IJEIS, Vol.6, No.1, April 2016, pp. 13~24
ISSN: 2088-3714

Implementasi *Optical Character Recognition* Berbasis *Backpropagation* untuk *Text to Speech* Perangkat Android

Kristina Apriyanti*[1], Triyogatama Wahyu Widodo[2]
[1]Prodi Elektronika dan Instrumentasi, Jurusan Ilmu Komputer dan Elektronika, FMIPA UGM
[2]Jurusan Ilmu Komputer dan Elektronika, FMIPA UGM, Yogyakarta
e-mail: ***[1]chatterburst@yahoo.com**, [2]yogatama@ugm.ac.id

Abstrak

Prosedur penggunaan aplikasi text to speech pada perangkat mobile yang ada umumnya saat ini yakni pengguna aplikasi ini harus menginput manual kata yang akan diaktualisasikan dengan suara. Pada penelitian ini, dirancang sebuah sistem input kata pada aplikasi text to speech dengan memanfaatkan pengolahan citra digital. Pengguna cukup mengambil gambar (capture) kata yang akan disuarakan tersebut tanpa harus mengetik manual pada area teks input.

Metode yang digunakan dalam sistem ini meliputi akuisisi citra, pra pengolahan citra, segmentasi karakter, pengenalan karakter, dan integrasi dengan engine text to speech pada perangkat Android. Akuisisi citra dilakukan menggunakan kamera pada perangkat mobile untuk mengambil gambar kata yang akan diinputkan. Pengenalan karakter menggunakan jaringan saraf tiruan (JST) algoritma perambatan balik (back propagation). Sistem pengolahan citra yang berhasil dibuat kemudian dihubungkan dengan engine Google Text to Speech.

Sistem pengenalan karakter pada penelitian ini menggunakan model jaringan syaraf tiruan (JST) dengan akurasi 97,58%. Sistem ini mampu mengenali beberapa tipe font yakni Arial, Calibri, dan Verdana. Rerata akurasi pengenalan pada sampel uji yang digunakan di dalam penelitian ini sebesar 94,7% dengan kondisi jarak pengambilan gambar pada rentang jarak 3 – 8 cm dan posisi kamera tegak lurus menghadap kertas tulisan.

Kata kunci*— Android, OCR, Back Propagation, OpenCV, Text to Speech*

Abstract

Procedures using text to speech application on a mobile device generally at this time is user must manually enter the word to be actualized in speech. In this study, designed a words input system for text to speech application using digital image processing. This system makes users simply to do the words capturing that will be voiced without manually typing in the text area input.

The method used in this system includes image acquisition, image pre-processing, character segmentation, character recognition, and integration with text to speech engine on mobile devices. Image acquisition was performed using the camera on a mobile device to capture the word to be entered. Character recognition using back propagation algorithm. Image processing system successfully created and then integrated with Google Text to Speech engine.

Character recognition system in this study using a model of neural networks (ANN) with an accuracy of 97.58%. The system is able to recognize some types of font that is Arial, Calibri, and Verdana. The mean recognition accuracy on the test sample used in this study 94.7% with distance shooting conditions within the range 3-8 cm and the camera upright position facing the letter.

Keywords*— Android, OCR, Back Propagation, OpenCV, Text to Speech*

Received January 21th,2015; Revised April 13th, 2016; Accepted April 29th, 2016

1. PENDAHULUAN

Salah satu aplikasi pada perangkat *mobile* yang cukup berkembang yakni aplikasi *text to speech*, aplikasi yang digunakan untuk menghasilkan suara pembacaan kata yang diinputkan oleh pengguna. Aktualisasi suara pembacaan kata tersebut dilakukan sesuai dialek bahasa tertentu. Prosedur penggunaan aplikasi *text to speech* pada perangkat Android yang ada umumnya saat ini yakni pengguna aplikasi ini harus menginput manual (mengetik) kata yang akan disuarakan pada area input teks.

