, 1, \ldots, I$ adalah *pemberat sinaptik* input, x_i, $i = 1, 2, \ldots, I$ adalah I input, dan $x_0 = 1$.

Satu fungsi tak linear $f(\alpha)$ dipilih iaitu

$$f(\alpha) = \begin{cases} 0 \text{ jika } \alpha < 0 \\ 1 \text{ jika } \alpha \geq 0 \end{cases}$$

Secara amnya **AG** melibatkan proses-proses memberikan nilai awal pemberat yang rawak, mengira nilai keupayaan, memilih **VP** untuk terus hidup, dan membentuk generasi baru. Bahagian berikut akan menjelaskan secara terperinci proses-proses tersebut.

AWALAN - MERAWAKKAN VP

Sebanyak (I+1)B bit **VP** disetkan dengan nilai rawak. Ini bermakna nilai pemberat adalah rawak dan disetkan dalam julat tertentu. Julat yang dipilih ialah antara -0.5 dan 0.5 berdasarkan saranan Fausett (1994).

PENGIRAAN NILAI KEUPAYAAN

Operasi ini adalah paling kritikal. Pengiraan *nilai keupayaan* dilakukan untuk setiap set pasangan pemberat dalam **VP** yang telah dipilih secara rawak. *Fungsi keupayaan* adalah seperti berikut:

$$\text{Keupayaan } VP_i = \sum_{i=1}^{T} \alpha_i \qquad (2)$$

$$_i = \begin{cases} 0 \text{ jika VP mengkelaskan vektor latihan i dengan salah,} \\ Pincang\,(n_i) \text{ jika VP mengkelaskan vektor latihan i dengan betul} \end{cases}$$

dengan

T = jumlah bilangan vektor dalam set latihan, dan

n_i = jumlah bilangan **VP** yang boleh mengkelaskan vektor latihan secara betul dan *Pincang(x)* adalah *fungsi pengurangan* secara monotonik untuk $x \geq 0$. Secara amnya, ia boleh dipilih secara bebas, sebagai contoh $Pincang(x) = \frac{1}{x^{\beta}}$, dan β adalah integer bukan-negatif.

Contoh, $Pincang(x) = \frac{1}{x^2}$, $Pincang(x) = \frac{1}{x^3}$.

Langkah selanjut boleh dijelaskan secara matematik seperti berikut:

Andaikan terdapat **T** pasangan input output dalam set latihan, dengan output: y_l^d, $l = 1, 2, ..., T$, dan input: x_i^l, $i = 1, 2, ..., I$; $l = 1, 2, ..., T$. Setiap neuron dijelaskan oleh **VP**. Katakan setiap neuron digambarkan oleh persamaan (1) seperti berikut:

$$y_{l,k} = f(\sum_{j=0}^{l} w_j^k x_j^l) k = 1,2,...,N; \quad l = 1,2,...,T. \tag{3}$$

dengan $y_{l,k}$, $l = 1, 2, ..., T$; $k = 1, 2, ..., N$ adalah output bagi N**VP** yang bertindak balas kepada vektor input ke l, w_j^k, $j = 1, 2, ..., I$; $k = 1, 2, ..., N$ adalah *pemberat sinaptik* yang menghubungkan input ke j kepada neuron ke k, dan $x_j^{(l)}$, $j = 1, 2, ..., I$; $l = 1, 2, ..., T$ adalah vektor input ke l ke lapisan berkenaan. Kesemuanya sejumlah T pasangan input output.

Ralat bagi output neuron ke l adalah

$$e_{l,k} = y_l^d - y_{l,k} = y_l^d - f(\sum_{i=0}^{I} w_j^k x_j^{(l)}). \tag{4}$$

Perlu diingatkan bahawa kita telah mengandaikan output itu perduaan, ralat $e_{l,k}$, $l = 1, 2, ..., T$; $k = 1, 2, ..., N$ kemungkinan bernilai 0 (pengkelasan betul) atau tak-sifar (pengkelasan salah). Untuk tujuan pengiraan markah, andaikan

$$e_{l,k} = \begin{cases} 0 & jika\ berada\ dalam\ kelas\ yang\ salah \\ 1 & jika\ berada\ dalam\ kelas\ yang\ betul. \end{cases} \tag{5}$$

Ralat boleh disusun dalam bentuk *matrik pemarkahan* $\boldsymbol{E} = [e_{l,k}, l = 1, 2, ..., T; k = 1, 2, ..., N]$.

