Pengenalan Pola Tulisan Tangan Huruf Sasak Menggunakan Metode Integral Projection dan Neural Network

(Handwritten Sasak Ancient Script Recognition using Integral Pojection and Neural Network)

Eka Dina Juliani Utari Ms*, I Gede Pasek Suta Wijaya, dan Fitri Bimantoro
Prodi. Teknik Informatika, Fakultas Teknik, Universitas Mataram
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA.
Email: ekadinajuliani@gmail.com, gpsutawijaya@unram.ac.id, bimo@unram.ac.id,

Penulis korespondensi

***Abstract* – In the introduction of Javanese and Balinese script patterns, there are some research that have been conducted to determine the right method with different levels of accuracy. Some factors are very influential in determining the final result of research, such as the method used, research data, etc. This research uses two methods, namely the integral projection for feature extraction and neural network for classification. The purpose both of methods is to determine the accuracy of the result in recognizing the handwritten of the Sasak script This research is carried out to determine the effect of the number of nodes in a hidden layer. The highest accuracy if found in the use of hidden layers, including 21 nodes for the first hidden layer, and 14 nodes for the second hidden layer. The level of accuracy of this research is 41,38%.**

Keywords: **Sasak ancient scripts, integral projection, backpropagation**

I. PENDAHULUAN

Huruf sasak merupakan salah satu aksara tradisional nusantara yang digunakan oleh masyarakat suku Sasak di Lombok, Indonesia, yang digunakan untuk menulis bahasa sasak. Huruf sasak terdiri dari 18 karakter dasar. Beberapa peninggalan kuno yang menceritakan sejarah Lombok ditulis menggunakan simbol aksara sasak, namun dalam mengenali penggalan karakter aksara tidak mudah dan beberapa diantaranya memiliki pola yang hampir serupa sehingga sulit dibedakan oleh orang yang baru mengenal aksara. Saat ini orang sudah sangat jarang mempelajari akasara sasak. Penelitian ini penting dilakukan untuk mendukung pengenalan huruf sasak dan untuk membangun sistem pembelajaran akasara sasak. Sebelumnya terdapat beberapa penelitian menggunakan berbagai metode yang tingkat akurasi yang berbeda-beda. Sebagai contoh beberapa penelitian terkait dengan aksara nusantara yakni penelitian pertama menggunakan pelatihan JST dan *wavelet Haar 2 level* menghasilkan tingkat akurasi sebesar 97,857[1]. Penelitian berikutnya terdapat penelitian untuk mengenali karakter aksara jawa dengan *multi layer perceptron* dengan algoritma pembelajaran *backpropagation* dimana sistem dapat mengenali sampel pelatihan sebesar 100% dan sampel pengujian sebesar 38,1%[2].

Berbagai faktor sangat berpengaruh dalam penentuan hasil akhir penelitian seperti metode yang diterapkan, data penelitian dan sebagainya. Penelitian ini akan menggabungan 2 metode yang belum pernah dilakukan pada penelitian sebelumnya yaitu metode *integral projection* dan *neural network*.

Integral projection merupakan metode yang digunakan untuk mencari area dari objek. Algoritma dari metode *integral projection* sangat sederhana yakni dengan melihat piksel dan menjumlahkan piksel-piksel pada citra secara vertikal dan horizontal. Sehingga didapatkan nilai yang signifikan pada area tertentu sebagai hasil dari proses ekstraksi fitur sedangkan *neural network* mengadopsi sistem kerja otak sehingga membuat sistem bekerja dan berfikir layaknya manusia.

Berdasarkan kegunaan dari kedua metode tersebut diharapkan kedua metode ini dapat diterapkan untuk mengenali pola tulisan tangan huruf sasak sehingga kedepannya kedua metode ini dapat dikembangkan untuk sistem pembelajaran dalam mengenal huruf sasak.

II. TINJAUAN PUSTAKA

Penelitian terdahulu yang terkait dengan metode yang digunakan dalam penelitian ini yaitu telah dilakukan penelitian pengenalan wajah menggunakan *image processing* dengan sistem YCrCb dalam mengidentifikasi warna kulit. Dalam penelitian ini, *integral projection* digunakan untuk mencari area wajah dan *edge detection* dalam mencari tepi obyek[3]. Hasil dari penelitian ini dapat mendeteksi wajah dengan tingkat keberhasilan sebesar 98,5%, mendeteksi fitur pada wajah sebesar 88,75% serta akurasi mengenali wajah sebesar 81,67%.

Penelitian serupa dalam pengenalan wajah yakni identifikasi karakteristik ekspresi wajah yang telah berhasil dilakukan dalam mengenali *mood* atau emosi

seseorang dengan tingkat akurasi sebesar 98,09%[4]. Dalam peneliatian ini metode *integral projection* diaplikasikan untuk mendeteksi wajah, *Principal Component Analysis* (PCA) untuk mereduksi dimensi *Linier Discriminant Analysis* (LDA) untuk ekstraksi ciri ekspreksi wajah serta penerapan metode *fisherface* dengan pendekatan jaringan syaraf tiruan untuk pengenalan ekspresi wajah.

Untuk kasus pengenalan huruf, terdapat beberapa penelitian yang telah dilakukan sebelumnya. Penelitian pertama yakni mempelajari pola suatu karakter aksara jawa dengan menggunakan pelatihan JST[1]. Metode yang digunakan yaitu *backpropagation* dengan 1 lapisan tersembunyi. Sebelum citra diproses dalam JST, dilakukan dekomposisi citra dengan menggunakan transformasi *Wavelet Haar* 2 level untuk mengurangi beban komputasi. Hasil penelitian berupa sebuah perangkat lunak yang dapat melakukan pelatihan dan pengenalan terhadap citra aksara Jawa. Perangkat lunak dengan model arsitektur JST yang optimal dapat mengenali citra aksara Jawa dengan tingkat akurasi 97,857% untuk citra uji yang termasuk dalam data pelatihan, 45% untuk citra uji yang tidak termasuk dalam data pelatihan, dan 70,625% untuk citra uji yang mengandung *noise*.

Penelitian kedua yakni mengenali karakter Aksara Jawa *Nglegna* menggunakan *neural network*[2]. Penelitian ini menerapkan metode *multilayer perceptron* dengan algoritma *backpropagation* untuk 100 set sampel pelatihan dan 50 set sampel pengujian. hasil dari penelitian ini dapat mengenali sampel pelatihan sebesar 100% dan sampel pengujian dengan akurasi 38,1%.

Penelitian ketiga yang menggunakan metode serupa yakni penelitian pengenalan pola tulisan tangan aksara jawa menggunakan jaringan syaraf tiruan dengan algoritma *backpropagation*[5]. Dalam penelitian ini menggunakan citra berukuran 40x40 piksel, dengan laju pembelajaran 0,003 dan jumlah iterasi sebanyak 50000. Penelitian ini memperoleh tingkat akurasi sebesar 99,8% untuk pelatihan dan 95,81% untuk pengujian.