Di sisi lain, pengolahan citra digital khususnya sistem OCR (Optical Character Recognition) terus dikembangkan dalam pemrosesan citra tulisan dengan berbagai metode klasifikasi karakter. Salah satu metode klasifikasi yang terus dikembangkan dalam proses pengenalan karakter yakni jaringan saraf tiruan (JST) dengan pembelajaran perambatan balik (*backpropagation*). Metode ini memiliki pendekatan algoritma yang sederhana dengan akurasi baik, sehingga memungkinkan untuk diimplementasikan, *well-tested* di lapangan, dan mudah untuk dibentuk ke dalam algoritma yang efisien dan spesifik [1].

Merujuk pada fitur *smartphone* Android yang berbasis kamera serta berkapabilitas pemrosesan yang tinggi, memungkinkan bagi *smartphone* melaksanakan proses pengolahan citra digital yang identik dengan kebutuhan proses komputasi yang tinggi, termasuk juga proses OCR. Dari ketiga hal tersebut, maka pada penelitian ini dibuat sebuah sistem input kata pada aplikasi *text to speech* dengan memanfaatkan proses OCR, sehingga pengguna cukup mengambil gambar (*capture*) kata yang akan disuarakan tersebut tanpa harus mengetik manual pada area teks input.

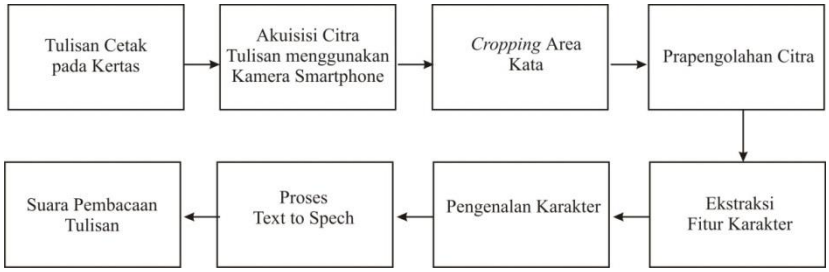
2. METODE PENELITIAN

2.1 Analisis Kebutuhan Sistem

Pada penelitian ini dibuat sebuah sistem yang mampu mengenali karakter dari citra tulisan cetak hasil *capture* oleh kamera pada perangkat Android dan hasil pengenalan tulisan tersebut kemudian dijadikan sebagai input text pada aplikasi *text to speech* perangkat Android. Masukan sistem berupa citra karakter tulisan cetak huruf dan angka hasil akuisisi menggunakan kamera smartphone Android. Karakter tulisan cetak yang menjadi masukan sistem meliputi huruf cetak A-Z, a-z, dan angka 0-9. Untuk pelatihan (*training*) menggunakan jaringan saraf tiruan (JST) dengan algoritma pembelajaran perambatan balik (*backpropagation*), citra masukan yang digunakan yaitu citra *training*. Citra *training* berupa citra karakter asli hasil akuisisi dari kamera. Keadaan citra *training* dibuat tegak, miring, dan berderau. Keluaran sistem berupa suara hasil pembacaan kata-kata yang berhasil diolah oleh sub sistem pengenalan karakter dari citra tulisan cetak tadi. Sistem diintegrasikan dengan *engine Google Text To Speech* sebagai unit pengkonversi karakter tulisan menjadi suara.

2.2 Rancangan Sistem

Secara keseluruhan diagram blok sistem yang dibuat pada penelitian ini dapat dilihat pada Gambar 1.



Gambar 1 Rancangan alur proses sistem secara keseluruhan

Proses pengenalan karakter tulisan membutuhkan beberapa tahapan pengolahan citra. Secara umum, proses pengenalan karakter pada citra teks meliputi akuisisi citra, pra pengolahan, ekstraksi ciri, dan pengenalan karakter. Kertas yang berisi kalimat teks tulisan cetak yang ditangkap citraannya oleh kamera pada perangkat Android tentunya mengandung *noise* dan bagian-bagian lain yang perlu dihilangkan untuk meringankan proses pengolahan, sehingga tahap pra pengolahan citra dibutuhkan untuk menangani hal tersebut [2]. Untuk dapat mengenali karakter pada citra tulisan, karakter tulisan pada citra harus diekstrak terlebih dahulu berdasarkan ciri masing-masing pola huruf tertentu, sehingga dibutuhkan metode ekstraksi ciri agar kemudian fitur yang dihasilkan dapat menjadi referensi pola pada proses pengenalan (klasifikasi) karakter. Algoritma klasifikasi yang akan diterapkan pada penelitian ini adalah algoritma perambatan balik (*backpropagation*). Proses *text to speech* akan menerima inputan kata-kata hasil proses pengenalan karakter berbasis pengolahan citra tadi. Proses pengolahan kata-kata menjadi suara dilakukan sepenuhnya menggunakan bantuan *engine Google Text to Speech* dengan *database* dialek bahasa Inggris.