Bertolak daripada rumus ini, n_l, $l = 1, 2, ..., T$ boleh dikira sebagai

$$n_l = \sum_{k=1}^{N} e_{l,k}. \tag{6}$$

Maka *fungsi keupayaan* bagi setiap **VP** boleh diperolehi sebagai

$$J(w_j^k, j = 0,2,...,I) = \sum_{\substack{l=1 \\ n_l^\beta \neq 0}}^{T} \frac{e_{l,k}}{n_l^\beta}, \tag{7}$$

dengan β adalah integer tak-negatif.

PILIHAN - MEMILIH VP UNTUK TERUS HIDUP

Tugas di sini hanyalah untuk tahap carian dan pembersihan. Saiz populasi semasa boleh dibinakan. Populasi ini boleh digunakan untuk menghasilkan generasi populasi seterusnya. Satu daripada pemberat dalam populasi awal boleh dipilih dengan menggunakan nilai kebarangkalian berikut:

Kebarangkalian memilih $VP_i = \frac{r_i}{\sum_{j=1}^{N} j} = \frac{2r_i}{N(N+1)}$.

dengan

VP_i = pemberat ke i dari populasi semasa,
N = jumlah bilangan **VP**,
dan r_i adalah kedudukan keupayaan pemberat i, yang menyatakan bahawa keupayaan yang tertinggi duduk di tempat tertinggi. Tiada dua pemberat mempunyai kedudukan yang sama.

Jumlahkan *nilai keupayaan* bagi kesemua **VP** menjadi *jumlah keupayaan*

$$J = \sum_{k=1}^{N} J\,(w_j^k, j = 1, 2, ..., I)$$

Kemudian, normalkan setiap keupayaan **VP** dengan *jumlah keupayaan* dan memperoleh

$$j_i = \frac{J\,(w_j^k, j = 1, 2, ..., I)}{J}$$

Akhir sekali magnitud relatif bagi *nilai keupayaan ternormalkan* j_i, $i = 1, 2, ..., N$ diisihkan dalam bentuk menurun supaya boleh memperoleh siri tak-menaik $r_1 \geq r_2 \geq ... \geq r_N$. Di sini kita peroleh *nilai keupayaan ternormalkan* r_i yang terisih. Seterusnya pemberat yang dikatakan terbaik akan disenaraikan untuk membentuk generasi baru dalam langkah yang akan dinyatakan di bahagian berikut.

PENGELUARAN SEMULA - MEMBINA VP BARU DARI VP LAMA

Terdapat tiga pilihan operasi terhadap **VP** lama supaya menghasilkan pemberat baru iaitu *pengeluaran-semula, menyilang, dan mutasi.* Proses-proses ini tidak akan dijelaskan disini. Lihat Rogers (1994); Tsoi (1994); Ramlan dan Azim (1995); dan Khairuddin dan Ramlan (1996).

Pembentukan set populasi pemberat baru atau **VP** baru ini akan membuka ruang semula untuk dinilai keupayaannya. **VP** baru ini akan membentuk generasi baru. Proses di sinilah dikatakan melatih **RN** berdasarkan **AG**. Latihan akan diteruskan sehinggalah terdapat pemberat yang dikatakan berkeupayaan

sebagai pemberat di dalam mana-mana lapisan pemberat. Perkara yang biasa dilihat ialah melihat tindak balas yang betul ke atas satu set atau lebih pemberat ke atas setiap pasangan corak input output yang digunakan.

Dengan menggunakan algoritma seperti yang telah dipaparkan didalam Rajah 1, satu set pemberat yang nilai genetiknya yang dikatakan berkeupayaan akan dipilih untuk mengawalkan pemberat dan pincang pada lapis atau lapis pemberat.

