PENGECAMAN HURUF JAWI TUNGGAL MENGGUNAKAN PENGELAS PERAMBAT BALIK

ABDUL AZIZ AZRI
KHAIRUDDIN OMAR

Fakulti Teknologi & Sains Maklumat, Universiti Kebangsaan Malaysia

ABSTRAK

Pengecaman Aksara Optikal atau OCR ialah kaedah mekanikal atau elektronik untuk menukar imej yang mengandungi teks, tulisan tangan dan teks cetakan mesin. OCR banyak diaplikasi digunakan untuk proses seperti komputer kognitif, pengecaman pertuturan(text-to-speech), data kekunci dan perlombongan teks. Pengelas perambat balik ialah algoritma latihan dan pembelajaran untuk komputer yang sering digunakan untuk pengkelasan OCR. Objektif utama kajian ini dilakukan ialah mengkaji kesan pengubahan fungsi kos, pengawal pemberat dan kadar pembelajaran terhadap ketepatan pengecaman. Selain itu, kesan penambahan bilangan nod tersembunyi kepada kadar ketepatan juga antara fokus utama kajian ini. Data set yang digunakan diperoleh dari pelajar sekolah menengah yang dinamakan KHAIR2017 dari pelbagai latar pendidikan, menulis semula huruf jawi. Jumlah set data ialah 73, mengandungi 2499 aksara jawi daripada pelbagai bentuk.

1 PENGENALAN

Pengecaman Aksara Optikal atau OCR ialah kaedah mekanikal atau elektronik untuk menukar imej yang mengandungi teks, tulisan tangan dan teks cetakan mesin. OCR banyak diaplikasi digunakan untuk proses seperti komputer kognitif, pengecaman pertuturan(text-to-speech), data kekunci dan perlombongan teks.

Terdapat beberapa jenis OCR contohnya:

i. Pengecaman Aksara optikal (OCR).
ii. Pengecaman perkataan optikal
iii. Pengecaman Aksara pintar
iv. Pengecaman perkataan pintar.

Contoh penggunaan OCR pada kehidupan ialah:

i. Kemasukkan data untuk dokumen perniagaan.
ii. Pengecaman nombor pendaftaran kenderaan automatik.
iii. Pengecaman poskod.
iv. Pengecaman jumlah yang ditulis pada cek bank.

Pengecaman jawi tunggal menggunakan pengelas perambat balik ialah proses melatih data baru dan membuat pengecaman aksara tunggal dan mencari kadar ketepatan. Antara proses yang dilakukan ialah membuat imbasan data baru dan penemberangan (*segmentation*). Seterusnya data tulisan tangan jawi yang sudah dibinari(binarization) disimpan di pangkalan data dan boleh digunakan sebagai bahan contoh untuk pengujian pengecaman oleh penyelidik.

2 PENYATAAN MASALAH

i. Kadar ketepatan pengecaman jawi.

ii. Bilangan titik, hingar dan ruang pada setiap aksara jawi memberi kesan kepada maksud perkataan.

iii. Kekurangan bahan dan pangkalan data tulisan tangan jawi.

3 OBJEKTIF KAJIAN

i. Mengkaji kesan mengubah fungsi kos, pengawal pemberat dan kadar pembelajaran kepada kadar ketepatan pengecaman.

ii. Mengkaji kesan menambah bilangan nod pada aras input dan aras tersembunyi kepada kadar ketepatan pengecaman.

iii. Membangunkan laman web yang menyimpan bahan dan pangkalan data mengandungi tulisan tangan jawi untuk pengujian pengecaman.

4 METADOLOGI

Metodologi yang digunakan ialah Kaedah Orientasi Penggunaan Semula (reuse-oriented). Penambahbaikkan dilakukan ke atas sistem sedia ada dan meningkatkan kadar ketepatan pengecaman.

Fasa 1 – Analisis Keperluan

Pada sesi permulaan untuk pembangunan perisian ini, pelbagai perancangan telah dilakukan. Antaranya tajuk sistem perisian, jenis sistem dan jenis sistem operasi yang akan dijalankan. Pada fasa ini juga, data tulisan tangan jawi dikumpul dari pelajar sekolah menengah untuk digunakan sebagai set latihan pengecaman. Selain itu carta gantt telah disusun supaya tempoh pembangunan sistem tersusun dan mengikut jadual yang telah ditetapkan.

Fasa 2 – Reka bentuk sistem

Sebelum mula perlaksanaan, seeloknya sistem direka bentuk dan dianalisis. Pelbagai perkara pelu dipilih contohnya penggunaan bahasa, perisian yang akan digunakan dan lakaran awal & akhir sistem yang akan dilaksana. Anataramuka sistem dan laman web direka bentuk supaya

lebih mesra pengguna dan mudah digunakan untuk pengguna permulaan. Set Data tulisan jawi perlu diimbas kepada format JPEG dan ditukarkan kepada bitmap dan seterusnya kepada format matlab.

Fasa 3 – Perlaksanaan

Pada fasa ini, data set tulisan jawi yang telah di terjemah kepada format matlab perlu dilatih. Jumlah Cost function, initial weight dan alpha juga diubah untuk menambah kadar ketepatan yang lebih tinggi. Nod pada lapisan tersembunyi dan lapisan input juga diubah dan diuji untuk mencari kadar ketepatan yang lebih tinggi. Pada fasa ini juga antaramuka dan fungsi sistem perisian ditulis mengikut lakaran akhir yang dilakukan pada fasa reka bentuk sistem.

Fasa 4 – Pengujian

Pada fasa ini, pengujian yang lebih kerap perlu dilakukan supaya tiada masalah ketirisan apabila pengguna akhir menggunakan perisian. Pengujian boleh dilakukan terhadap pakar pengecaman karakter dan pengguna biasa. Lebihan data set boleh digunakan untuk fasa pengujian pengecaman karakter.

Fasa 5 – Penggunaan

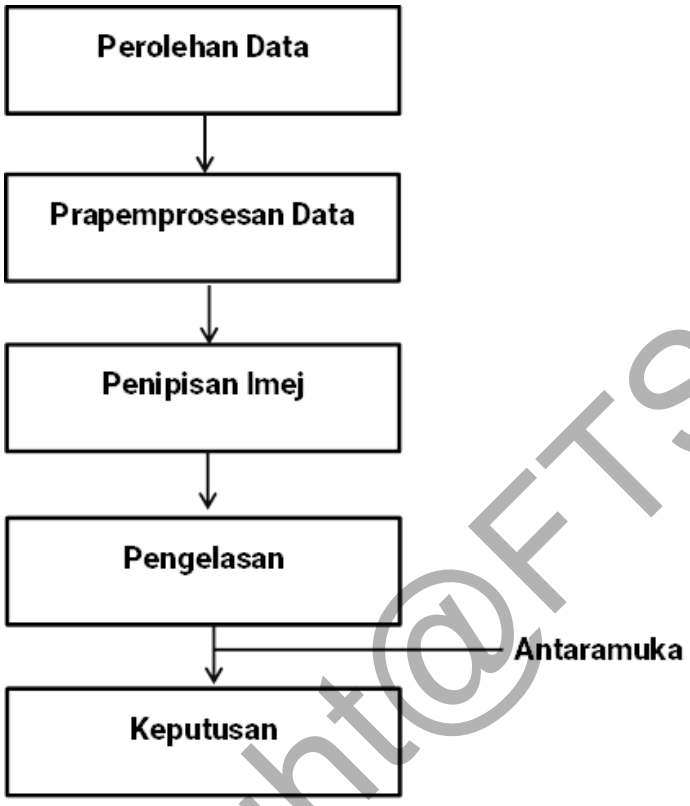
Pada fasa ini adalah fasa terakhir untuk dilancarkan kepada pengguna. Manual pengguna perlu disertakan agar memudahkan pengguna.

