Aplikasi Pengenalan Aksara Jepang *Katakana* Menggunakan Jaringan Syaraf Tiruan *Learning Vector Quantization* (LVQ)

Angga Pradana Saputra dan Sukmawati Nur Endah

Departemen Ilmu Komputer/Informatika Universitas Diponegoro
[1]anggapradana167@gmail.com, [2]sukmawati020578@gmail.com

Abstract

Japanese katakana characters was used to wrote all non japanese words, like foreigner's name and uptake words from foreign language. For people who studying about japanese katakana, sometimes got a problem to remember it, so a compter application for japanese katakana's recognition has been developed. This application was developed using artificial neural network learning vector quantization. List of step of this application were pre-processing on characters image, extracting the feature of image using discrete wavelet transform, storing image's data vector, perform data training, and recognize an image. This application had a total 360 data of handwriting katakana's character which collected from 8 peoples. Then, all of these data will divided in to 8 subset on validity testing using k-fold cross validation to get an accuration value. Best value of character's recognition accuracy only 47,50% from parameter α = 0.08 and 0.09 and epoch = 10 as the parameter.

Keywords **:** *Japanese katakana characters, Artificial Neural Network, Learning Vector Quantization, K-Fold Cross Validation*

Abstrak

Aksara Jepang katakana digunakan pada saat menuliskan kata-kata selain bahasa Jepang, misalnya nama orang asing dan kata-kata serapan dari bahasa asing. Bagi orang-orang yang sedang belajar aksara Jepang katakana, terkadang mengalami kesulitan dalam menghafalkannya, sehingga dibuat sebuah aplikasi pengenalan aksara Jepang katakana. Aplikasi ini dikembangkan menggunakan jaringan syaraf tiruan *learning vector quantization*. Tahapan yang dimiliki oleh aplikasi ini yaitu melakukan pre-processing pada citra aksara, mengekstrak fitur citra menggunakan transformasi wavelet diskrit, menyimpan data vektor citra, melakukan pelatihan data, dan melakukan pengenalan terhadap citra yang dimasukkan. Data pelatihan yang digunakan dalam aplikasi ini sebanyak 360 data tulisan tangan aksara katakana yang berasal dari 8 orang. Selanjutnya data-data ini akan terbagi menjadi 8 subset pada pengujian validitas menggunakan k-*fold cross validation* untuk mendapatkan nilai akurasi. Nilai akurasi pengenalan aksara terbaik hanya sebesar 47,50% dengan parameter α = 0.08 dan 0.09 dan *epoch* = 10.

Kata kunci **:** Aksara Jepang Katakana, Jaringan Syaraf Tiruan, *Learning Vector Quantization*, K-*Fold Cross validation*

1. Pendahuluan

Bagi pelajar internasional yang menuntut ilmu di Jepang, walaupun telah menguasai bahasa internasional yakni bahasa Inggris, dalam bersosialisasi dengan masyarakat Jepang, tentulah harus mengetahui bahasa dan aksara yang digunakan di Jepang. Hal inilah yang terkadang menjadi suatu permasalahan bagi pelajar internasional, khususnya Indonesia. Dalam hal aksara, pelajar Indonesia telah terbiasa menggunakan aksara latin, namun masyarakat Jepang dalam kesehariannya menggunakan 4 macam aksara yakni *kanji*, *hiragana*, *katakana*, dan *romaji* (latin). *Kanji* dan *hiragana* digunakan untuk menuliskan kata-kata yang berasal dari bahasa Jepang asli, sedangkan *katakana* digunakan untuk menuliskan kata-kata asing yang diserap kedalam bahasa Jepang. Sebagai calon

mahasiswa asing yang akan belajar ke Jepang, penggunaan aksara *katakana* menjadi suatu hal yang penting guna menunjang kelancaran komunikasi sehari-hari. Kendala yang muncul dalam mempelajari aksara Jepang *katakana* yaitu proses dalam mengingat aksara katakana yang jumlahnya tidak sedikit. Pada saat mendapat sebuah tulisan yang berisi kata-kata dengan aksara *katakana*, kondisi lupa atau keliru dalam membedakan huruf-hurufnya seringkali muncul. Sehingga dibutuhkan suatu solusi yang dapat membantu dalam mengenali aksara-aksara yang ada dalam katakana. Salah satu solusi yang dapat dilakukan yaitu dengan membuat sebuah aplikasi komputer yang dapat mengenali *input* sebuah aksara katakana, kemudian memberikan *output* hasil pengenalan aksara yang di-*input*-kan.