MEMBENTUK SET LATIHAN UNTUK LAPISAN BERIKUTNYA

Set latihan yang dipilih sebelum ini adalah penyelesaian bagi **RN** satu-lapis sahaja. Oleh itu, bagi **RN** multi-lapis penjanaan untuk lapis yang berikut adalah dengan merambat input bagi vektor latihan dalam set latihan semasa melalui lapisan ini dan akan menjadi input kepada neuron-neuron bagi lapisan yang berikutnya. Output yang dikehendaki untuk setiap vektor latihan mestilah sama dengan output yang dikehendaki untuk vektor latihan dalam set latihan semasa.

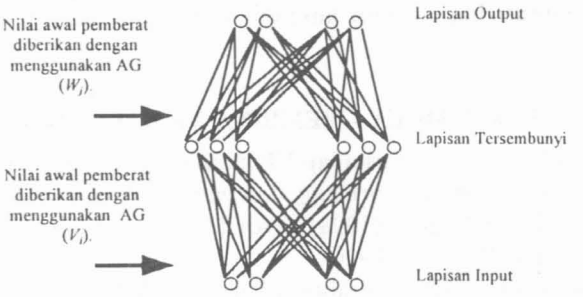
PARAMETER UTAMA

Berikut diberikan parameter-parameter yang diperlukan dalam **AG**:

- Bilangan bit per pemberat, G_b.
- Julat magnitud pemberat, G_r.
- Saiz Populasi, N - bilangan set pemberat secara individu.
- Kebarangkalian Menyilang, P_c - Kebarangkalian berlakunya proses menyilang.
- Kebarangkalian Mutasi, P_m - Kebarangkalian berlakunya mutasi.
- Bilangan Generasi untuk carian, G_s.
- Bilangan Generasi untuk pembersihan, G_c.

SENIBINA RANGKAIAN NEURAL GENETIK

Dalam kajian ini kita akan melihat penyelesaian pengkelasan aksara jawi dengan menggunakan rangkaian neural multi-aras yang perambatan-balik dipilih

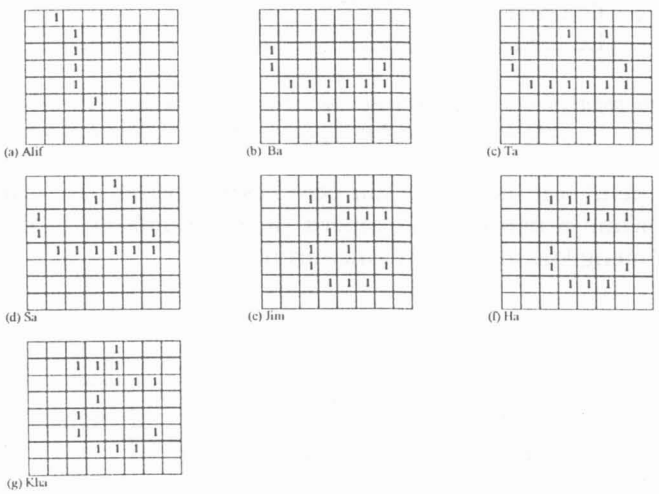


Rajah 2. Struktur genetik rangkaian neural

sebagai pengkelasnya, dan **AG** digunakan untuk mengawalkan nilai pemberat pada lapisan yang menghubungkan lapisan input ke lapisan tersembunyi dan lapisan tersembunyi ke lapisan output. Secara amnya senibina bagi rangkaian ini digambarkan di dalam Rajah 2.

Corak input dan tindak balasnya yang digunakan ketika latihan diwakilkan dalam bentuk perwakilan dwikutub supaya sesuai untuk diproses dalam rangkaian neural. Perwakilan dwikutub dikatakan mempunyai ciri-ciri yang sangat baik jika dibandingkan dengan perwakilan perduaan seperti yang dihuraikan oleh Fausett (1994).