Fasa 6 – Penyelenggaraan

Fasa penyelenggaran adalah fasa untuk memperbaiki jika terdapat pepijat pada sistem perisian. Jika terdapat penambah baikkan fungsi dan ciri – ciri perisisan, boleh menyertakan pengemaskinian dan sertakan pemberitahuan log perubahan (changelog).

5 HASIL KAJIAN

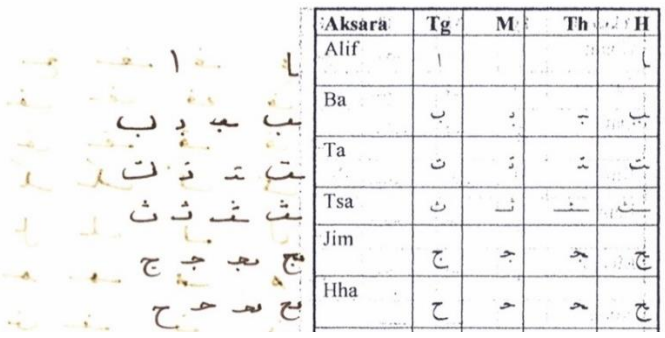
5.1 RANGKA KERJA



Rajah 5.1 Menunjukkan rangka kerja proses pengecaman aksara jawi

5.1.1 Perolehan data

Data diperoleh dari pelajar sekolah menengah. Pelajar dikehendaki menulis semula semua huruf jawi tunggal termasuk pecahan disebabkan aksara jawi mempunyai pecahan awal, tengah hujung dan tunggal.



Rajah 5.2 Contoh keratan set data

5.1.2 Prapemprosesan data

Setiap set data akan diimbas dan kemudiannya akan dipotong mengikut jenis huruf jawi dengan format *bitmap*(.bmp). Setiap aksara akan diubah saiz kepada 20 x 20 piksel untuk memudahkan proses pindahan ke format matlab atau octave. Kemudian imej aksara akan ditukar kepada *grayscale*.

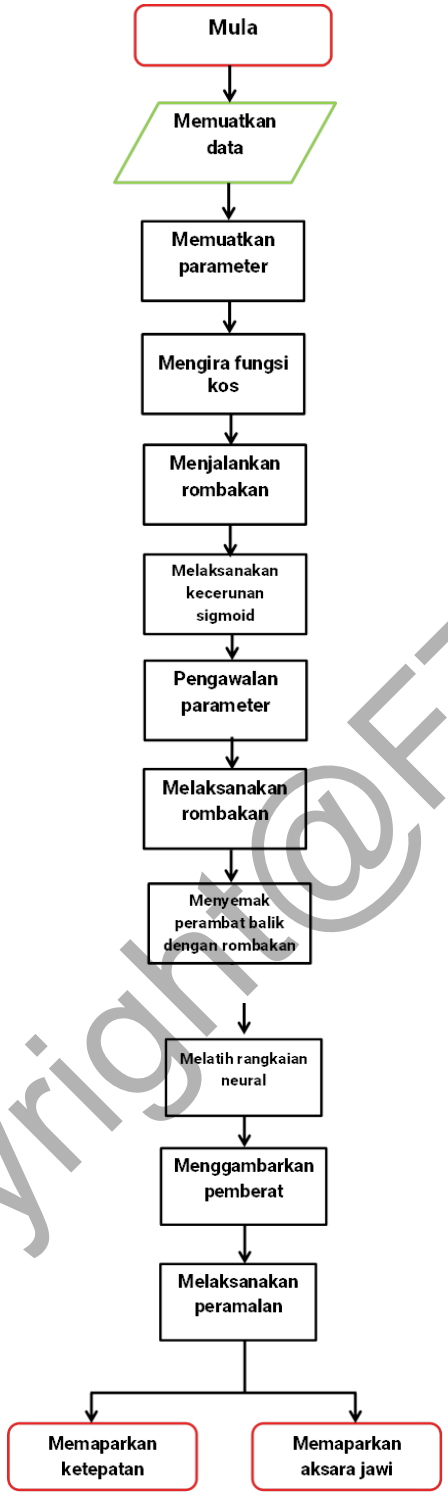
Rajah 5.3 Aksara jawi yang telah ditukar ke grayscale dengan format bitmap(.bmp)

5.1.3 Penipisan imej

Imej yang bergrid 20 sebanyak 20 piksel akan divektorkan menjadi 400 dimensi matriks. Setiap satu daripada contoh latihan menjadi satu baris dalam matriks. Ini memberikan 1 5000 oleh 400 matriks x di mana setiap baris adalah contoh latihan untuk imej digit tulisan tangan.

Bahagian kedua set latihan adalah 15000 dimensi vektor y yang mengandungi label untuk set latihan. Untuk membuat perkara yang lebih serasi dengan pengindeksan Octave/ MATLAB, di mana tidak ada indeks sifar, kami telah dipetakan angka sifar kepada nilai 36. Oleh itu, "0" digit dilabelkan sebagai "alif" sehingga 36 label "ya" merujuk susunan aksara jawi.

5.1.4 Pengkelasan aksara menggunakan rangkaian neural



Rajah 5.5 Carta alir proses pengkelasan menggunakan rangkaian neural

I. Memuatkan data

Data yang sudah diproses akan dipanggil dan 100 contoh data akan dipaparkan.

II. Memuatkan parameter

Pemberat neural network akan dimuatkan ke pemboleh ubah theta1 dan theta2.

III. Mengira fungsi kos

Kepada rangkaian neural, pertama harus bermula dengan melaksanakan suap depan bahagian rangkaian neural yang mengembalikan kos sahaja. Kos fungsi untuk suap depan dilaksanakan tanpa rombakan dahulu supaya ia menjadi lebih mudah untuk dinyahpepijat.