Dalam ilmu komputer, terdapat sebuah cabang ilmu yang dikenal dengan *Artificial Intelligence* (AI). AI didefinisikan sebagai suatu studi tentang bagaimana membuat komputer dapat mengerjakan sesuatu yang dapat dikerjakan oleh manusia (Rich, 1991). Dalam AI sendiri, terdapat beberapa cabang ilmu, diantaranya yaitu : Sistem Pakar, Jaringan Syaraf Tiruan, Logika Fuzzy, dan Algoritma Genetik. Cabang ilmu AI yang sering digunakan dalam proses pengenalan suatu objek oleh komputer adalah Jaringan Syaraf Tiruan (JST). JST merupakan representasi buatan yang mencoba mensimulasikan proses pembelajaran pada otak manusia. Beberapa metode dalam JST yang dapat digunakan untuk mengenali suatu objek berupa aksara, diantaranya adalah *Backpropagation* dan *Learning Vector Quantization* (LVQ).

Pada penelitian yang dilakukan oleh Muhamad Fithri Qomari Azizi, metode LVQ mampu memberikan presentasi keberhasilan pengenalan yang lebih tinggi dibanding *Backpropagation*. Metode *Backpropagation* hanya memberikan hasil pengenalan sebesar 75,5%, sedangkan LVQ memberikan hasil 94%, dengan waktu *training* 102 detik untuk Backpropagation dan 0,3 detik untuk LVQ [1]. Kemudian, pada penelitian yang lain, tingkat keberhasilan LVQ pada pengenalan mencapai 87,093% [5]. Hal ini membuktikan bahwa metode LVQ cukup efektif untuk digunakan dalam proses pengenalan sebuah objek.

Berdasarkan uraian diatas, dapat dibuat sebuah aplikasi pengenalan aksara Jepang *Katakana* dengan menggunakan jaringan syaraf tiruan *Learning Vector Quantization* (LVQ) yang diharapkan dapat menjadi sebuah solusi dalam membantu mengenali dan mempelajari aksara Jepang *Katakana*.

2. Tinjauan Pustaka

Aksara Jepang Katakana

Aksara *katakana* digunakan untuk menuliskan kata-kata yang tidak berasal dari Jepang. Kata-kata yang dimaksud seperti kata serapan asing, nama-nama orang asing, serta kota-kota luar negeri dari Jepang. Aksara *katakana* terdiri atas 45 huruf dasar yang disebut dengan *Gojuon*. Selain 45 huruf dasar (*gojuon*), terdapat juga huruf-huruf hasil modifikasi dari aksara dasar yang terdiri dari 20 huruf *dakuon*, 5 huruf *sandakuon*, dan 36 huruf *yōon*. Namun pada penelitian ini hanya berfokus pada 45 aksara dasar saja. Berikut ini disajikan 45 huruf dasar katakana (*gojuon*) yang dapat dilihat pada Tabel 2.1 berikut: [4]

Tabel 2.1 Huruf Dasar *Katakana* (*Gojuon*)

ア	イ	ウ	エ	オ
a	i	u	e	o
カ	キ	ク	ケ	コ
ka	ki	ku	ke	ko
サ	シ	ス	セ	ソ
sa	shi	su	se	so
タ	チ	ツ	テ	ト
ta	chi	tsu	te	to
ナ	ニ	ヌ	ネ	ノ
na	ni	nu	ne	no
ハ	ヒ	フ	ヘ	ホ
ha	hi	fu	he	ho
マ	ミ	ム	メ	モ
ma	mi	mu	me	mo
ヤ	イ	ユ	エ	ヨ
ya	(i)	yu	(e)	yo
ラ	リ	ル	レ	ロ
ra	ri	ru	re	ro
ワ	イ	ウ	エ	オ
wa	(i)	(u)	(e)	(o)
n	ン			

Grayscaling

Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pixelnya. Nilai ini didapatkan dari hasil rata-rata nilai warna *Red*, *Green*, dan *Blue* pada citra berwarna RGB. Warna yang dimiliki citra

grayscale adalah warna dari hitam, keabuan, dan putih. Tingkatan keabuan disini dapat berbeda berdasarkan jumlah bit yang digunakan. Misalnya untuk citra *grayscale* 2 bit memiliki $2^2 = 4$ kombinasi warna keabuan dan untuk citra *grayscale* 8 bit memiliki 2^8=256 kombinasi warna keabuan [8].