Pertimbangkan huruf-huruf Jawi tunggal seperti yang digambarkan di dalam Rajah 3. Sebanyak tujuh corak input telah digunakan di dalam algoritma latihan. Kita akan menggunakan rangkaian ini untuk mengkelaskan setiap vektor input itu dipunyai atau tidak dipunyai bagi setiap tujuh kelas kategori.



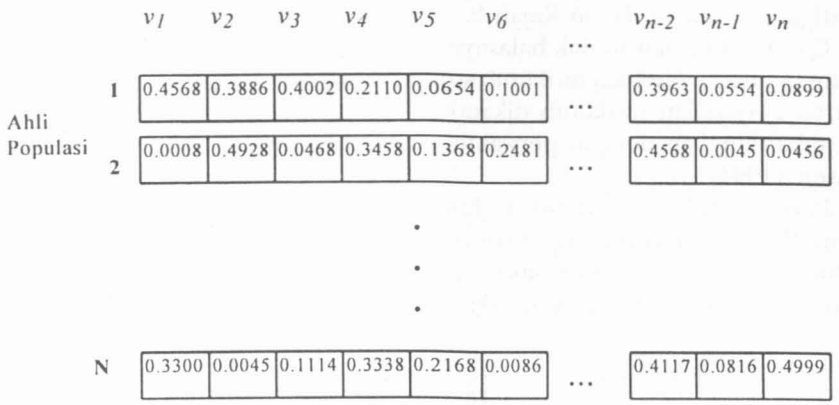
Rajah 3. Contoh Aksara Jawi yang direkabentuk pada Paparan Komputer

Rangkaian akan mempunyai tujuh unit output untuk mengkelaskan setiap unit input. Corak input tersebut boleh diwakil dalam bentuk vektor. Saiz corak piksel yang digunakan ialah 8x8, ini bermakna saiz vektor adalah hasil darab saiz corak tadi iaitu 64-tutupan. Lihat Ramlan dan Khairuddin (1996a).

Rangkaian ini bermula dengan memberikan satu populasi nilai awal pemberat yang rawak atau disebut sebagai *Vektor Pemberat* (**VP**), kemudian diikuti mengira nilai keupayaan **VP** tadi atau setiap ahli didalam populasi tersebut, seterusnya memilih **VP** untuk terus hidup, atau membentuk generasi baru.

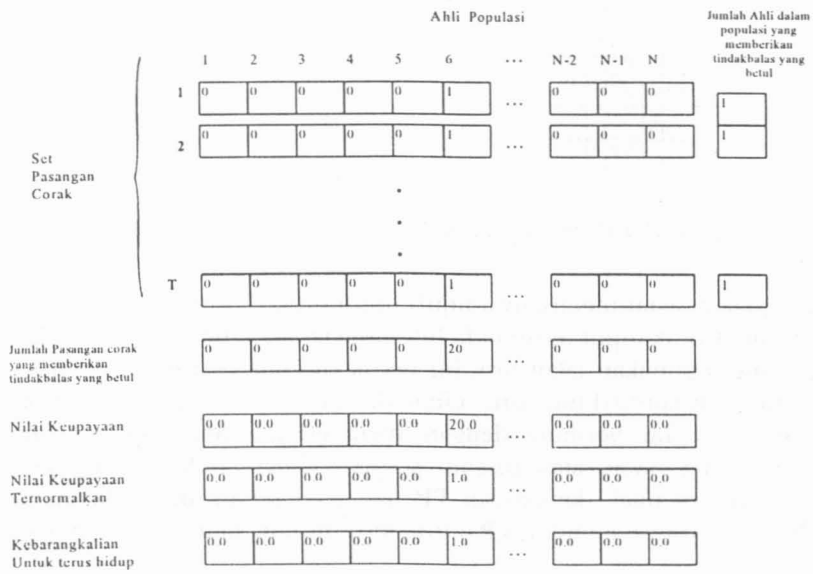
VP akan mengandungi semua maklumat yang diperlukan untuk mendefinisikan satu neuron, iaitu pemberat setiap input kepada neuron dan juga ambangnya. Satu neuron dengan I input dan satu ambang input akan didefinisikan dengan n = I+1 pemberat, (Tsoi, 1994). Dalam kajian ini I=64.