Fungsi kos bagi rangkaian neural (tanpa rombakan) adalah:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log((h_\theta(x^{(i)}))_k) - (1-y_k^{(i)})\log(1-(h_\theta(x^{(i)}))_k)\right]$$

IV. Melaksanakan rombakan

Fungsi kos untuk rangkaian neural dengan rombakan diberi:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log((h_\theta(x^{(i)}))_k) - (1-y_k^{(i)})\log(1-(h_\theta(x^{(i)}))_k)\right] + \frac{\lambda}{2m}\left[\sum_{j=1}^{25}\sum_{k=1}^{400}(\Theta_{j,k}^{(1)})^2 + \sum_{j=1}^{10}\sum_{k=1}^{25}(\Theta_{j,k}^{(2)})^2\right].$$

V. Melaksanakan kecerunan sigmoid

Sebelum bermula melaksanakan rangkaian neural, kecerunan bagi fungsi sigmoid perlu dilaksanakan dahulu. . Kecerunan untuk fungsi sigmoid boleh dikira sebagai:

$$g'(z) = \frac{d}{dz}g(z) = g(z)(1-g(z))$$

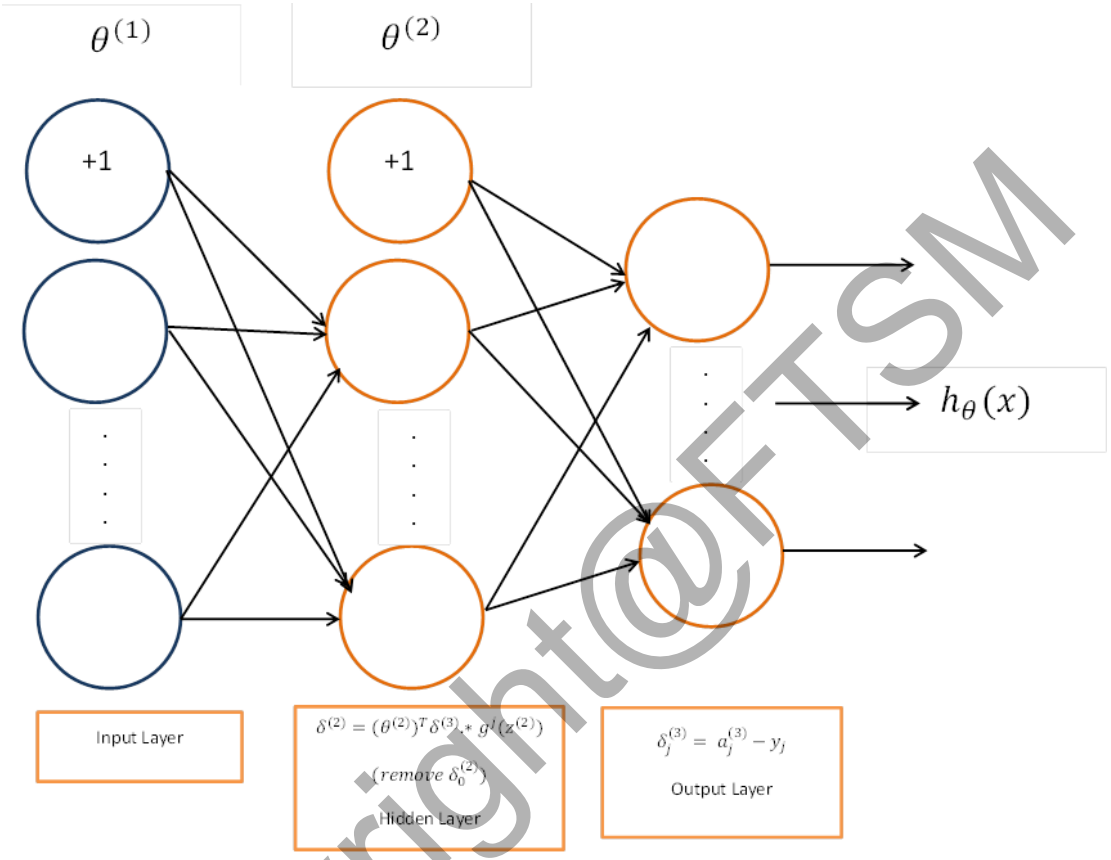
Dimana,

$$\text{sigmoid}(z) = g(z) = \frac{1}{1+e^{-z}}.$$

VI. Pengawalan parameter

Untuk memulakan rangkaian neural dua lapisan, fungsi untuk memulakan pemberat bagi rangkaian neural yang mengklasifikasikan aksara akan dilaksanakan dahulu.

VII. Melaksanakan perambat balik



Rajah 5.5 Kemaskini perambat balik

Rambatan balik adalah satu ungkapan untuk terbitan ∂C / ∂w separa fungsi kos C berkenaan dengan apa-apa berat w (atau bias b) dalam rangkaian. Ungkapan memberitahu kita berapa cepat kos berubah apabila kita menukar berat dan bias. Rambatan balik bukan sahaja satu algoritma cepat untuk pembelajaran. Ia sebenarnya memberikan kita pandangan terperinci mengenai cara mengubah berat dan berat sebelah perubahan tingkah laku keseluruhan rangkaian.

VIII. Melaksanakan rombakan sekali lagi

Setelah pelaksanaan rambatan balik adalah betul dan berjaya dilaksanakan, kos dan kecerunan rombakan boleh dilakukan.

IX. Melatih rangkaian neural network

Pada fasa ini, rangkaian neural akan dilatih menggunakan fungsi “fmincg”. pengoptimal lanjutan mampu melatih fungsi kos dengan cekap selagi kita menyediakan mereka dengan pengiraan kecerunan

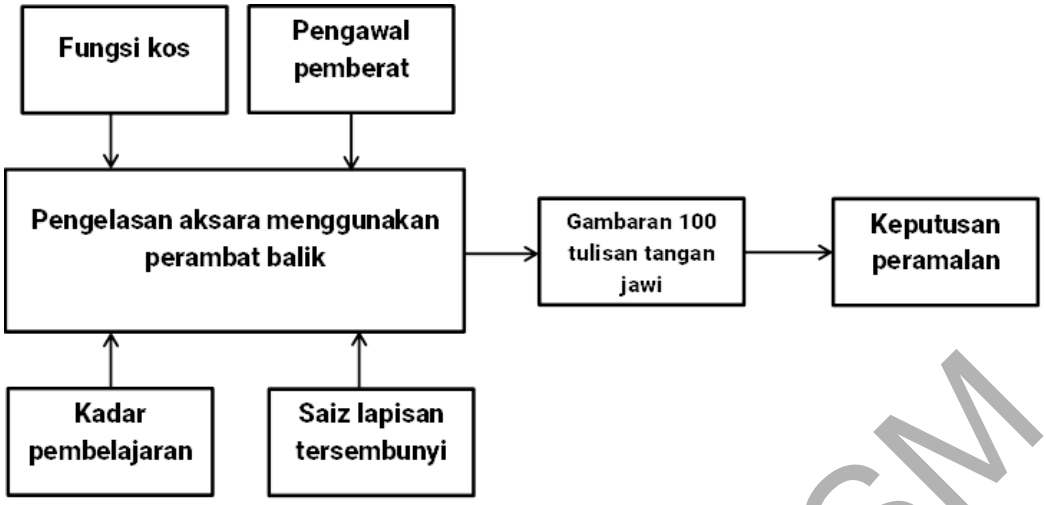
X. Menggambarkan pemberat

Anda kini boleh "menggambarkan" apa rangkaian neural adalah belajar dengan memaparkan unit tersembunyi untuk melihat apa ciri-ciri mereka menangkap dalam data.