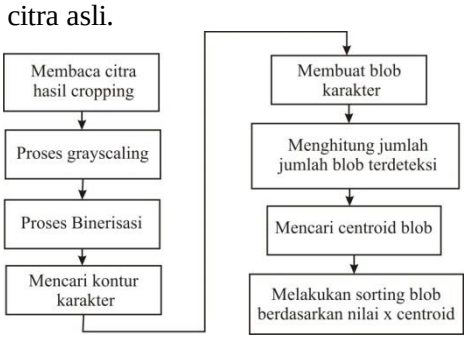
2.2.1 Akuisisi citra

Akuisisi citra tulisan cetak menggunakan kamera *smartphone* Android beresolusi 8 MP. Citra diambil dengan jarak yang cukup dekat dengan kertas yang berisi tulisan. Citra yang didapatkan kemudian dilakukan *cropping* pada area kata-kata yang hendak disuarakan. Citra hasil cropping ini kemudian diolah pada proses pengenalan karakter yang terdapat dalam sistem ini.

2.2.2 Pra pengolahan citra

Citra tulisan yang akan diekstraksi terlebih dahulu diolah untuk didapatkan citra yang lebih bersih dan ringan untuk diproses pada tahap selanjutnya. Alur tahapan proses pra pengolahan dapat dilihat pada Gambar 2. Citra hasil penangkapan kamera ditranformasikan ke dalam citra abu-abu (*grayscale*). Kemudian tahap binerisasi ini menggunakan metode *adaptive thresholding*. *Thresholding* dilakukan sebagai langkah pra pengolahan untuk menghilangkan derau (*noise*) latar belakang dari gambar sebelum ekstraksi karakter dan pengenalan karakter.

Segmentasi dilakukan untuk mendapatkan potongan karakter-karakter yang sudah terpisah satu sama lain dan siap untuk diproses untuk berdasarkan fiturnya. Segmentasi pada penelitian ini meliputi beberapa tahapan, yakni dengan mendeteksi karakter berdasarkan kontur yang didapatkan, menghitung jumlah blob (potongan) karakter yang berhasil terdeteksi, mencari centroid masing-masing blob, dan kemudian dilakukan sorting blob-blob yang terdeteksi berdasarkan nilai x-centroid yang telah didapatkan. Sorting blob berdasarkan nilai x-centroid perlu dilakukan untuk menjaga posisi urutan blob karakter yang terdeteksi agar tetap sesuai dengan urutan karakter pada citra asli.