Penelitian berikutnya yaitu penelitian *Self Organizing Maps* (SOM) yang yang digunakan untuk Pengenalan Aksara Jawa[6]. Penelitian ini menunjukkan bahwa algoritma SOM dapat digunakan dalam proses pengenalan pola dengan persentase keberhasilan hasil pengujian pada aplikasi ini adalah 73,57 %, nilai ini didapat dari hasil pengujian 140 aksara Jawa.

Penelitian selanjutnya yakni pengenalan Huruf Bali menggunakan *Modified Direction Feature* dan Jaringan Syaraf Tiruan[7]. Hasil dari penelitian ini diperoleh tingkat akurasi untuk data penulis yang pernah menjadi data latih sebesar 80% dan 70% untuk data uji yang diperoleh dari penulis yang berbeda.

Berdasarkan tinjauan pustaka tersebut, dapat diketahui bahwa pengenalan pola karakter huruf sasak belum pernah pernah dilakukan sebelumnya begitu pula sistem pengenalan pola serupa yang menggunakan metode *integral projection* dan *neural network* belum pernah dilakukan sebelumnya. Dilihat dari beberapa penelitian yang menggunakan metode *integral projection* dan *neural network* memiliki tingkat akurasi yang beragam. Maka untuk mengetahui seberapa besar tingkat keberhasilan yang dihasilkan oleh kedua metode ini, dilakukan sebuah penelitian mengenali pengenalan pola huruf sasak menggunakan metode *integral projection* dan *neural network*.

A. Huruf Sasak

Aksara berfungsi sebagai media penulisan untuk membaca karya sastra kuno berbahasa daerah. Huruf sasak merupakan salah satu aksara tradisional nusantara yang digunakan oleh masyarakat suku Sasak di Lombok, Indonesia untuk menulis bahasa daerah yaitu bahasa sasak, yang terdiri dari 18 karakter dasar[8]. Semua karakter huruf sasak dapat dilihat pada Gambar 1

Gambar.1. Karakter Huruf sasak

B. Pengenalan Pola

Pengenalan pola merupakan sebuah proses dalam mengenali atau mengidentifikasi suatu entitas, kemudian mengelompokkannya kedalam kelas tertentu dimana isi dari kelas tersebut memiliki ciri-ciri yang serupa. Akurasi dari pengenalan pola berbeda-beda tergantung dari metode yang digunakan[9]. Dalam Proses pengenalan pola terdapat dua fase yakni:

1. Fase pelatihan

   Pada fase ini terjadi proses pembelajaran sistem dalam mengetahui dan menentukan ciri-ciri sebagai informasi untuk proses pengenalan dan klasifikasi.
2. Fase pengenalan

   Pada fase ini objek lain digunakan untuk diambil ciri-ciri nya lalu dikelompokkan pada kelas yang memiliki ciri-dominan yang serupa. Pada proses ini dapat diketahui seberapa besar tingkat keakurasian sistem dalam mengidentifikasi objek. Apakah sistem dapat mengelompokkan objek sesuai dengan kelas yang seharusnya.

C. Thinning

Thinning merupakan proses mengubah bentuk citra biner menjadi citra dengan tampilan batas objek sebesar satu piksel. Dalam proses *thinning* menghilangkan piksel lebih pada citra biner dengan mengubah 0 ke 1 atau sebaliknya [10].

a. Citra Asli b. Citra hasil *Thinning*

Gambar.2. *Image thinning*

Pada Gambar 2 terdapat 2 contoh gambar. Gambar 2a menggambarkan citra tulisan tangan huruf Sasak hasil *scan* dan Gambar 2b merupakan hasil proses *thinning* untuk Gambar 2a..

D. Principal Component Analysis (PCA)

Principal component analysis merupakan metode untuk mengubah bentuk dari data asli yang akan diteliti menjadi data yang lebih sederhana dengan cara mentransformasi data secara linier dan membentuk sistem sistem koordinat baru dengan varian maksimum. Sehingga dalam penerapan PCA didapatkan data baru yang lebih kecil tanpa menghilangkan karakteristik dan informasi dari data asli[11].

E. Integral projection

Integral projection merupakan metode dalam menentukan area atau letak dari suatu objek pada citra[4]. Algoritma *integral projection* yakni dengan menjumlahkan piksel tiap baris dan kolom pada citra, fungsi *vertical projection* dirumuskan pada Persamaan (1)

$$S(i) = \sum_{j=1}^{n\,(kolom)} x(i,j) \quad (1)$$

Sedangkan untuk persamaan fungsi *horizontal projection* ditunjukkan pada persamaan (2).

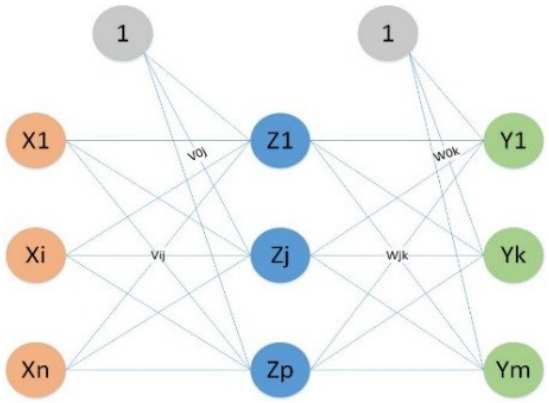
$$S(j) = \sum_{i=1}^{n\,(baris)} x(i,j) \quad (2)$$

dimana,
x(i,j) = piksel baris ke- i, kolom ke- j
S(i) =notasi untuk hasil penjumlahan pada baris ke-i
S(j) = notasi untuk hasil penjumlahan pada kolom ke-j

F. Backpropagation Neural Network

Backpropagation salah satu algoritma pembelajaran dalam jaringan syaraf tiruan dengan menggunakan banyak lapisan unit dalam menyesuaikan nilai bobot yang ada pada setiap lapisan tersembunyi[12]. Hal ini dilakukan untuk mencapai akurasi tertinggi yang sesuai antara hasil dari prediksi sistem dengan keluaran yang diinginkan [9].

Gambar 3 menggambarkan model dari arsitektur *neural network*. Variabel x diinisialisasikan sebagi *input*. Data *input* yang dimasukkan berasal dari hasil proses ekstraksi fitur, variabel z diinisialisasikan sebagai *node* pada *hidden layer* dan variabel y sebagai *layer* untuk *output*. Jaringan *backpropagation* dengan satu *layer* tersembunyi (unit Z (Z1 …Zp) tampak pada Gambar 3. Unit keluaran (unit Y (Y1...Ym) dan unit tersembunyi memiliki bias. Bobot bias pada unit keluaran Yk dinyatakan dengan W0k, Bobot bias pada unit tersembunyi Zj dinyatakan dengan Voj. Vij merupakan bobot garis dari unit Xi ke unit *layer* tersembunyi Zj. Wjk merupakan bobot garis dari Zj ke unit keluaran Yk.