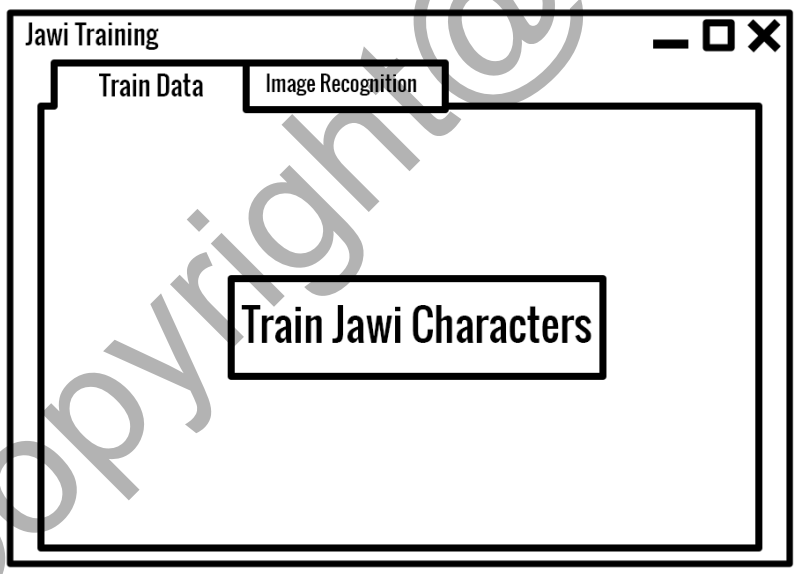
XI. Melaksanakan peramalan

Selepas melatih rangkaian neural, kami ingin menggunakannya untuk meramal label. Anda kini akan melaksanakan fungsi "meramalkan" untuk menggunakan rangkaian neural untuk meramalkan label set latihan. Ini membolehkan anda mengira ketepatan set latihan.