Nilai pemberat adalah rawak dan disetkan dalam julat tertentu. Julat yang dipilih ialah antara -0.5 dan 0.5. Lihat Rajah 4 berikut.



Rajah 4. Ahli populasi untuk penyelesaian percubaan bagi menentukan nilai awal pemberat yang menghubungkan lapisan input ke lapisan tersembunyi

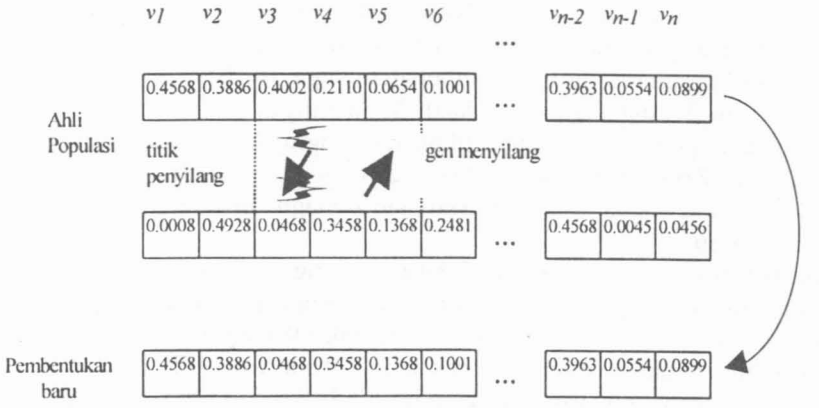
Setiap pemberat yang dijadikan penyelesaian percubaan akan ditentukan nilai kesesuaiannya dan direkod seperti yang ditunjukkan di dalam Rajah 5 serta diisih mengikut susun tertib menurun.



Rajah 5. Ahli populasi untuk penyelesaian percubaan bagi menentukan nilai kesesuaian untuk terus hidup

Pengiraan bermula dengan menentukan pengkelasan vektor sama ada salah atau betul bagi setiap pasangan corak input yang dijadikan sebagai corak latihan. Pengkelasan yang salah akan diberikan nilai 0 manakala pengkelasan yang betul akan diberikan suatu nilai fungsi pincang, lihat Tsoi (1994); Ramlan dan Khairuddin (1996b); Ramlan *et al* (1996). Seterusnya fungsi pincang ini akan digunakan dalam menentukan nilai keupayaan.

Proses yang paling penting di dalam **AG** ialah proses menyilang. Proses menyilang yang dimaksudkan disini ialah menentukan pemberat baru dengan memilih dua ahli di dalam populasi atau **VP** yang terbaik untuk dibiakkan dan dijadikan sebagai ahli baru di dalam populasi. Maksud dibiakkan di sini ialah menemberengkan kedua-dua ahli tadi pada satu atau dua titik menyilang seperti yang digambarkan pada Rajah 6 berikut.



Rajah 6. Ahli populasi terhasil daripada proses menyilang

Dengan menggunakan **AG** seperti yang telah dijelaskan di bahagian 2.0, nilai genetik yang dikatakan berkeupayaan akan dipilih untuk mengawal nilai pemberat dan nilai pincangnya.