Gambar.3. Arsitektur jaringan *backpropagation* dengan satu *hidden layer*

Berikut algoritma dari BPNN:

1. Memberikan inisialisasi bobot (ambil bobot awal dengan nilai random yang cukup kecil), tetapkan maksimum iterasi, target *error* dan laju pembelajaran.
2. Jika iterasi< maksimum iterasi dan MSE (*Mean Square Error*)> target *error* maka proses perhitungan akan terus berlanjut.
3. Tiap-tiap unit *input*(X_i, i=1,2…n) menerima sinyal X_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada diatasnya (lapisan tersembunyi).
4. Tiap unit pada suatu lapisan tersembunyi (Z_j, j=0,1,2…p) menjumlahkan sinyal-sinyal *input* terbobot menggunakan persamaan (3)

$$Z_{\_net\,j} = V_{0j} + \sum_{j=1}^{p} X_i V_{ij} \quad (3)$$

dimana,
Z_net =nilai *input* untuk *hidden layer*.
V = bobot awal dari unit *input* ke unit tersembunyi.
X = unit *input*.

Menggunakan fungsi aktivasi sigmoid untuk menghitung sinyal *output* dengan persamaan (4).

$$Z_j = f(Z_{\_net\,j}) \quad (4)$$

dimana,
Z = *output* sinyal dari *hidden layer*.

Kemudian mengirim sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*).

5. Tiap-tiap unit *output* Y_k (k=0,1,2,…,m) menjumlahkan sinyal-sinyal *input* terbobot menggunakan persamaan (5).

$$Y_{\_net\,k} = W_{0j} + \sum_{k=0}^{m} Z_j W_{jk} \quad (5)$$

Menggunakan fungsi aktivasi untuk menghitung sinyal *output* menggunakan persamaan (6).

$$Y_k = f(Y_{\_net\,k}) \quad (6)$$

Mengirim sinyal tersebut ke semua unit di lapisan atasnya (unit-unit *output*). Perhitungan ini dilakukan sesuai dengan jumlah unit tersembunyi.
dimana,
Y_net= sinyal *input* dari *hidden layer* ke unit *output.*
Y = *output* sistem.
W = bobot *hidden layer* untuk unit *output.*

6. Tiap-tiap unit *output* Y_k (k=0,1,2,…,m) menerima target pola yang berhubungan dengan pola *input* pelatihan, kemudian menghitung informasi *error*-nya menggunakan persamaan (7).

$$\delta_k = (t_k - Y_k) f'(Y_{\_net\,k}) \quad (7)$$

Kemudian menghitung suku perubahan bobot yang akandigunakan untuk merubah bobot W_{jk} dengan laju pembelajaran α menggunakan persamaan (8).

$$\Delta w_{jk} = \alpha \delta_k Z_j \quad (8)$$

dimana,
δ= informasi kesalahan yang akan digunakan dalam perubahan bobot *layer.*
α = laju perubahan (modifikasi) bobot di setiap iterasi.
Δw = suku perubahan bobot pada *hidden layer* dan unit *output*.
t = target *output* sistem.

7. Menghitung faktor δunit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi Z_j (j=0,1,2,…,p) menggunakan persamaan (9).

$$\delta_{net\,j} = \sum_{k=0}^{m} \delta_k w_{jk} \quad (9)$$

Mengalikan nilai ini dengan turunan dari fungsi aktivasinya untuk menghitung informasi *error* menggunakan persamaan (10).

$$\delta_j = \delta_{net\,j} f'(z_{net\,j}) \quad (10)$$

Kemudian menghitung suku perubahan bobot V_{ij} yang akan digunakan untuk merubah nilai V_{ij} (j=0,1,2...p; i=0,1,2..n) menggunakan persamaan (11)

$$\Delta v_{ij} = \alpha \delta_j x_i \quad (11)$$

dimana,
$\delta_{net\,j}$ = informasi kesalahan pada unit tersembunyi.
Δv = suku perubahan bobot pada unit *input* dan unit *hidden layer*.

8. Menghitung semua perubahan bobot (k=0,1,2,…,m) memperbaiki bias dan bobot (j=0,1,2…,p) menggunakan persamaan (12).

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk} \quad (12)$$

Tiap-tiap unit tersembunyi Zj (j=0,1,2,…,p) memperbaiki bias dan bobotnya (i=0,1,2,…,n) menggunakan persamaan (13).

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad (13)$$

9. Hitung MSE (*Mean Square Error*) menggunakan persamaan (14).

$$MSE = \sum (t - Y_k)^2 / n \quad (14)$$

dimana n merupakan jumlah dari unit *input*.

III. METODE PENELITIAN

A. Alat dan Bahan

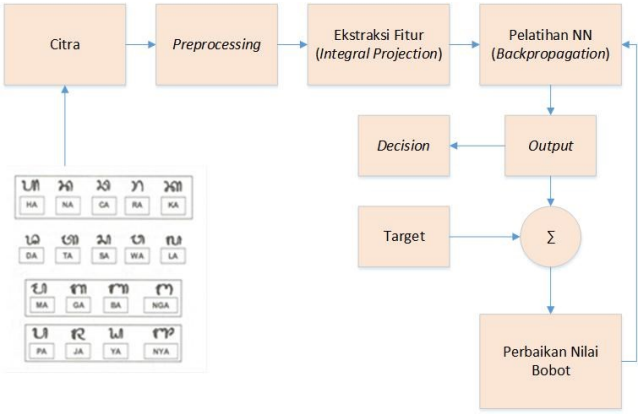
Penelitian yang dilakukan untuk sistem pengenalan pola huruf sasak menggunakan metode *integral projection* dan *neural network* membutuhkan alat dan bahan sebagai berikut:

Alat:
1. *Hardware* Laptop: menggunakan Intel® Celeron CPU N4000, 2.6 GHz, RAM 4 GB
2. *Operating system*: menggunakan *Operating System Windows*10*Ultimate* 64-bit.
3. *Software* Matlab: menggunakan *software* Matlab diperlukan untuk pembuatan program
4. *Software coreldraw x4*: untuk memotong citra hasil *scanner*.
5. *Microsoft office* 2010 untuk penyusunan laporan.
6. *Scanner* Canon LiDE 120. Resolusi 2400 dpi. Maksimum kertas *letter*.

Bahan:
1. Tulisan tangan karakter huruf sasak ditulis oleh 14 orang berbeda sebanyak 5 kali sehingga jumlah data yang diperoleh sebanyak 14x5x18 = 1260 tulisan tangan. Ditulis dengan kertas putih dengan alat tulis yang sama.
2. Citra huruf sasak dalam format jpg dengan ukuran maksimal 5 MB diperoleh dari hasil *scanning* tulisan tangan huruf sasak.