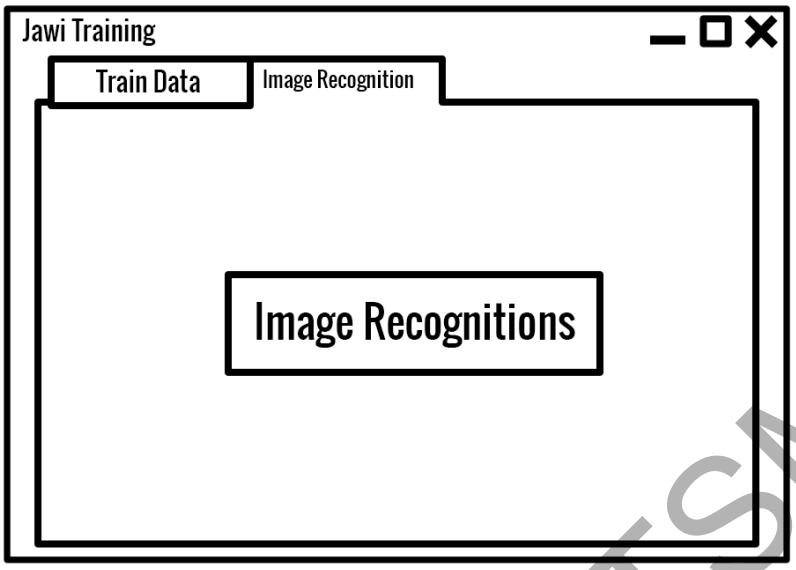
5.2 HASIL REKAAN ANTARAMUKA



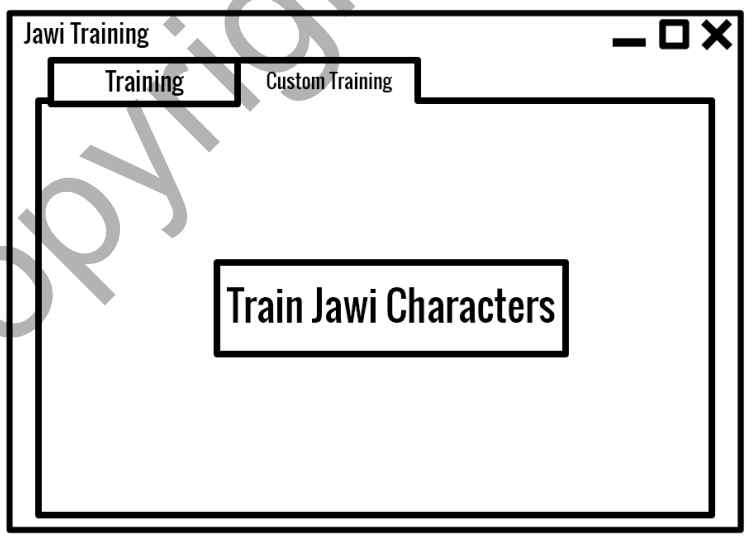
Rajah 5.6 Senibina perisian



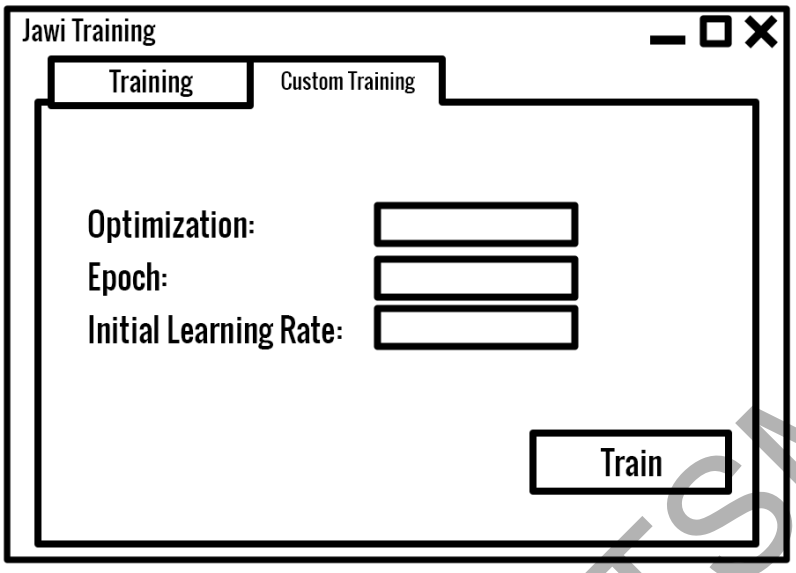
Rajah 5.7 Antaramuka utama perisian



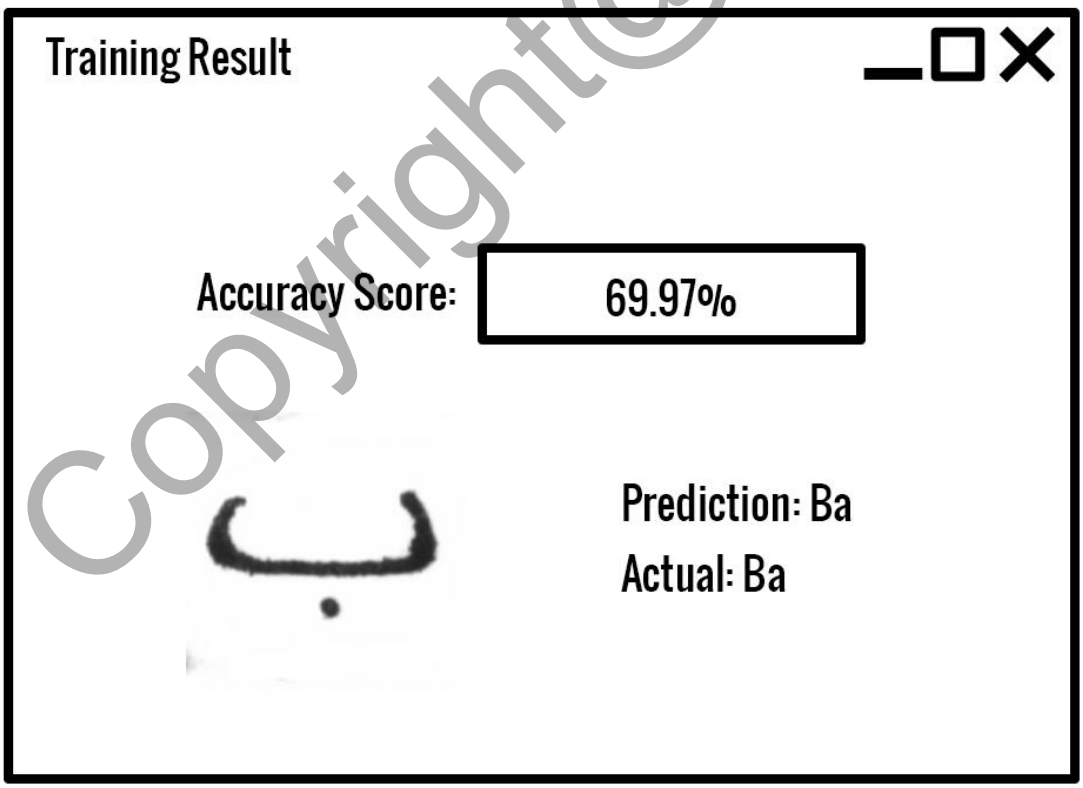
Rajah 5.8 Antaramuka Untuk Pengecaman Aksara



Rajah 5.9 Melatih Huruf Jawi Secara Keseluruhan



Rajah 5.10 Antaramuka membuat perubahan pada nilai parameter



Rajah 5.11 Antaramuka keputusan pengujian dan kadar ketepatan

5.3 KETERANGAN UJIKAJI

Pengujian telah dilakukan dengan dua bahagian: melatih data menggunakan fungsi *logistic* dan yang kedua menggunakan dan fungsi logistik + *regularization*. Kedua – dua bahagian menggunakan fungsi kos yang sama iaitu *logistic sigmoid* dan 3 jenis pengoptimum yang digunakan secara berturut iaitu kaedah *Quasi-Newton, Stochastic Gradient Descent* (SGD) dan *Adaptive Moment Estimation* (Adam). Bezanya latihan dan pengujian menggunakan fungsi logistik + *regularization* menggunakan parameter yang ditetapkan dengan nilai 0.01.

Rajah 5.12 Aksara jawi yang sudah ditukar ke *grayscale*

Sebanyak 2499 aksara jawi tulisan tangan digunakan dan disimpan sebagai pembolehubah X dan 2499 label untuk pembolehubah y. Kesemua data ini diambil dari 73 set data aksara jawi tunggal yang ditulis oleh 73 pelajar yang mempunyai latar belakang pendidikan yang berbeza. Pelajar tersebut juga diberi kebebasan untuk memilih mata pen untuk tujuan tulis semula aksara.

5.3.1 Data latihan

Kesemua 2499 aksara jawi akan dipotong menjadi bentuk segi empat sama. Kemudian, kesemua aksara akan ditukar kepada *grayscale* dan diubah saiz kepad 20 x 20 piksel menggunakan algoritma yang dihasilkan sendiri. Kesemua algoritma yang digunakan diimport dari ‘scikit-kit learn’, sebuah pustaka khusus untuk tujuan penyelidikan pembelajaran mesin. Sebelum dijadikan set data, setipa aksara ini akan divektor semula menjadi 400.

```
[ 254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  253.  254.  253.  233.  232.  253.  254.  254.  253.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  253.  254.  236.
  104.   95.  231.  254.  254.  252.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  241.  128.   96.  227.  254.
  252.  252.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  243.  233.  252.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  241.  220.  240.
  251.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  236.  129.   61.   90.  157.  245.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  179.   49.   22.   19.   64.  226.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  177.   56.   38.   47.
   45.  199.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  245.  186.  174.  168.   61.  167.  253.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  253.  205.   63.  157.  253.  254.  254.  254.  254.  254.
  254.  254.  254.  217.  217.  253.  254.  254.  254.  254.  238.  125.
   55.  188.  254.  254.  254.  254.  254.  254.  254.  254.  254.  186.
  113.  224.  252.  254.  252.  236.  153.   58.  131.  242.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  240.  139.  100.  169.  203.
  157.   99.   72.  121.  231.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  237.  166.   98.   89.   73.  102.  161.  234.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  251.  235.  225.  220.  241.  252.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.  254.
  254.  254.  254.  254.]
```

Rajah 5.13 Satu aksara yang ditukar ke *array* 400

Kesemua 2499 akan dimuatkan ke dalam *Hierachical Data Format* (HDF5) untuk dijadikan set data. Seiring dengan data aksara jawi, label untuk aksara jawi juga dimuatkan kedalam HDF5 berdasarkan susunan data aksara jawi. Label yang disertakan adalah dalam bentuk nombor 1 sehingga 36 mewakili Alif sehingga Va.