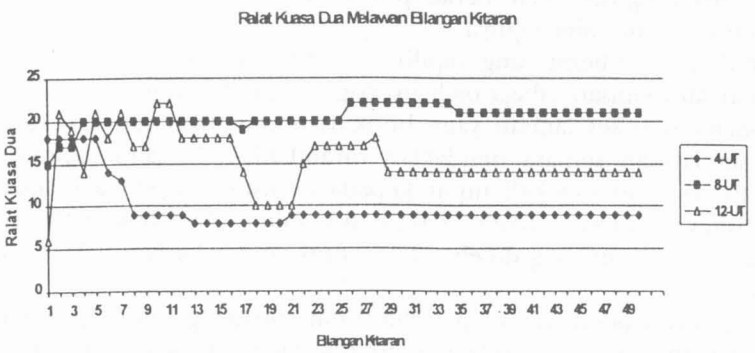
Populasi pemberat yang dipilih sebelum ini adalah penyelesaian bagi rangkaian satu lapisan sahaja oleh itu bagi rangkaian neural multi-aras untuk menjanakan satu set lapisan yang berikut, output bagi input vektor latihan dalam set latihan semasa hendaklah dirambatkan melalui lapisan ini dan seterusnya ia akan menjadi input kepada neuron-neuron bagi lapisan yang berikutnya. Output yang dikehendaki untuk setiap vektor latihan dibuat supaya sama dengan output yang dikehendaki untuk vektor latihan dalam set latihan semasa.

Akhir sekali pemberat yang terhasil dari proses genetik (pemberat pada lapisan yang menghubungkan lapisan input ke lapisan tersembunyi dan pemberat pada lapisan yang menghubungkan lapisan tersembunyi ke lapisan output) disimpan dan digunakan pada rangkaian neural multi-aras yang dilatih menggunakan perambatan-balik.

EKSPERIMEN

Seni bina rangkaian ini telah diimplemenkan pada mikrokomputer 486 serasi IBM dan dilarikan menggunakan bahasa C++. Bilangan unit tersembunyi (UT) yang digunakan ialah 4, 8, dan 12 untuk diaplikasikan kepada beberapa huruf jawi seperti yang telah dipaparkan dalam Rajah 3. Dalam hal ini sebanyak tiga ujikaji telah dijalankan iaitu ujian ke atas (i) 4-UT; (ii) 8-UT; dan (iii) 12-UT. Saiz populasi yang dipilih ialah 10 dan daripada sepuluh populasi tersebut, kami telah bahagikan kepada empat subpopulasi supaya senang diproses. Daripada empat subpopulasi tersebut 25% diperuntukkan untuk populasi terbaik, 25% populasi baru dan 50% diperuntukkan untuk populasi menyilang. Proses mutasi tidak dipertimbangkan. Menurut De et al (1996) mutasi boleh memberi kesan kepada penumpuan yang dikehendaki iaitu memperlahankan proses penumpuan dalam keadaan mana ia boleh menukarkan nilai bit terpenting dalam kromosom yang baru (zuriat baru) dan seterusnya memberikan nilai keupayaan yang tidak dikehendaki. Bilangan generasi untuk carian ialah 10, manakala generasi untuk pembersihan juga 10. Nilai parameter β (bagi fungsi pincang) yang digunakan ialah 3. Saiz input rangkaian ialah 64 dan 1 input pincang pemberat. Kadar latihan yang digunakan untuk kesemua latihan bernilai 0.2. Kriteria penamat bagi setiap uji kaji adalah 50,000 kitar; atau jumlah ralat kuasa-dua adalah bersamaan denagn nilai sifar; atau yang mana dicapai dahulu.

Berpandukan kepada Rajah 7 latihan dengan menggunakan nilai awal pemberat ini tidak dapat mengecam corak yang dipertimbangkan tadi. Untuk kes ini walaupun latihan telah mencecah 50,000 kitaran, seni bina rangkaian masih belum dapat mengenali corak tadi. Untuk kes 4-UT, jumlah kuasa-dua ralat pada kitaran 1,000 yang pertama bernilai 18.0 dan tidak berubah sehingga ke kitaran 5,000 dan beransur menyusut ke nilai 8.0 pada kitaran ke 13,000 dan meningkat sebanyak 1.0 pada kitaran berikutnya menjadikan 9.0 sehingga ke kitaran 50,000. Bagi 8-UT pula mempunyai ralat kuasa dua permulaan yang