B. Rancangan Sistem

Pada tahap awal dilakukan tahap *preprocessing* pada citra aksara. Kemudian dilakukan ekstraksi fitur menggunakan metode *integral projection*. Pada proses *integral projection* didapatkan 128 keluaran. Hasil dari ekstraksi tersebut kemudian direduksi menggunakan metode PCA, sehingga dari 128 didapatkan 22 nilai. Hasil dari PCA tersebut kemudian dijadikan nilai *input* dalam tahap pelatihan pengenalan pola menggunakan *backpropagation neural network*



Gambar.4. Blok Diagram.

Gambar 4 merupakan diagram blok dari sistem yang akan dibuat. Pada tahap awal dilakukan tahap *preprocessing* pada citra aksara. Kemudian dilakukan ekstraksi fitur menggunakan metode *integral projection*. Pada proses *integral projection* didapatkan 128 keluaran. Hasil dari ekstraksi tersebut kemudian direduksi menggunakan metode PCA, sehingga dari 128 didapatkan 22 nilai. Hasil dari PCA tersebut kemudian dijadikan nilai *input* dalam tahap pelatihan pengenalan pola menggunakan *backpropagation neural network.*

Output dari proses pelatihan akan dijadikan acuan untuk kesimpulan dari sistem apakah sistem dapat mengenali dan mengklasifikasikan karakter huruf sasak atau tidak. *Output* akan disesuaikan dengan target yang diinginkan jika nilai kesalahan lebih besar dari target yang ditentukan maka akan dilakukan perbaikan kesalahan dan sistem akan mengulang kembali proses pelatihan. Terdapat 2 tahap dalam perancangan sistem ini yaitu:

1. Tahap Pelatihan

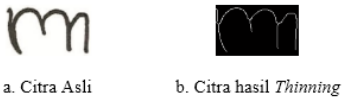
Dalam tahap pelatihan terdiri dari berbagai proses yakni sebagai berikut:

a. Menyiapkan citra digital huruf sasak untuk proses pelatihan.

Gambar.5. Citra *input* tahap pelatihan

Gambar 5 merupakan salah satu contoh citra yang akan digunakan dalam proses pelatihan.

b. Melakukan proses *preprocessing* dengan melakukan normalisasi agar ukuran citra sesuai dengan yang telah ditentukan. Proses ini diterapkan kepada seluruh citra dari 18 karakter huruf sasak. Data citra tersebut merupakan citra tipe RGB dengan format JPG. Proses awal dimulai dengan melakukan konversi citra RGB ke *Grayscale*. *Grayscaling* citra dilakukan dengan menjumlahkan tiap piksel pada 3 warna dasar yakni merah, hijau dan biru dan mengambil nilai rata-rata dari penjumlahan tersebut. Masing-masing nilai rata-rata pada setiap piksel memberikan warna sehingga warna menjadi *grayscale*. Hasil *grayscaling* kemudian dikonversi menjadi format *biner* dengan proses *binarization*. Setelah proses *binarization* kemudian dilakukan proses *thinning*. Proses *thinning* dilakukan untuk menyeragamkan ketebalan dari setiap data *training* yang akan digunakan pada penelitian ini.



a. Citra Asli b. Citra hasil *Thinning*

Gambar.6. Proses *thinning* pada citra latih

Pada Gambar 6 terdapat 2 gambar dimana pada Gambar 6a merupakan contoh citra latih dan Gambar 6b merupakan hasil proses *thinning* untuk citra pada Gambar 6a.

c. Melakukan proses ekstraksi fitur pada data latih. Ekstraksi fitur merupakan proses pengambilan ciri dari objek pada citra dimana ciri tersebut berisi informasi yang dapat membedakan objek tersebut dengan objek lainnya sehingga objek tersebut dapat dikenali. Tidak hanya pada citra tetapi ekstraksi fitur dapat dilakukan pada media lain seperti suara, video, teks dan sebagainya. Berbagai metode dapat digunakan dalam proses ekstraksi fitur salah satunya dengan *feature selection* pada penelitian *Twitter Sentiment Analysis*[13]. Namun pada penelitian ini menggunakan metode *integral projection*. Pada metode *integral projection*, piksel citra akan dijumlahkan secara *vertical* dan *horizontal* sehingga membentuk 2 buah vektor. Kedua vektor tersebut yang akan digunakan sebagai *input* dalam proses klasifikasi. Proses ini menghasilkan vektor dengan ukuran 128.
d. Mereduksi fitur yang didapat dari *Integral projection* menggunakan PCA sehingga dihasilkan 22 variabel yang akan digunakan dalam proses klasifikasi. Penggunaan PCA bertujuan untuk mereduksi jumlah variabel menjadi lebih sederhana tanpa menghilangkan informasi dari data asli, mengurangi waktu komputasi dan mengurangi pemakaian memori.
e. Melakukan pelatihan BPNN (*Backpropagation Neural Network*) untuk mendapatkan bobot *training sample* yang dapat digunakan untuk mengenali atau mengklasifikasikan huruf sasak.
f. Simpan nilai bobot *training sample* yang didapatkan dari proses pelatihan.

2. Tahap Pengenalan

Dalam tahap pengenalan terdiri dari berbagai proses yakni sebagai berikut:

a. Menyiapkan citra digital huruf sasak untuk proses pengenalan.

Gambar.7. Citra *input* tahap pengenalan

Gambar 7 merupakan contoh citra yang digunakan untuk proses identifikasi berdasarkan jaringan yang terbentuk dari proses pelatihan.

b. Melakukan proses *preprocessing* dengan melakukan normalisasi agar ukuran citra sesuai dengan yang telah ditentukan. Proses ini diterapkan kepada seluruh citra dari 18 karakter huruf sasak. Data citra tersebut merupakan citra tipe RGB dengan format JPG. Proses awal dimulai dengan melakukan konversi citra RGB ke *Grayscale*. *Grayscaling* citra dilakukan dengan menjumlahkan tiap piksel pada 3 warna dasar yakni merah, hijau dan biru dan mengambil nilai rata-rata dari penjumlahan tersebut. Masing-masing nilai rata-rata pada setiap piksel memberikan warna sehingga warna menjadi *grayscale*. Hasil *grayscaling* kemudian dikonversi menjadi format *biner* dengan proses *binarization*. Setelah proses *binarization* kemudian dilakukan proses *thinning*.
c. Melakukan ekstraksi fitur dengan menggunakan metode *integral projection* pada citra *testing sample*. Pada metode *integral projection*, piksel citra akan dijumlahkan secara *vertical* dan *horizontal* sehingga membentuk 2 buah vektor. Kedua vektor tersebut yang akan digunakan sebagai *input* dalam proses klasifikasi. Proses ini menghasilkan vektor dengan ukuran 128.
d. Mereduksi fitur yang didapat dari *Integral projection* menggunakan PCA sehingga dihasilkan 22 data set. 22 data tersebut digunakan dalam klasifikasi. Penggunaan PCA bertujuan untuk mereduksi jumlah variabel menjadi lebih sederhana tanpa menghilangkan informasi dari data asli, mengurangi waktu komputasi dan mengurangi pemakaian memori.
e. Melakukan klasifikasi berdasarkan hasil ekstraksi fitur citra *input* huruf sasak dengan metode BPNN.
f. Diperoleh hasil keluaran berupa klasifikasi huruf sasak.