Huruf	Label	Huruf	Label
Alif	1	Ghain	19
Ba	2	Fa	20
Ta	3	Qaf	21
Tsa	4	Kaf	22
Jim	5	Lam	23

Hha	6	Meem	24
Kho	7	Noon	25
Dal	8	Waw	26
Zal	9	Ha	27
Ra	10	Lamalif	28
Zai	11	Hamzah	29
Seen	12	Ya	30
Sheen	13	Nga	31
Sad	14	Pa	32
Dhad	15	Ga	33
Tta	16	Nya	34
Dha	17	Cha	35
Ain	18	Va	36

Jadual 5.1 Senarai Huruf dan Label

	Aksara	Label
Set Data Latihan	2369	130
Set Data Ujian	2369	130

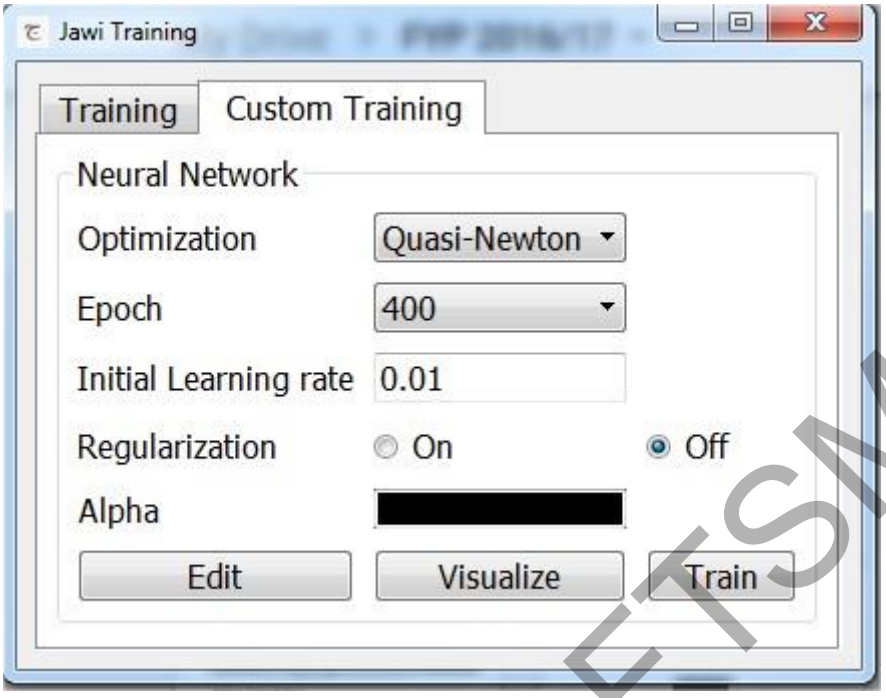
Jadual 5.2 Jumlah Set data yang diasingkan ke data latihan dan ujian

Sebelum data dimuatkan ke dalam pengelas, set data aksara akan diselarikan dalam lingkungan 0-1 manakala data label ditukarkan ke nombor binari. Seterusnya kedua set data akan dipisah secara rawak sebanyak 0.052 kepada data latihan dan data pengujian. Hasilnya 2369 diambil untuk latihan manakala 130 data aksara untuk pengujian. Jumlah yang sama juga

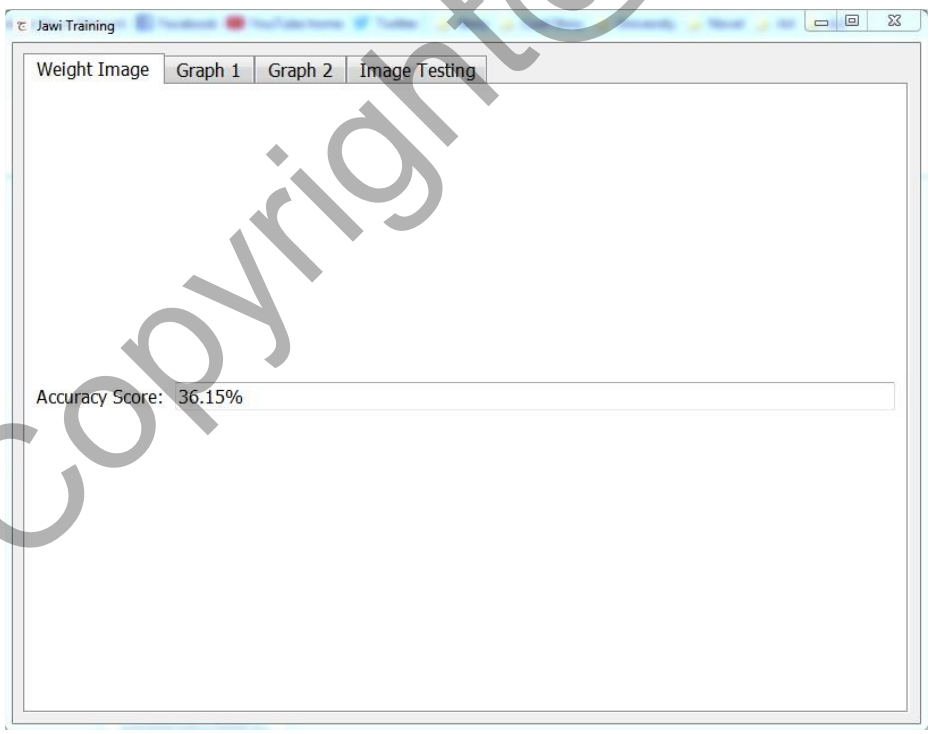
Seterusnya adalah data latihan akan dimuatkan kedalam pengelas *MultiLayer Perceptron* (MLP) untuk tujuan pembelajaran. Jenis parameter yang digunakan untuk pembelajaran ialah jenis pengoptimum, saiz lapisan tersembunyi, jenis fungsi pengaktifan, kadar pembelajaran dan pengawalan kadar pembelajaran. Jenis pengoptimum, saiz lapisan tersembunyi dan kadar pembelajaran ditetapkan kepada ‘Logistic’, 100 dan ‘constant’. Latihan dilarikan mengikut urutan fungsi pengaktifan logistic menggunakan pengoptimum *Quasi-Newton*, SGD dan Adam. Latihan diteruskan dengan mengaplikasi fungsi pengaktifan logistic + *regularization* dan pengoptimum seperti fungsi pengaktifan ‘logistic’ mengikut turutan. Proses melarikan latihan dijalankan sebanyak 400 kali (epochs).

5.3.2 Pengujian data

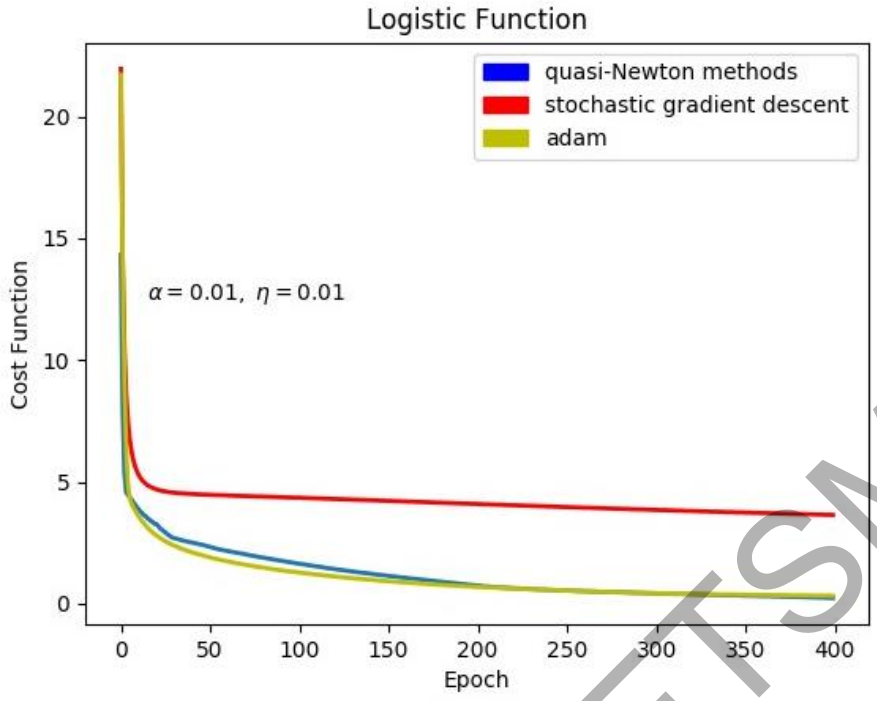
Setelah set data latihan dimuatkan kedalam pengelas MLP, set data ujian akan diuji untuk mendapatkan kadar ketepatan bagi setiap jenis fungsi kos dan jenis pengoptimum. Kadar ketepatan dihantar dan dicetak untuk mengukur sejauh mana keberkesanan data latihan. Nilai kehilangan juga disimpan untuk dijadikan graf untuk mengukur keberkesanan diantara pengoptimum yang diguna pakai.