Gambar 2 Alur tahapan proses pra pengolahan citra

2.2.3 Tahap ekstraksi fitur

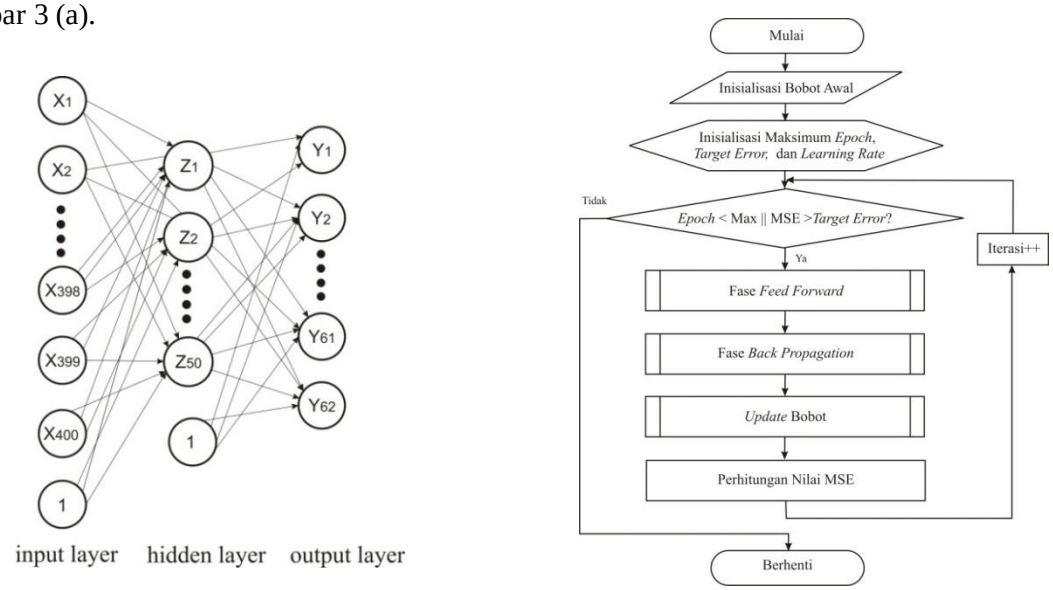
Ekstraksi fitur karakter digunakan untuk mengambil fitur pada citra karakter hasil pencarian blob karakter. Setiap citra karakter diekstrak fiturnya dari citra intensitas berdasarkan *contour* yang telah didapat.

Sebelum ekstraksi fitur, ukuran blob karakter dinormalisasi sesuai ukuran karakter dataset training karakter agar fitur yang diambil berukuran sama. Hasil dari normalisasi yaitu

citra karakter berukuran 20x20. Citra ini kemudian diubah ke deret biner untuk mendapatkan deret berukuran 1x400 sebagai masukan jaringan saraf tiruan (JST) perambatan balik.

2.2.4 Tahap klasifikasi

Tahap klasifikasi adalah proses pengenalan yang dalam penelitian ini menggunakan algoritma Jaringan Saraf Tiruan (JST) perambatan balik. Arsitektur JST perambatan balik pada penelitian ini menggunakan model *multilayer neural network* dan hanya meliputi 3 layer, yakni 1 *layer input*, 1 *layer hidden layer*, dan 1 *layer output*, dengan masing-masing layer memiliki sejumlah unit node. Rancangan arsitektur jaringan yang akan dibangun dapat dilihat pada Gambar 3 (a).



(a) (b)

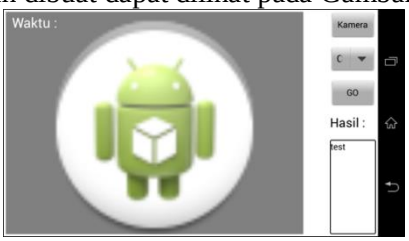
Gambar 3 Arsitektur dan algoritma JST *Backpropagation*

Tahap pembuatan model JST melewati 2 proses, yakni proses pelatihan (*training*) dan proses pengujian (*testing*). *Training* dilakukan untuk mendapatkan nilai bobot dari vektor masukan dan dilatih sesuai target yang telah ditentukan. Data *training* yang digunakan adalah citra berisi karakter a-z, A-Z, dan angka 0-9 yang disimpan dalam *database training*, di antaranya terdiri dari karakter dengan variasi jenis font Calibri, Verdana, dan Arial. Terdapat 3 fase dalam pelatihan menggunakan algoritma *backpropagation*, yaitu fase maju (*feed forward*), fase mundur (*back propagation*), dan fase *update bobot* [3]. Dalam fase *feed forward*, pola masukan dihitung maju dimulai dari lapisan input hingga lapisan output. Dalam fase *backpropagation*, tiap-tiap unit output menerima target pola yang berhubungan dengan pola input untuk dihitung nilai kesalahan. Kesalahan tersebut dipropagasikan mundur. Fase *update bobot* bertujuan untuk menurunkan kesalahan yang terjadi. Ketiga fase tersebut diulang secara terus menerus hingga kondisi penghentian dipenuhi. Adapun algoritma *backpropagation* yang diterapkan pada tahap pelatihan tersaji pada Gambar 3(b).