Gambar.8. Hasil klasifikasi citra huruf sasak

Gambar 8 merupakan *interface* dari sistem dalam mengidentifikasi huruf sasak.

C. Teknik Pengujian

Teknik pengujian yang digunakan oleh sistem harus diuji dengan parameter-parameter statistik yang dikumpulkan[14]. Parameter-parameter tersebut meliputi tingkat akurasi. Tingkat akurasi merupakan nilai kesesuaian antara *input*-an dengan hasil klasifikasi. Untuk menghitung parameter akrurasi digunakan Persamaan

$$\text{Akurasi} = (TP+TN)/(TP+TN+FP+FN) \times 100\% \quad (15)$$

dimana TP (*True Positive*): banyaknya hasil klasifikasi benar untuk suatu kelas yang bernilai *positive*, TN (*True Negative*): banyaknya hasil klasifikasi benar untuk suatu kelas yang bernilai *negative* FP (*False Positive*) adalah banyaknya hasil klasifikasi salah untuk suatu kelas yang bernilai *negative*. FN (*False Negative*) adalah banyaknya hasil klasifikasi Salah untuk suatu kelas yang bernilai *positive*

K-Fold Cross Validation adalah pendekatan yang digunakan untuk memperkirakan performa model. Algoritma dari pengujian ini dengan memilah data sampel menjadi beberapa k bagian sama besar. Satu bagian akan menjadi datu uji dan model akan melakukan *fitting* terhadap k bagian lainnya. Proses ini dilakukan sebanyak k kali menggunakan *validation* yang berbeda.

IV. Hasil dan Pembahasan

A. Pengujian

Dalam pengujian ini akan dilakukan dengan 2 skenario berbeda yakni

1. Skenario pengujian pertama: pada proses pelatihan, data yang digunakan yaitu 18x5x10=900 data citra latih dan 18x5x4=360 data citra uji. Data dipilih secara random.
2. Skenario pengujian kedua: proses pengujian menggunakan *k-fold cross validation* (Tabel I) dengan menggunakan total data sampel yaitu 18x5x14=1260 data sampel dimana k yang digunakan sebanyak 5. Data dikelompokkan dengan jumlah 14 data per kelompok yang dipilih secara random. Setiap data memiliki pengaruh yang berbeda sehingga pengujian ini dilakukan untuk mengetahui setiap kemungkinan hasil yang diperoleh dari data yang ada.

TABEL I. Tabel skenario Pengujian K-Fold

Tahap 1	Tahap 2	Tahap 3	Tahap 4	Tahap 5
Fold1 testing	*Fold2 testing*	*Fold3 testing*	*Fold4 testing*	*Fold5 testing*
Fold2 training	*Fold1 training*	*Fold1 training*	*Fold1 training*	*Fold1 training*
Fold3 training	*Fold3 training*	*Fold2 training*	*Fold2 training*	*Fold2 training*
Fold4 training	*Fold4 training*	*Fold4 training*	*Fold3 training*	*Fold3 training*
Fold5 training	*Fold5 training*	*Fold5 training*	*Fold5 training*	*Fold4 training*

Beberapa variasi pengujian akan dilakukan untuk mengetahui performa dari teknik klasifikasi. Variasi tersebut antara lain:

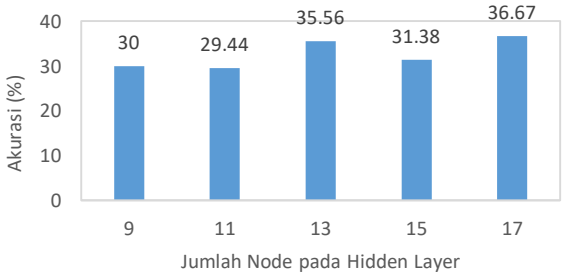
1. Pengaruh jumlah *node* pada sebuah *hidden layer* terhadap performa
2. Pengaruh jumlah *hidden layer* terhadap performa

Parameter yang digunakan untuk menentukan performa adalah akurasi yang dihitung menggunakan persamaan (15)

A.1 Pengaruh jumlah node pada sebuah hidden layer terhadap performa

Tujuan dari pengujian ini untuk mengetahui pengaruh jumlah *node* dari sebuah *hidden layer* terhadap performa dari proses klasifikasi. Pada pengujian ini bentuk dari jaringan *Neural Network* yang dibangun yakni terdiri dari 22 *node* untuk *input layer*, 1 *Hidden Layer*, dan 5 *node* untuk *Output layer*. Skenario dari pengujian ini yakni perubahan jumlah *node* pada sebuah *hidden layer*. Jumlah *node* yang digunakan yakni rentang dari jumlah *input* dan *output* yaitu dari 22 *node* hingga 5 *node*. Pemilihan jumlah *node* yang akan digunakan dengan mengambil nilai tengah dari rentan 22 dan 55 yakni 13. Kemudian menambah 2 *node* sebanyak 2 kali menjadi 15 dan 17 serta mengurangi 2 *node* sebanyak 2 kali menjadi 11 dan 9 sehingga pada percobaan ini terdapat 5 kali percobaan dengan variasi *node* yang berjumlah 9 *node*, 11 *node*, 13 *node*, 15 *node*, 17 *node*.

Dari proses tersebut didapatkan tingkat akurasi dari beberapa fitur yang ditunjukkan pada Gambar 4



Gambar.9. Tingkat Akurasi dalam berbagai Jumlah *Node*

Gambar 9 menggambarkan bahwa jumlah *node* yang digunakan pada sebuah *layer* berpengaruh terhadap tingkat akurasi. Banyaknya *node* yang digunakan tidak menjamin tingkat keakurasian semakin baik dan semakin banyak *node* yang digunakan maka waktu komputasi akan semakin lama. Percobaan ini menghasilkan tingkat akurasi tertinggi yang terdapat pada jumlah *node* 17 dengan tingkat akurasi sebesar 36,67%.

A.2 Pengaruh jumlah hidden layer terhadap performa

Untuk mengetahui tingkat akurasi terhadap *hidden layer* dilakukan perbandingan hasil akurasi dari berbagai jumlah *hidden layer* dimana jumlah *hidden layer* yang digunakan yaitu 2 *hidden layer* dan 3 *hidden layer* dengan beberapa variasi jumlah *node* yang berbeda.