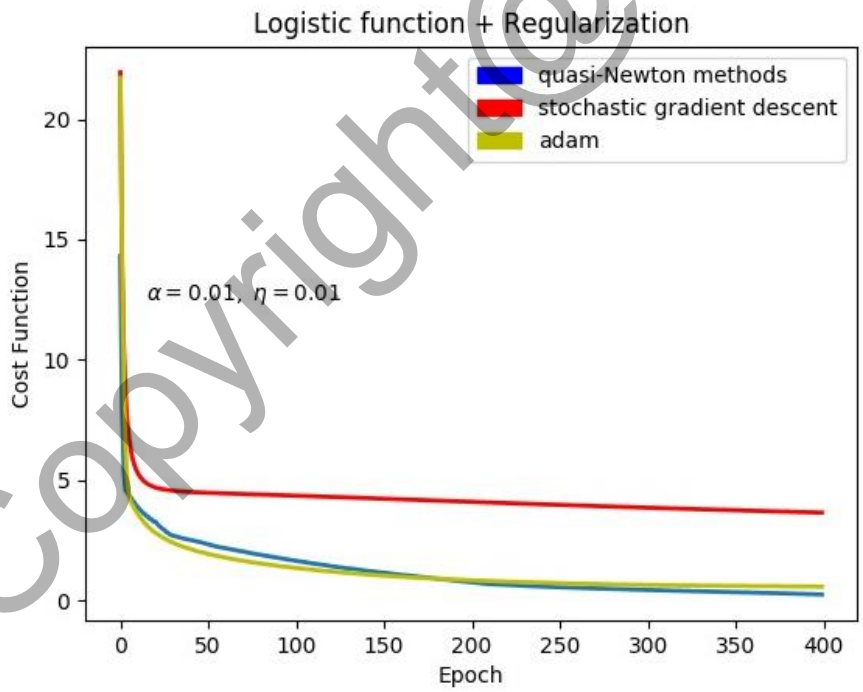
Rajah 5.14 Input untuk Quasi-Newton, Epoch dan Learning Rate



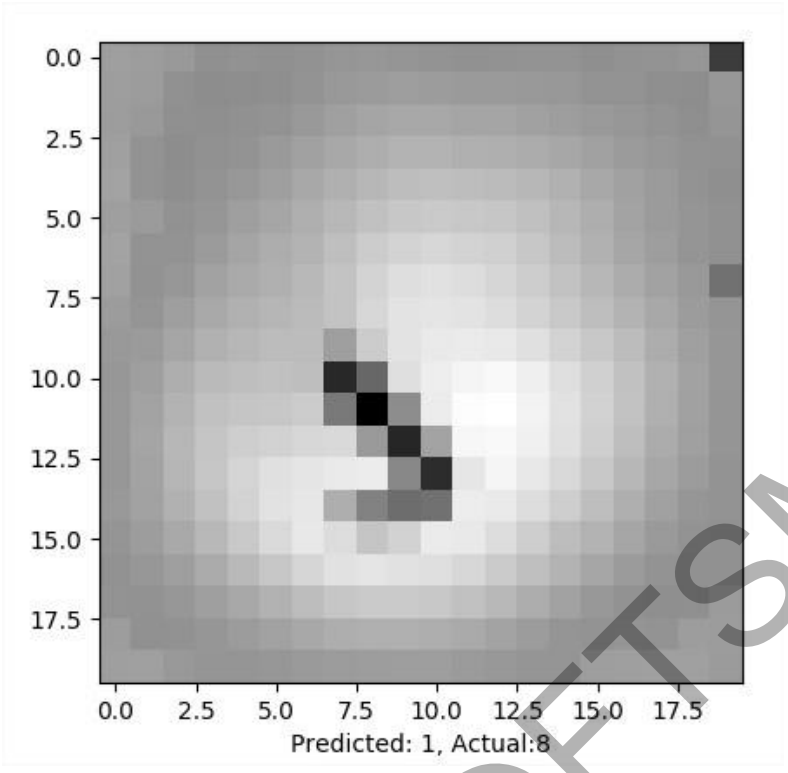
Rajah 5.15 Kadar ketepatan yang Berjaya dicapai



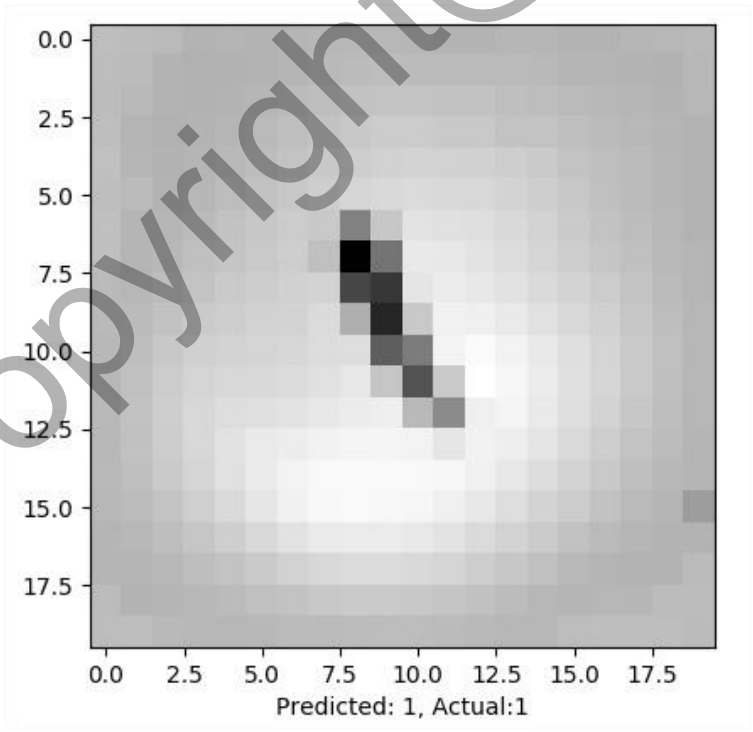
Rajah 5.16 Graf Fungsi Logistik yang dihasilkan



Rajah 5.17 Graf Fungsi Logistik + *Regularization*



Rajah 5.18 Peramalan Terhadap set data ujian yang tidak tepat



Rajah 5.19 Peramalan Terhadap set data ujian yang tepat

5.4 KEPUTUSAN UJIKAJI

Sebanyak 4 set uji kaji dilakukan menggunakan kadar pembelajaran pengawalan yang berbeza bermula dari 0.001, 0.01, 0.1 dan 0.9. Kadar ketepatan yang direkodkan bermula dari 0% sehingga 50%. Pengujian dilakukan dengan data ujikaji yang berbeza berdasarkan data yang dipisah secara rawak untuk mendapatkan keputusan yang mencapai objektif kajian.