2.3 Implementasi

Implementasi rancangan sistem pengenalan karakter pada citra menggunakan Android berbasis Java. Program aplikasi Android dibuat menggunakan Eclipse. Sebelum rancangan sistem diimplementasikan, perlu dilakukan beberapa konfigurasi pada Eclipse dan *smartphone* Android yang digunakan. Android NDK plugin diinstal pada Eclipse untuk mendukung implementasi beberapa kode berbasis C++. Library OpenCV4Android diimport ke dalam *project* Android yang dibuat agar program dapat mengakses fungsi-fungsi pemrosesan citra yang terdapat di dalam *library* OpenCV. Pada *smartphone* yang digunakan pun perlu diinstal aplikasi OpenCV Manager. Di dalam aplikasi OpenCV Manager terdapat pustaka-pustaka

OpenCV yang diakses oleh aplikasi yang dibuat pada penelitian ini. Hasil implementasi rancangan antarmuka yang telah dibuat dapat dilihat pada Gambar 4.



Gambar 4 Antarmuka aplikasi

Pengguna dapat mendengarkan tulisan pada kertas tanpa perlu mengetiknya dengan menggunakan aplikasi ini. Pengguna cukup mengambil gambar (*capture*) tulisan yang dikehendaki dengan menekan *button* kamera. Setelah citra berhasil diambil, pengguna dapat melakukan *cropping* area kata-kata pada citra yang telah berhasil ditangkap. Pengguna kemudian dapat langsung mendengarkan penyuaraan frasa kata yang hasil *capture* tadi setelah menekan tombol *crop*. Pengguna dapat melihat hasil pemrosesan pengenalan karakter pada area Hasil.

Tahapan-tahapan proses yang dipaparkan pada perancangan sistem telah diimplementasikan menggunakan pemrograman Java untuk platform Android. Building project aplikasi pada penelitian ini mengimport library JNI (Java Native Interface) OpenCV untuk membantu mengimplementasikan beberapa kode yang berbasis C++. Project Android yang dibuat dalam penelitian ini memiliki 4 class utama yakni class MainActivity, class CropActiviy, class ImageConverter, dan class Inquisitor.

3. HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan tentang hasil dari implementasi sistem dan pembahasan mengenai hasil pengujian.

3.1 Hasil pra pengolahan citra dan deteksi karakter

Hasil tahapan pra pengolahan citra berupa *grayscaling* dan binerisasi dari salah satu citra tulisan cetak (Gambar 5(a)) yang ditangkap ditunjukkan pada Gambar 5(b), dan Gambar 5(c). Citra biner yang didapatkan kemudian masuk pada tahap segmentasi untuk mendeteksi karakter. Deteksi karakter bertujuan untuk mencari posisi karakter pada citra. Hasil deteksi karakter yakni berupa kontur objek yang akan dideteksi sebagai blob karakter (Gambar 5(d)).

(a) Citra asli (b) Citra *Grayscale* (c) Citra biner (d) Blob karakter

Gambar 5 Hasil pra pengolahan citra

3.2 Hasil ekstraksi ciri

Hasil deteksi karakter berupa blob karakter kemudian dinormalisasi sesuai ukuran fitur yang dijadikan masukan JST yakni berukuran 20x20. Hasil normalisasi blob karakter salah satu citra sampel dapat dilihat pada Tabel 1.

Tabel 1 Hasil normalisasi

Karakter	**B**	**u**	**y**	**3**	**G**	**e**	**t**	**1**
Blob Karakter								
Normalisasi Karakter								

3.3 Hasil pelatihan dan pengenalan karakter

Hasil pelatihan (*training*) sampel dengan akurasi terbaik diambil untuk digabungkan ke dalam aplikasi Android. Model JST di dalam aplikasi akan berperan sebagai *predictor* karakter pada setiap pola fitur yang berhasil diproses oleh aplikasi.