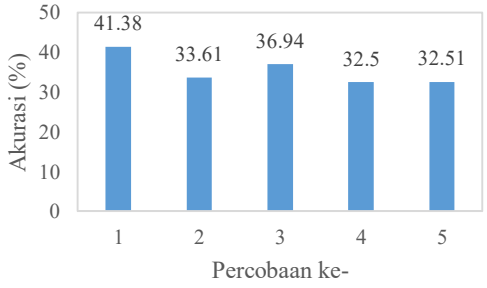
1. Pengujian pada 2 *hidden layer*

Pada pengujian ini menggunakan 2 *Hidden layer* dengan beberapa variasi *node* yang berbeda pada setiap *layer*-nya. Pemilihan jumlah *node* pada *hidden layer* mengambil nilai tertinggi dan nilai tengah antara rentang jumlah *input* dan *output*. Percobaan dilakukan sebanyak 5 kali. Dengan mengurangi 2 *node* untuk setiap percobaannya sehingga kombinasi untuk masing-masing *node* pada *hidden layer* dipaparkan pada Tabel II.

TABEL II. TABEL PENGUJIAN TERHADAP 2 HIDDEN LAYER

Percobaan ke-	HL ke 1	HL ke 2
1	21 *node*	14 *node*
2	19 *node*	12 *node*
3	17 *node*	10 *node*
4	15 *node*	8 *node*
5	13 *node*	6 *node*

Dari pengujian tersebut didapatkan hasil akurasi dari proses klasifikasi yang ditunjukan pada Gambar 10



Gambar.10. Tingkat Akurasi pada Pengujian 2 *Hidden Layer*

Gambar 10 menggambarkan bahwa akurasi tertinggi terletak pada percobaan pertama dengan tingkat akurasi sebesar 41,38%. Dibandingkan dengan menggunakan 1 *Hidden layer*, percobaan yang dilakukan dengan penggunaan 2 *Hidden layer* memiliki tingkat akurasi yang lebih tinggi.

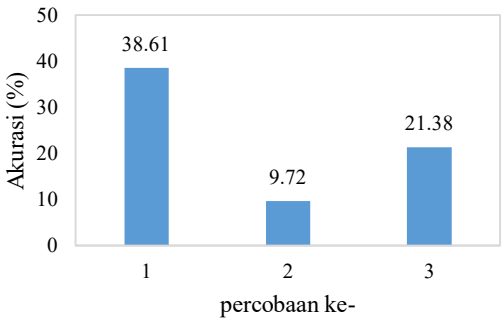
2. Pengujian pada 3 *hidden layer*

Pada pengujian ini menggunakan 3 *Hidden layer* dengan beberapa variasi *node* yang berbeda pada setiap *layer*-nya. Untuk masing-masing *node* pada *hidden layer* dipaparkan pada Tabbel III.

TABEL III. TABEL PENGUJIAN TERHADAP 3 HIDDEN LAYER

Percobaan ke	HL ke 1	HL ke 3	HL ke 3
1	20 *node*	16 *node*	12 *node*
2	18 *node*	14 *node*	10 *node*
3	16 *node*	12 *node*	8 *node*

Dari pengujian tersebut didapatkan hasil akurasi dari proses klasifikasi yang ditunjukan pada Gambar 11.



Gambar.11. Tingkat Akurasi pada Pengujian 3 *Hidden Layer*

Gambar 11 menggambarkan bahwa akurasi dari tertinggi terdapat pada percobaan pertama dengan tingkat akurasi sebesar 38,61%.

Hasil pengujian menggunakan 2 buah *hidden layer* dan 3 buah *hidden layer* tidak menjamin bahwa tingkat akurasi dalam proses pengklasifikasian bertambah. Jika dibandingkan kedua hasil percobaan tersebut, penggunaan 2 buah *hidden layer* memiliki tingkat akurasi yang lebih tinggi 0,83% dengan estimasi waktu pengujian yang lebih singkat. Dari beberapa percobaan yang telah dilakukan, tingkat akurasi yang dihasilkan tidak lebih dari 50%.

A.3 Pengaruh jumlah node pada sebuah hidden layer dengan k-fold

Pengujian kedua ini dilakukan menggunakan 1 *hidden layer* dalam berbagai variasi jumlah *node* yakni 17, 15, 13, 11 dan 9 dengan proses *K-fold cross validation* dimana k=5. Jika total data *input* yang digunakan berjumlah 1260 data maka pembagian data *input* untuk masing-masing K dipaparkan pada Tabel IV.

TABEL IV. PEMBAGIAN DATA INPUT DENGAN MENGGUNAKAN K-FOLD

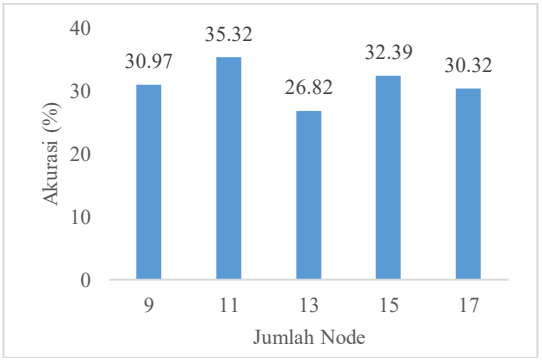
Kelas	Ha	Na	…..	…..	…..	Ya	Nya	Total
1	14	14	14	14	14	14	14	252
2	14	14	14	14	14	14	14	252
3	14	14	14	14	14	14	14	252
4	14	14	14	14	14	14	14	252
5	14	14	14	14	14	14	14	252
Rata-rata								

Berdasarkan pembagian data *input* pada Tabel 4 dapat diketahui pengaruh jumlah *node* terhadap performa sistem dikaji pada Tabel V.

TABEL V. TABEL PERFORMA MENGGUNAKAN BEBERAPA VARIASI NODE DENGAN K-FOLD

Percobaan	Akurasi (%)				
	17 *node*	15 *node*	13 *node*	11 *node*	9 *node*
Tahap 1	25,79	32,14	28,96	41,27	29,36
Tahap 2	32,94	34,52	26,99	32,14	37,7
Tahap 3	32,94	34,52	33,33	37,30	20,36
Tahap 4	28,17	29,36	27,38	29,76	37,3
Tahap 5	31,75	30,95	17,46	36,11	30,16
Rata-rata	**30,32**	**32,39**	**26,82**	**35,32**	**30,97**

Pengujian pada beberapa variasi jumlah *node* dalam sebuah *hidden layer* menunjukkan akurasi terbaik terletak pada *hidden layer* dengan jumlah *node* 11 yaitu 35,32 %. Berdasarkan kedua skenario pengujian yang telah dilakukan, hasil dari pengujian tersebut ditunjukkan dalam grafik pada Gambar 12



Gambar.12. Grafik performa pengujian satu *hidden layer* dengan *k-fold*

Sebaran data hasil klasifikasi yang ditunjukkan pada Gambar 12 menunjukkan bahwa perbandingan hasil akurasi pada percobaan sebelumnya dibandingkan dengan menggunakan pengujian *K-fold* hanya mengalami peningkatan pada pada jaringan dengan jumlah node 9, 11 dan 15.

A.4 Pengaruh jumlah hidden layer terhadap performa dengan k-fold

Tujuan pengujian ini yaitu mengetahui pengaruh dari jumlah *hidden layer* terhadap performa sistem dengan pengujian *K-Fold*. Pada pengujian ini menggunakan variasi *node* pada setiap *hidden layer*.