Persamaan (1) adalah rumus mendapatkan citra *grayscale* [9].

$$Citra\ Grayscale = \frac{r+g+b}{3} \quad \text{.....(1)}$$

Thresholding

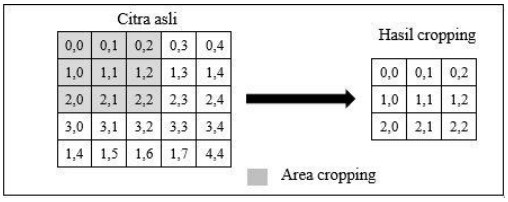
Operasi pengambangan (*thresholding*) mengelompokkan nilai derajat keabuan setiap *pixel* ke dalam 2 kelas, hitam dan putih [7]. Pada umumnya hasil thresholding adalah citra biner, namun pada penelitian ini, hasil thresholding berupa citra dengan piksel 0 dan 255. Rumus operasi thresholding yang digunakan pada penelitian ini dapat dilihat pada persamaan 2

Citra Thresholding $=$

$$\begin{cases} 0, f_{g(i,j)} \leq T \\ 255,\ lainnya \end{cases} \quad \text{.....(2)}$$

***Cropping* (Pemotongan Citra)**

Cropping adalah proses pemotongan citra pada koordinat tertentu pada area citra. Untuk memotong bagian dari citra digunakan dua koordinat, yaitu koordinat awal yang merupakan awal koordinat bagi citra hasil pemotongan dan koordinat akhir yang merupakan titik koordinat akhir dari citra hasil pemotongan. Sehingga akan membentuk bangun segi empat yang mana tiap – tiap *pixel* yang ada pada area koordinat tertentu akan disimpan dalam citra yang baru [6]. Proses pemotongan citra dapat dilihat pada gambar 2.1 berikut:



Gambar 2.1. Proses Pemotongan Citra

Scaling

Scalling(pengskalaan) adalah sebuah operasi geometri yang memberikan efek memperbesar atau memperkecil ukuran citra *input* sesuai dengan variabel pengskalaan citranya. Ukuran baru hasil pengskalaan didapat melalui perkalian antara ukuran citra *input* dengan variabel pengskalaan.

Proses pengskalaan dapat dilakukan dengan rumus :

$$P_o = S_p \times P_i \quad \text{.....(3)}$$

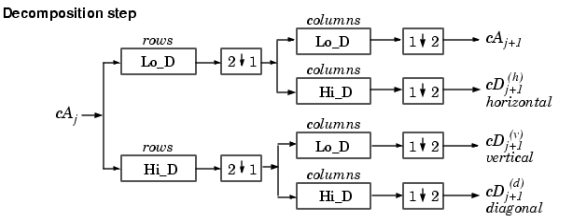
$$L_o = S_l \times L_i \quad \text{.....(4)}$$

Dimana (Pi, Li) adalah ukuran citra *input*, (Po, Lo) adalah ukuran citra *output*, dan (Sp, S_l) adalah variabel pengskalaan yang diinginkan. Jika variabel pengskalaan bernilai lebih besar dari 1 maka hasil pengskalaan akan memperbesar ukuran citra, sebaliknya apabila variabel pengskalaannya lebih kecil dari 1 maka hasilnya akan memperkecil ukuran citra [8].

Transformasi Wavelet Diskrit

Transformasi Wavelet Diskrit (*Discrete Wavelet Transform*) secara umum merupakan dekomposisi citra pada frekuensi *subband* citra tersebut dimana komponennya dihasilkan dengan cara penurunan level dekomposisi. Implementasi transformasi wavelet dapat dilakukan dengan cara melewatkan sinyal frekuensi tinggi atau *highpass filter* dan frekuensi rendah atau *lowpass filter*, dua komponen tersebut merupakan yang paling penting dalam transformasi wavelet. *Lowpass filter* disebut juga *scaling function* fungsi ini akan mengambil citra dengan gradiasi intensitas yang halus dan perbedaan intensitas yang tinggi akan dikurangi atau dibuang, sedangkan *highpass filter* disebut juga wavelet *function* fungsi ini mengambil citra dengan gradiasi intensitas yang tinggi dan perbedaan intensitas yang rendah akan dikurangi atau dibuang [2].