3.3.1 Hasil pelatihan sampel

Pelatihan (*training*) model JST algoritma perambatan balik (*backpropagation*) menggunakan bantuan *library machine learning* OpenCV [4]. Data *training* terdiri dari sampel karakter huruf A-Z, huruf a-z, dan angka 0-9, meliputi *font* Arial, Verdana, dan Calibri yang diambil menggunakan kamera *smartphone* dengan kondisi normal, miring, blur, dan berderau. Jumlah *database* sampel yang digunakan yakni sebanyak 1240 sampel karakter. Keakuratan pengenalan hasil *training* diuji dengan *data testing.*

Arsitektur jaringan yang dibentuk yakni terdiri dari 1 *input layer*, 1 *hidden layer*, dan 1 *layer output*. Layer input terdiri dari 400 node yang merupakan banyaknya fitur yang diambil dari tiap karakter, sedangkan layer output terdiri dari 62 node yang mewakili jumlah pola karakter yang dikenali (A-Z, a-z, dan 0-9). Penentuan jumlah hidden node pada model JST dilakukan dengan proses *trial and error* [5]. Beberapa percobaan dapat dilihat pada Tabel 2.

Tabel 2 Tabel hasil uji coba jumlah *node hidden*

No.	Jumlah *Hidden Node*	Akurasi Model JST (%)
1.	20	59.41
2.	30	85.75
3.	50	97.58
4.	70	97.31
5.	100	92.67
6.	150	97.58
7.	200	96.68
8.	240	97.51
9.	280	96,68
10.	300	92,96

Dari beberapa nilai yang dicoba kemudian dipilih jumlah node yang memiliki akurasi tertinggi. Berdasarkan Tabel 2, terdapat 2 jumlah node dengan nilai akurasi model yang paling tinggi (97,58%), yakni 50 node dan 150 node. Pada penelitian ini, jumlah node yang dipilih adalah 50, dengan pertimbangannya adalah dipilih model JST yang lebih sederhana. Banyaknya jumlah node akan mempengaruhi ukuran file model JST yang dibentuk dan mempengaruhi lamanya proses komputasi. Karena model JST ini pada akhirnya akan dipanggil (*load*) di dalam aplikasi, maka dipilihlah ukuran model yang lebih sederhana agar proses klasifikasi di aplikasi tidak memakan waktu yang lama.

3.3.2 Hasil pengenalan karakter

Proses pengenalan karakter pada aplikasi Android yang dibuat juga mengimport *library* OpenCV *machine learning.* Model JST hasil training dengan akurasi paling baik kemudian diimport ke dalam aplikasi ke dalam folder *asset*. Model JST yang disimpan dalam format file .xml akan dipanggil pada saat proses pengenalan karakter mulai dijalankan.

3.4 Hasil pemanggilan engine Text to Speech

Di dalam aplikasi yang dibuat, *engine text to speech* telah berhasil diimplementasikan untuk mensintesis kata-kata hasil pengenalan karakter yang ditunjukkan pada area teks hasil. Instance *text to speech* dengan dialek bahasa Inggris UK telah berhasil dipanggil untuk mendapatkan string kata-kata yang terdapat pada area teks hasil. Pembacaan yang dihasilkan oleh *engine* ini sudah sesuai dengan karakter yang tertera pada area teks hasil. Integrasi ini memberikan hasil bahwa aplikasi berbasis *text to speech* ini mampu menyuarakan kata-kata yang dikehendaki tanpa perlu mengetik manual pada area teks input.

3.5 Pengujian variasi jarak kamera saat pengambilan gambar

Kondisi pengambilan gambar yang dilakukan oleh kamera *smartphone* sangat berpengaruh besar terhadap keberhasilan pendeteksian karakter dan begitu pula dengan hasil pengenalan terhadap karakter-karakter yang berhasil dideteksi tersebut. Jarak kamera dengan tulisan menentukan besarnya ukuran citra karakter yang akan diproses oleh sistem. Untuk mengetahui rentang jarak bagi sistem untuk bisa mendapatkan hasil pengenalan karakter dengan baik, maka dilakukan pengujian terhadap beberapa jarak pengambilan gambar secara tegak lurus terhadap tulisan. Pengujian dilakukan dengan ukuran font yang bervariasi yakni 12pt, 14pt, dan 18pt.