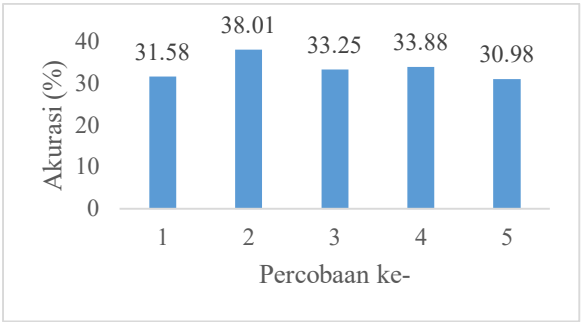
1. Pengujian terhadap 2 *Hidden Layer*

Pada pengujian ini menggunakan 2 *Hidden layer* dengan variasi *node* yang sama seperti percobaan sebelumnya pada Tabel 2. Untuk kolom *node* pada Tabel 6, angka pertama menjelaskan jumlah node untuk *hidden layer* yang bertama dan angka kedua menjelaskan jumlah *node* untuk *hidden layer* yang kedua. Dari pengujian ini didapatkan akurasi pengujian yang ditunjukkan pada Tabel VI.

TABEL VI. TABEL PERFORMA MENGGUNAKAN 2 HIDDEN LAYER DENGAN K-FOLD

Percobaan	Akurasi (%)				
	Node	*Node*	*Node*	*Node*	*Node*
	21 dan 14	19 dan 12	17 dan 10	15 dan 8	13 dan 6
Tahap 1	32,54	30,55	21,82	36,11	26,19
Tahap 2	26,98	38,09	38,10	32,54	18,25
Tahap 3	39,68	40,47	26,98	31,34	36,11
Tahap 4	34,92	39,68	41,26	34,92	37,69
Tahap 5	23,81	41,27	38,09	34,52	39,68
Rata-rata	**31,58**	**38,01**	**33,25**	**33,08**	**30,98**

Pada pengujian menggunakan 2 *Hidden layer* menunjukkan akurasi tertinggi sebesar 38,01% dimana pada *hidden layer* pertama terdapat 19 *node* dan untuk *hidden layer* kedua terdapat 12 *node*.



Gambar.13. Grafik performa pengujian pada 2 *hidden layer* dengan *k-fold*

Gambar 13 menggambarkan keseluruhan akurasi dari setiap percobaan pada Tabel 6.

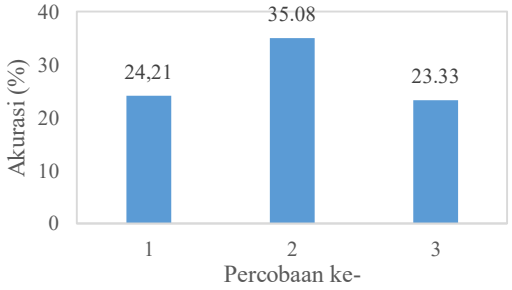
2. Pengujian terhadap 3 *Hidden Layer*

Pada pengujian ini menggunakan 3 *Hidden layer* dengan variasi *node* yang sama seperti percobaan sebelumnya. Untuk kolom *node* pada Tabel 7, angka pertama menjelaskan jumlah *node* untuk *hidden layer* yang bertama, angka kedua menjelaskan jumlah *node* untuk *hidden layer* yang kedua dan angka ketiga menjelaskan jumlah *node* pada *hidden layer* ketiga. Dari pengujian ini didapatkan akurasi pengujian yang ditunjukkan pada Tabel VII.

TABEL VII. TABEL PERFORMA MENGGUNAKAN 3 HIDDEN LAYER DENGAN K-FOLD

Percobaan	Akurasi (%)		
	Node	*Node*	*Node*
	20, 16, 12	18, 14, 10	16, 12, 8
Tahap 1	38,1	37,7	47,62
Tahap 2	13,89	42,06	7,54
Tahap 3	20,24	20,64	23,02
Tahap 4	44,84	45,63	18,65
Tahap 5	3,97	29,37	19,84
Rata-rata	**24,208**	**35,08**	**23,334**

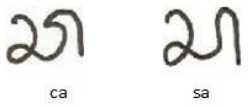
Pada pengujian menggunakan 3 *Hidden layer* menunjukkan akurasi terbaik terdapat pada percobaan kedua yakni 18 *node* pada *hidden layer* pertama, 14 *node* untuk *hidden layer* kedua dan 10 *node* untuk *hidden layer* ketiga dengan hasil akurasi sebesar 35,08 %. Berdasarkan kedua skenario pengujian yang telah dilakukan, hasil dari pengujian tersebut ditunjukkan dalam grafik pada Gambar 14.



Gambar.14. Grafik performa pengujian pada 3 *hidden layer* dengan *k-fold*

Sebaran data hasil klasifikasi yang ditunjukkan pada Gambar 14 menunjukkan bahwa hasil akurasi dengan menggunakan *K-fold* mengalami peningkatan percobaan kedua dan ketiga. Akurasi tertinggi terdapat pada percobaan kedua dengan selisih kenaikan tingkat akurasi sebesar 25, 36%.

Dari serangkaian pengujian yang dilakukan didapatkan hasil tingkat akurasi tertinggi terletak pada pengujian terhadap 2 *hidden layer*. Tingkat akurasi yang dihasilkan tidak melebihi 50%. Hal ini diakibatkan beberapa jenis karakter memiliki bentuk yang serupa sehingga dalam proses ekstraksi data memungkinkan nilai yang dihasilkan untuk karakter yang berbeda akan menghasilkan nilai ekstraksi yang hampir sama.

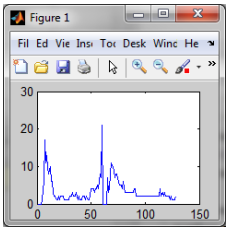


Gambar.15. Contoh karakter aksara yang memiliki kemiripan

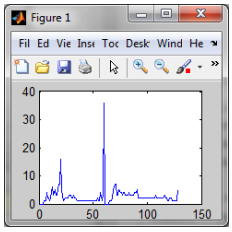
Gambar 15 merupakan contoh huruf sasak yang berbeda namun memiliki bentuk yang serupa. Faktor lainnya yakni metode dari *integral projection* dalam ekstraksi data hanya berdasarkan jumlah piksel yang didapat secara vertikal dan horizontal dengan mengabaikan bentuk dari karakter tersebut sehingga memungkinkan beberapa karakter yang bentuknya sangat berbeda namun jika piksel dari beberapa citra tersebut dijumlahkan dapat memberikan nilai yang serupa. Sehingga dari faktor-faktor tersebut menyebabkan proses pengklasifikasian tidak dapat sesuai dengan kelas yang semestinya.

a. Karakter "nga" b. Karakter "ra"