Transformasi wavelet diskrit dua dimensi dengan level dekomposisi satu dapat dilihat pada Gambar 2.2

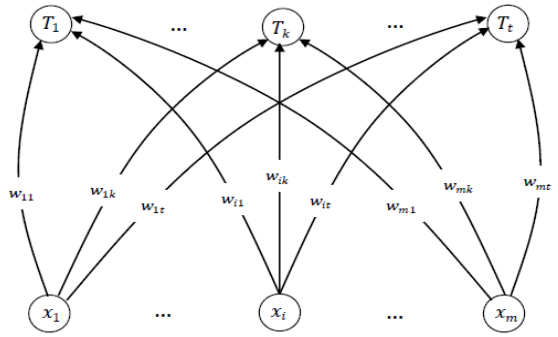


Gambar 2.2 Langkah Dekomposisi Transformasi Wavelet Diskrit

***Learning Vector Quantization* (LVQ)**

Learning Vector Quantization (LVQ) adalah metode jaringan syaraf tiruan yang melakukan

pembelajaran pada lapisan kompetitif terbimbing (*supervised*). LVQ merupakan algoritma yang cocok untuk klasifikasi pola yang masing-masing unit outputnya telah ditentukan target/kelasnya. Tujuan dari algoritma ini adalah untuk mendekati distribusi kelas vektor agar meminimalkan kesalahan dalam pengklasifikasian [3]. Arsitektur jaringan LVQ yang ditunjukkan pada Gambar 2.3.



Gambar 2.3 Arsitektur Jaringan LVQ

Algoritma LVQ pada tahap pelatihan adalah sebagai berikut [3]:

Langkah 0 : Inisialisasi bobot awal (w_k), *learning rate* (α), *max epoch*, dan *minimum error* (eps)

Langkah 1 : Ketika kondisi berhenti (*epoch* < max *epoch* || learning rate > eps), lakukan langkah 2 sampai 6

Langkah 2 : Untuk setiap input pelatihan vektor x, lakukan langkah 3 sampai 4

Langkah 3 : Temukan J sehingga ||x – w_k|| minimum

$$J = \sqrt{\sum_{i=1}^{n}(x_i - w_k)^2} \quad \text{.....(5)}$$

Langkah 4 : Perbaharui w sesuai dengan persamaan

Jika T = C_k, maka

$$w_{k\,(baru)} = w_{k\,(lama)} + \alpha[x - w_{k(lama)}] \quad \text{.....(6)}$$

Jika T ≠ C_k, maka

$$w_{k\,(baru)} = w_{k\,(lama)} - \alpha[x - w_{k\,(lama)}] \quad \text{.....(7)}$$

Langkah 5 : Kurangi learning rate (α). (Pengurangan nilai α dapat dilakukan dengan $\alpha = 0.1 * \alpha$)

Langkah 6 : Tes kondisi berhenti

Keterangan :

x : vektor pelatihan (*input*) (x_1, ..., x_i, ..., x_m)

T : kategori yang tepat atau kelas untuk vektor pelatihan

w_k : bobot vektor untuk unit *output* ke-k

C_k : kategori atau kelas yang ditampilkan oleh unit output ke-k

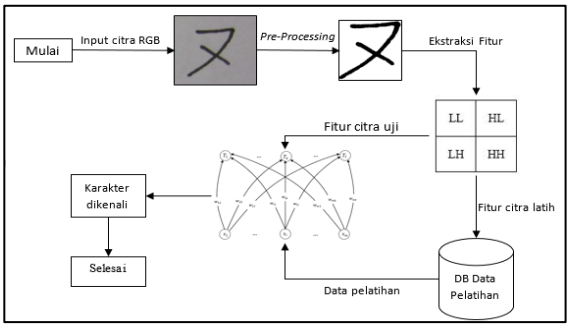
||x – w|| : jarak *Euclidean* antara vektor *input* dan bobot vektor untuk *layer output* ke-k.