		KOS LOGISTIK			
		η= 0.001	η= 0.01	η= 0.1	η= 0.9
PENGOPTIMUM	Quasi – Newton	35.38%	50%	34.61%	43.07%
	Stochastic Gradient Descent	0%	27.30%	36.15%	36.15%
	Adam	31.53%	42.30%	26.61%	7%

Jadual 5.3 Keputusan kadar ketepatan terhadap jenis pengoptimuman yang berbeza serta parameter yang berbeza

		FUNGSI LOGISTIC + REGULARIZATION			
		η= 0.001 α = 0.01	η = 0.01 α = 0.01	η = 0.1 α = 0.01	η= 0.9 α = 0.01
PENGOPTIMUM	Quasi – Newton	35.38%	50%	34.61%	43.07%
	Stochastic Gradient Descent	0%	41.62%	36.92&	40.7%
	Adam	37.69%	40%	29.23%	0%

Jadual 5.4 Keputusan kadar ketepatan terhadap jenis pengoptimuman yang berbeza serta parameter yang berbeza dengan menggunakan Kos Fungsi + *Regularization*

Keputusan ujikaji ini menunjukkan bahawa kadar pengesanan hanya mencapai 50% sahaja. Ini membuktikan bahawa kadar pengesanan aksara jawi mencapai objektif walaupun kadar ketepatan yang paling tinggi hanyalah 50%.

5.5 ANALISIS KEPUTUSAN

Daripada keputusan yang diperolehi melalui kesemua larian yang dijalankan, keputusan kadar ketepatan yang berjaya dicapai adalah berada kadar 50% yang paling tinggi. Tetapi keputusan ini tidak boleh dibandingkan dengan keputusan yang diperolehi oleh kajian lain contohnya oleh Jabrail Ramdan kerana kajian tersebut dilakukan menggunakan set data tulisan arab.

Graf fungsi kos pada Rajah 4.7 menunjukkan kadar pembelajaran yang dilakukan pada set data pada keseluruhan epochs dijalankan dengan baik untuk pengoptimum Quasi-Newton dan Adam tetapi pengoptimum SGD mencapai kadar pembelajaran yang sangat perlahan selepas epoch ke – 25.

Jika dapat dilihat dari Jadual 4.1, jenis pengoptimum Quasi-Newton dan kadar pembelajaran 0.01 menghasilkan kadar ketepatan yang sangat tinggi berbanding jenis pengoptimum SGD dan Adam. Jumlah yang sama juga terhasil selepas mengaplikasikan regularization pada Jadual 4.2 .

6 KESIMPULAN

Rangkaian neural buatan adalah satu sistem memproses maklumat yang mempunyai ciri-ciri yang sama dengan rangkain neural biologi. Ia telah dibangunkan sebagai perwakilan am bagi model matematik manusia. Algoritma Rambatan balik adalah satu algoritma latihan atau pembelajaran untuk komputer. Ia belajar melalui pengalaman lalu.

Berdasarkan keputusan ujikaji yang diperoleh melalui kesemua larian yang dijalankan, keputusan kadar ketepatan yang berjaya dicapai adalah berada kadar 50% yang paling tinggi. Tetapi keputusan ini tidak boleh dibandingkan dengan keputusan yang diperolehi oleh kajian lain kerana masih kurang pengkaji menggunakan set data.

6.1 RUMUSAN KAJIAN

Mengesan huruf jawi merupakan tugas yang mencabar dan menjadi permasalahan yang dibincangkan. Dalam kajian ini, pengesanan aksara huruf jawi menggunakan pengelas perambat balik menggunakan set data yang belum masih digunakan dan menjadikan skrip bahasa Python sebagai bahasa pengaturcara utama. Selain itu pustaka PyQT digunakan untuk menghasilkan GUI dan Scikit-Learn untuk mengaplikasikan algoritma pembelajaran mesin.

Keputusan ujikaji yang terhasil menunjukkan kajian ini memberikan keputusan kadar ketepatan yang kurang baik, namun pengaplikasian pra-pemprosean data bukan sesuatu tugas yang mudah kerana pemilihan data yang salah bakal menentukan pengkelasan yang salah

7 RUJUKAN

ABBY:What is an OCR technology: https://www.abbyy.com/en-apac/finereader/aboutocr/what-is-ocr/ [18 Oktober 2016]

Jabril Ramdan, Khairuddin Omar, Mohammad Faidzul, Ali Mady. 2013. Arabic Handwriting Data Base for Text Recognition. *The 4th International Conference on Electrical Engineering and Informatics (ICEEI 2013)*: 580-584.

Iping Supriana, Albadr Nasution. 2013. Arabic Character Recognition System Development. *The 4th International Conference on Electrical Engineering and Informatics (ICEEI 2013)*: 334-341.

Roopali Gupta, Prof. Neeraj Shukla. 2012. Character Recognition using Back Propagation Neural Network. *International Journal of Digital Application & Contemporary Search*. ISSN: 2319-4863

Micheal Nielsen: http://neuralnetworksanddeeplearning.com/chap2.html [Januari 2016]

Honey Mehta, Sanjay Singla, Aarti Mahajan. 2016. Optical character recognition (OCR) system for Roman script & English language using Artificial Neural Network (ANN) classifier. *2016 International Conference on Research Advances in Integrated Navigation Systems (RAINS)*. 10.1109/RAINS.2016.7764379

M.D. Ganis, C.L. Wilson, J.L. Blue. 2002. Neural network-based systems for handprint OCR applications. *IEEE Transactions on Image Processing*. p.p 1097 – 1112.

Christos Stergiou, Dimitrios Siganos. Neural Networks. https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html. [2007]

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors, *Nature*, Vol 323 L. Bottou. Stochastic Gradient Descent. http://leon.bottou.org/projects/sgd. [2014]

Andrew Ng, Jiquan Ngiam, Chuan Yu Foo, Yifan Mai, Caroline Suen. http://ufldl.stanford.edu/wiki/index.php/Backpropagation_Algorithm [2011]

Y. LeCun, L. Bottou, G. Orr, K. Müller. Efficient Backprop. Neural Networks: Tricks of the Trade. [1998]

Kingma, Diederik, and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations 2015*, arXiv preprint arXiv:1412.6980