Rajah 7. Latihan menggunakan pengawalan pemberat genetik untuk seni bina 4-UT, 8-UT, dan 12-UT
(Catatan: *Bilangan kitaran adalah dalam '0000)*

sedikit rendah berbanding dengan 4-UT (iaitu bernilai 15.0 dan meningkat ke nilai 22.0 pada kitaran 26,000 dan terus tidak banyak perubahan sehingga ke kitaran 50,000) tetapi masih gagal mengecam corak tadi. Bagi 12-UT pula penumpuan bertambah baik dan nilai kuasa dua ralat menjadi 6.0 pada kitaran ke 1,000 yang pertama dan bertambah secara mendadak pada 1,000 kitaran yang berikutnya dan terus meningkat pada kitaran berikutnya. Jumlah ralat tersebut berkurangan pada kitaran 18,000 hingga 21,000 kepada 10.0, dan kembali meningkat pada kitaran berikutnya dengan nilai konsisten iaitu 14.0.

KESIMPULAN

Daripada corak yang diberikan itu, seni bina ini masih gagal untuk mengecam aksara jawi. Nilai pemberat yang diperoleh daripada proses genetik itu masih gagal dan terperangkap di dalam minimum-tempatan pada titik kitaran yang ralat kuasa duanya tidak berubah, misalnya untuk 4-UT pada kitaran 20,000 hingga 50,000, manakala untuk 8-UT pada kitaran 35,000 hingga 50,000 dan untuk 12-UT pada kitaran 29,000 hingga 50,000.

Faktor-faktor utama yang menyebabkan kegagalan ini ialah bilangan saiz sampel data yang kecil iaitu sebanyak 1set sempel; dan bilangan unit tersembunyi masih jauh kecil dalam ketiga-tiga uji kaji yang telah dijalankan.

Kesimpulannya ialah penumpuan rangkaian neural genetik gagal menumpu apabila menggunakan saiz set sampel yang kecil serta bilangan unit tersembunyi yang kecil.

Penemuan yang sangat berguna dalam kajian ini bukan sahaja boleh memperoleh sat set pemberat rangkaian malah dapat mengetahui bagaimanakah **RN** boleh dilatih menggunakan **AG**. Perkara ini jelas dapat dilihat pada Bahagian 2.4 dan Bahagian 2.5.

BIBLIOGRAFI

De, S., Gosh, A., dan Pal, S. K. 1996. Fitness Evaluation in Genetic Algorithms with Ancestors' Influence. Dalam *Genetic Algorithms for Pattern Recognition*, ed. S. K. Pal dan Wang, P. P. Boca Raton Florida: CRC Press.

Khairuddin bin Omar dan Ramlan bin Mahmod. 1996. Genetik-Rangkaian Neural Untuk Pengkelasan Aksara Jawi. Dalam *Pascasidang National Conference on Research and Development in Computer Science and Its Applications (REDECS '96)* pada 26-27 Jun 1996. Anjuran Jabatan Sains Komputer, Universiti Pertanian Malaysia.

Ramlan bin Mahmod, dan Khairuddin bin Omar. 1996a. Analisis Pengawalan Pemberat Rangkaian Neural Perambatan-Balik untuk Pengecaman Aksara. Pracetak.

Ramlan bin Mahmod, dan Khairuddin bin Omar. 1996b. Genetik-Rangkaian Neural. Laporan Teknik SAK/TR-007/96.

Ramlan bin Mahmod, Khairuddin bin Omar dan Md. Nasir bin Sulaiman. 1996. Genetik-Rangkaian Neural. Dalam *Pascasidang National Conference on Research and Development in Computer Science and Its Applications (REDECS '96)* pada 26-27 Jun 1996. Anjuran Jabatan Sains Komputer, Universiti Pertanian Malaysia.

RAMLAN BIN MAHMOD dan ABDUL AZIM BIN ABDUL GHANI. 1995. Pengoptimuman Menggunakan Algoritma Genetik. Jabatan Sains Kumputer, Universiti Pertanian Malaysia, Serdang. Laporan Teknik SAK/TR-012/95.

ROGERS, D. 1994. Whether Prediction Using A Genetic Memory. Dalam *Neural Networks: Concepts, Applications, and Implmentations.* Vol. IV, 1994, ms. 275-289. New Jersey: Prentice-Hall.

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