3. Analisis dan Perancangan

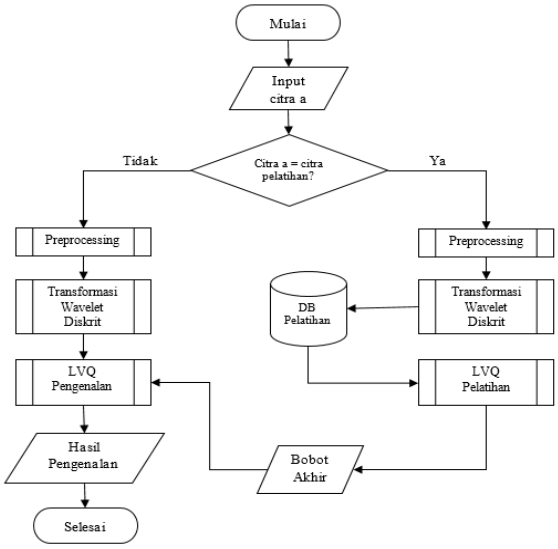
Deskripsi Umum Sistem

Aplikasi Pengenalan Aksara Jepang *Katakana* Menggunakan Jaringan Syaraf Tiruan *Learning Vector Quantization* adalah sebuah aplikasi berbasis desktop yang dikembangkan dengan menggunakan bahasa pemrograman MatLab dan basis data *Microsoft Access* yang berfungsi untuk mengenali masukan sebuah citra aksara *katakana* kemudian memberikan keluaran aksara yang cocok dengan citra yang dimasukkan sebelumnya. Aplikasi ini memanfaatkan jaringan syaraf tiruan LVQ dalam proses pelatihan dan pengenalannya. Aplikasi ini dapat diakses oleh semua pengguna. Sasaran pengguna dari aplikasi ini adalah kalangan umum khususnya pelajar yang ingin mempelajari dan menghafal aksara Jepang *katakana*.

Blok diagram proses aplikasi ini dapat dilihat pada Gambar 3.1 sedangkan *flowchart* untuk keseluruah aplikasi dapat dilihat pada Gambar 3.2



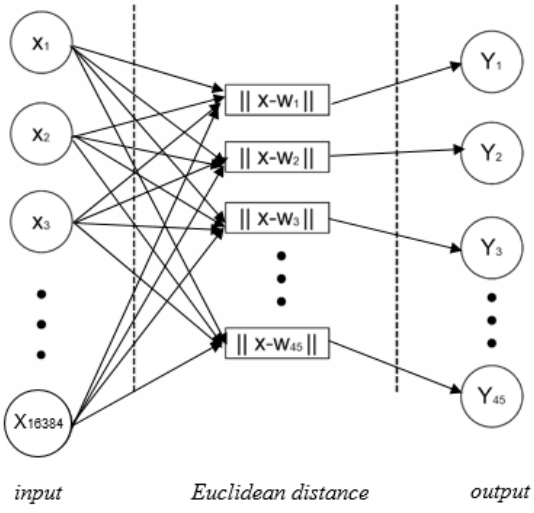
Gambar 3.1 Blok diagram proses aplikasi



Gambar 3.2 *Flowchart* keseluruhan aplikasi

Arsitektur Jaringan Syaraf Tiruan LVQ

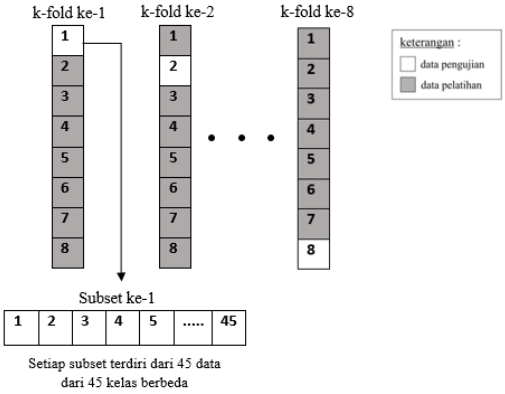
Hasil akhir transformasi wavelet diskrit berupa citra LL yang berukuran 128x128. Karena masih berbentuk matriks, citra akan diubah ke bentuk vektor agar dapat di proses pada LVQ. *Input* pada arsitektur LVQ merepresentasikan banyaknya nilai yang ada dalam vektor, yakni 128*128 = 16384. Sementara nilai Y(*output*) merupakan kelas/target yang terdiri dari 45 jenis aksara Jepang *katakana*. Arsitektur jaringan syaraf tiruan LVQ untuk aplikasi pengenalan aksara Jepang *katakana* dapat dilihat pada Gambar 3.3