Gambar.16. Contoh karakter aksara yang memiliki perbedaan bentuk



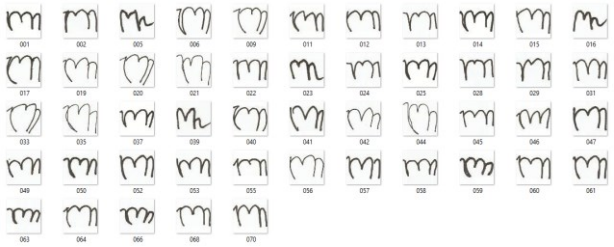
a. *Plot* diagram dari karakter "nga"



b. *Plot* diagram dari karakter "ra"

Gambar.17. Contoh plot karakter aksara untuk Gambar 16

Gambar 16 merupakan salah satu contoh karakter yang tampak berbeda namun ketika dilakukan proses *thinning* dan dilakukan ekstraksi fitur menggunakan *integral projection* pada kedua citra tersebut. Hasil grafik keduanya memiliki pola yang hampir serupa. Sehingga saat dilakukan proses klasifikasi, karakter "nga" diidentifikasi oleh sistem sebagai karakter "ra". Hasil dari grafik tersebut ditunjukkan pada Gambar 17. Faktor berikutnya yang dapat mempengaruhi hasil akurasi yakni data yang digunakan dalam penelitian.



Gambar.18. Contoh data latih untuk huruf sasak "ba"

Gambar 18 merupakan salah satu contoh citra huruf aksara "ba" yang digunakan pada proses *training*. Terlihat bahwa pola tulisan dari setiap huruf "ba" yang telah ditulis memiliki bentuk yang berbeda. Keragaman bentuk tersebut menyebabkan kemiripan nilai untuk setiap huruf "ba" terlampau kecil. Hal ini dapat mempengaruhi proses *training* sehingga tingkat akurasi yang dihasilkan pun kurang optimal

Dari berbagai jenis percobaan yang dilakukan, berbagai faktor terkait metode maupun data latih yang digunakan dalam penelitian dapat mempengaruhi tingkat akurasi yang dihasilkan. Selain itu jumlah *node* dan *hidden layer* yang semakin banyak dalam jaringan tidak menjamin bahwa tingkat akurasi akan mengalami peningkatan. Sebaliknya penggunaan *node* dan *hidden layer* yang semakin banyak akan memakan waktu komputasi yang cukup lama. Sehingga tidak disarankan untuk penambahan yang signifikan pada dua bagian tersebut.

V. PENUTUP

A. Kesimpulan

Berdasarkan penelitian pada bab sebelumnya, maka dapat disimpukan bahwa:

1. Jumlah *hidden layer* dan *node* dalam jaringan dapat mempengaruhi tingkat akurasi sistem dan waktu komputasi sistem.
2. Pengimplementasian PCA dalam sistem digunakan untuk mereduksi fitur hasil ekstrasi dengan *integral projection* sehingga didapat jumlah data set yang lebih sedikit.
3. Ekstraksi fitur menggunakan *integral projection* untuk beberapa jenis karakter dapat menghasilkan nilai yang hampir serupa sehingga hasil kelasifikasi kecil yaitu akurasi kurang dari 50%.
4. Tingkat akurasi tertinggi terletak pada pengujian pada percobaan menggunakan 2 *Hidden layer* dengan nilai akurasi sebesar 41,38%
5. Pengklasifikasian huruf sasak menggunakan metode *integral projection dan Neural network* dalam penelitian ini kurang mampu memberikan hasil akurasi lebih dari 50%.
6. Keanekaragaman data yang digunakan untuk proses *training* dapat mempengaruhi nilai akurasi yang dihasilkan.

B. Saran

Jika dilakukan penelitian lebih lanjut pada kasus ini dapat mempertimbangkan saran-saran dan perubahan sebagai berikut:

1. Pengimplementasian metode ektrasi fitur lainnya yang mampu memberikan informasi lebih baik dari *integral projection*
2. Pengembangan sistem lebih lanjut seperti pengenalan kata ataupun kalimat.
3. Modifikasi parameter dengan mengubah ukuran alat tulis maupun ukuran tulisan dalam citra.
4. Modifikasi model pembagian kelas berdasarkan beberapa parameter pendukung lainnya sehingga sistem mampu memberikan hasil klasifikasi yang lebih baik.
5. Penambahan kelas tidak dikenali untuk setiap citra *input* selain huruf sasak.
6. Modifikasi proses dapat dilakukan dengan menambahkan proses pengolahan citra lainnya untuk mendapatkan hasil yang lebih baik.

DAFTAR PUSTAKA

[1] D. E. J. Weisling, "Pembangunan Aplikasi Pengenalan Citra Aksara Jawa menggunakan metode Backpropagation dengan Wavelet sebagai Pemrosesan Awal Citra," 2011.

[2] M. C. Wibowo, "Pengenalan Citra Aksara Jawa menggunakan Metode Backpropagation dengan Wavelet sebagai Pemrosesan Awal Citra," 2015.

[3] M. Maziyah, "Implementasi VB 6.0 pada Face Detection berbasis Image Processing untuk Sistem Identifikasi," 2007.

[4] Zaenal Abidin, "Pengembangan Sistem Pengenalan Ekspresi Wajah menggunakan Jaringan Syaraf Tiruan Backpropagation (Studi Kasus pada Database MUG)," 2011.

[5] F. Asriani, "Pengenalan Pola Aksara Jawa Tulisan Tangan dengan Jaringan Syaraf Tiruan Perambatan Balik," 2009.

[6] A. Hidayat, "Selt Organizing Maps (SOM) metode untuk Pengenalan Aksara Jawa," 2016.
[7] I. G. R. Hermanto, "Analisis dan Implementasi Pengenalan Huruf Bali menggunakan Modified Direction Feature dan Jaringan Syaraf Tiruan," 2008.
[8] H. Yasri, *Cara Cepat Belajar Aksara Sasak*. Mataram: Pustaka Widiya, 2010.
[9] F. Pramesti, "Pengenalan Pola Angka dengan Wavelate Haar," 2007.
[10] V. Ariyaningrum, "Implementasi Algoritma Thinning pada Citra untuk Menilai Posisi Duduk Seseorang dari Segi Pencegahan Gangguan Penyakit Tulang Belakang," *Int. J. Comput. Appl. (0975 – 8887)*, vol. 91, 2014.
[11] D. E. Pratiwi, "Implementasi Pengenalan Wajah Menggunakan PCA (Principal Component Analysis)," *IJEIS*, vol. 3, pp. 175–184, 2013.
[12] E. Prasetyo, *Data Mining Mengolah Data menjadi Informasi menggunakan Matlab*. Yogyakarta: ANDI, 2014.
[13] M. A. Ulfa, "Twitter Sentiment Analysis using Na¨ıve Bayes Classifier with Mutual Information Feature Selection," *j-cosine*, vol. 2, 2018.
[14] R. Wati, "Penerapan Algoritma Genetika untuk Seleksi Fitur pada Analisis Sentimen Review Jasa," 2016. .