Gambar 3.3 Arsitektur jaringan LVQ

Identifikasi Data Pelatihan dan Data Pengujian

Data-data hasil ekstraksi ciri yang disimpan didalam basis data, selanjutnya akan di proses menggunakan *K-Fold Cross Validation*. Tahapan ini dilakukan untuk membagi data hasil menjadi data pelatihan dan pengujian. Penelitian ini menggunakan dataset sejumlah 360 yang terdiri dari 45 kelas jenis aksara. Proses *K-Fold Cross Validation* ini menggunakan nilai k = 8, sehingga dataset akan dibagi menjadi 8 subset yang terdiri dari data dengan perbandingan jumlah kelas yang sama. Setiap subset akan terdiri 45 data, dimana 45 data tersebut mewakili dari setiap kelas aksara. Metode ini akan melakukan iterasi sebanyak k yaitu 8 kali, dimana pada iterasi pertama, subset ke-1 menjadi data pengujian sedangkan sisa subset lainnya akan menjadi data pelatihan. Pada iterasi kedua, subset ke-2 digunakan sebagai data pengujian dan subset lainnya sebagai data pelatihan. Penggambaran metode 8-*Fold Cross Validation* pada dataset lebih jelasnya ditunjukkan pada Gambar 3.4



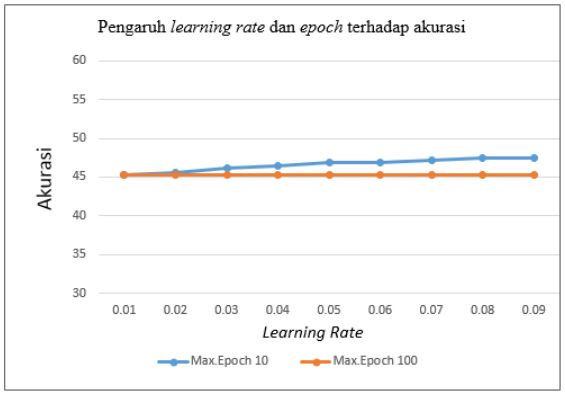
Gambar 3.4 8-*Fold Cross Validation* pengujian aplikasi

4. Implementasi

Pelatihan dan Pengujian Aplikasi

Proses pelatihan jaringan syaraf tiruan LVQ pada penelitian ini melibatkan kombinasi variabel LVQ yaitu inisialisasi bobot awal, *learning rate* (α) dengan parameter nilai antara 0.01 sampai dengan 0.09, *error minimum* (*eps*) dengan parameter nilai 0.001, 0.0001, dan 0.00001 serta maksimum *epoch* dengan parameter nilai 10*epoch* dan 100*epoch*. Proses pelatihan ini menghasilkan bobot akhir yang nantinya akan digunakan pada proses pengujian.

Pada pengujian menggunakan 8-*Fold Cross Validation*, akan digunakan seluruh parameter *learning rate* (α), *error minimum (eps)*, dan maks. *epoch*, sehingga kombinasi keseluruhannya menghasilkan 54 eksperimen. Hasil pengujian dapat dilihat pada grafik yang ditunjukkan pada Gambar 4.1

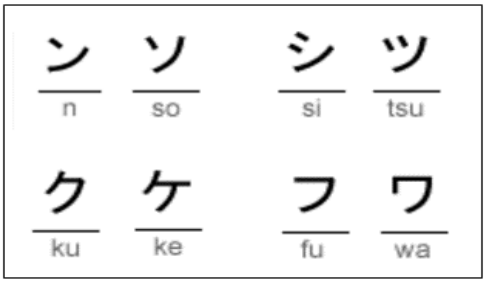


Gambar 4.1 Grafik hasil pengujian 8-*Fold Cross Validation*

Analisis Hasil Pengujian

Dari hasil pengujian dapat dilihat bahwa nilai learning rate yang besar dengan maksimum *epoch* yang kecil mampu memberikan kenaikan akurasi walaupun sangat kecil. Akurasi rata-rata tertinggi diperoleh dari learning rate 0.08 dan 0.09 dengan max.*epoch* 10*epoch* yaitu 47.50%. Menaikkan nilai *epoch epoch* memberikan hasil yang konstan atau tidak memberikan dampak pada akurasi. Rendahnya nilai akurasi dipengaruhi oleh beberapa faktor yaitu:

1. Terdapat beberapa aksara yang memiliki pola yang hampir sama, aksara-aksara tersebut yaitu n dan so, si dan tsu, ku dan ke, fu dan wa seperti yang ditunjukkan pada Gambar 4.2. Kemiripan pola aksara ini memiliki dampak yang cukup besar dalam proses pengenalan terutama pada saat perhitungan *euclidean distance* sehingga menyebabkan hasil pengenalan yang tertukar. Seperti pada Gambar 4.2, dapat dilihat adanya kemiripan pola aksara n dan so menyebabkan aksara n sering dikenali sebagai so dan begitu pula sebaliknya.



Gambar 4.2 Aksara katakana yang memiliki kemiripan pola

2. Beberapa citra aksara tulisan tangan yang diambil sebagai sampel untuk data latih tidak sesuai dengan pola penulisan aksara katakana yang seharusnya. Beberapa contoh sampel citra latih dengan pola penulisan yang kurang sesuai dengan aksara sebenarnya dapat dilihat pada Tabel 4.1

Tabel 4.1 Contoh sampel citra latih dengan pola penulisan tidak sesuai

Katakana	Sampel Citra Latih		Katakana	Sampel Citra Latih	
シ SI			ア A		
ツ TSU			ン N		

Pada Tabel 4.1 dapat dilihat beberapa sampel dengan penulisan yang kurang sesuai dengan aksara asli. Ketidaksesuaian pola penulisan memberikan pengaruh yang cukup besar pada proses perhitungan *euclidean distance*, sehingga pada proses pengenalan akan memberikan hasil yang berbeda. Selain itu, akurasi pengenalan dipengaruhi juga oleh kemiringan penulisan serta panjang pendeknya setiap garis yang ada pada citra aksara latih.

5. Kesimpulan

Dapat dibangun sebuah aplikasi pengenalan aksara Jepang *katakana* menggunakan jaringan syaraf tiruan LVQ dengan kesimpulan sebagai berikut :

1. Parameter yang digunakan saat pelatihan hanya memberikan perubahan akurasi yang sangat kecil. Menaikkan jumlah maksimum *epoch* tidak memberikan perubahan akurasi. Nilai learning rate yang lebih besar memberikan kenaikan akurasi walaupun sangat kecil.
2. Akurasi rata-rata yang diperoleh hanya sebesar 47,50%. Kecilnya nilai akurasi dapat dipengaruhi oleh beberapa faktor yaitu citra latih yang digunakan, dan banyaknya target kelas. Selain itu terdapat juga beberapa aksara yang memiliki pola yang hampir sama sehingga meningkatkan persentase *error*.

Referensi

[1]. Azizi, M. F. Q., 2013. *Perbandingan Antara Metode Backpropagation Dengan Metode Learning Vector Quantization (LVQ) Pada Pengenalan Citra Barcode*, Semarang: Universitas Negeri Semarang.

[2]. Banarjee, A. & Mistry, D., 2013. Discrete Wavelet Transform Using Matlab. *IJCET*, pp. 252-259.
[3]. Fausett, L. V., 1994. *Fundamentals Of Neural Networks.* New Jersey: Prentice-Hall.
[4]. japanese-lesson.com, 2014. [Online] Available at: www.japanese-lesson.com/characters/katakana/index.html [Diakses 31 Juli 2016].
[5]. Maulana, A., 2013. Aplikasi Pengenalan Karakter Pada Plat Nomor Kendaraan Bermotor Dengan Learning Vector Quantization. *SESINDO*, pp. 486-491.
[6]. Minartiningtyas, B. A., 2013. *http://informatika.web.id/operasi-cropping.htm.* [Online] [Diakses 15 January 2016].
[7]. Munir, R., 2004. *Pengolahan Citra Digital Dengan Pendekatan Algoritmik.* Bandung: Informatika.
[8]. Putra, D., 2010. *Pengolahan Citra Digital.* Yogyakarta: Penerbit Andi.
[9]. Santi, C., 2011. Mengubah Citra Berwarna Menjadi Gray-Scale dan Citra Biner. *XVI(1)*, pp. 14-19.