Tabel 3 Hasil pengujian variasi jarak pengambilan gambar

Jarak (cm)	Ukuran Font	Jumlah Karakter Pengenalan			Total Karakter Dikenali Benar	Rerata Akurasi Pengenalan (%)
		Sampel 1	Sampel 2	Sampel 3		
3	12	11	15	11	37	92.5
	14	0	0	0	0	0
	18	0	0	0	0	0
4	12	13	13	12	38	97.5
	14	13	13	13	39	97.5
	18	11	0	12	23	57.5
5	12	8	11	12	31	77.5
	14	13	15	12	40	100
	18	13	14	12	39	97.5
6	12	4	8	8	20	50
	14	9	15	9	33	82.5
	18	13	15	12	40	100
7	12	0	0	4	4	10
	14	5	12	0	17	42.5
	18	13	14	12	39	97,5
8	12	0	0	8	8	20
	14	0	0	0	0	0
	18	13	12	11	36	90
9	12	0	0	0	0	0
	14	0	0	0	0	0
	18	10	0	6	16	40

Pengujian memberikan beberapa hasil rentang jarak yang berbeda terhadap masing-masing ukuran karakter uji yang masih bisa dikenali dengan baik. Rentang jarak agar tulisan bisa dikenali dengan baik untuk font berukuran 12pt yakni pada rentang jarak capture 3 - 4 cm, untuk font berukuran 14pt yakni pada rentang jarak capture 4 – 6 cm, dan untuk font berukuran 18pt yakni pada jarak capture 5 – 8 cm. Dari ketiga rentang jarak *capture* dengan hasil pengenalan yang baik untuk masing-masing ukuran font yang diuji tersebut, maka dapat diambil rerata akurasi pengenalan yang dicapai pada pengujian ketiga sampel yang digunakan yakni

$$\text{akurasi (\%)} = \frac{jumlah\ semua\ karakterdikenali}{jumlah\ semua\ karakter\ uji} x100$$

$$= \frac{37+38+39+40+33+39+40+39+36}{360} = \mathbf{94{,}7\ \%}$$

3.6 Pengujian variasi jenis font

Perbedaan karakteristik bentuk antar tipe font umumnya terletak pada lebar setiap huruf, lebar lengkungan, tipe ornamen huruf (huruf dengan ujung-ujung berekor atau tidak), jarak antar huruf, ketebalan setiap huruf, dan bentuk dasar beberapa hurufnya (seperti misal huruf ‘a’ sudah berbeda bentuk dasar dengan huruf ‘a’). Karena adanya beberapa perbedaan tersebut, maka dilakukan pengujian terhadap beberapa jenis font cetak tegak standar yang terdapat dalam *database font* Microsoft Office Word. Pengujian dilakukan dengan percobaan mengenali

beberapa jenis font yang menjadi data training maupun beberapa jenis font di luar *data training*. Pengujian variasi jenis font dilakukan untuk menguji dan membandingkan seberapa jauh sistem masih mampu mengenali beberapa tipe bentuk font lain yang tidak masuk dalam *data training* karakter. Untuk dapat membandingkan akurasi pengenalan antar tipe font, pengujian dilakukan dengan ukuran huruf dan frasa kata yang sama. Tabel hasil pengenalan karakter pada beberapa tipe font pengujian dapat dilihat pada Tabel 4.

Tabel 4 Hasil pengenalan karakter dengan berbagai tipe font

Jenis Font	Karakter Dikenali			Total Karakter Dikenali Benar	Rerata Akurasi Pengenalan (%)
	Huruf Besar	Huruf Kecil	Angka		
Arial	MY HEART BREAKS	a hUndred years	02746351498	36	97.3
Calibri	MY HEART BREAKS	a hundred years	02746351498	37	100
Verdana	MY HEART BREAKS	a hundred years	02746351498	37	100
Courier	HY xE L1 BxLAKS	a hundLed yearS	02T46351498	27	72.97
Letter Gothic Std	MY HEART BREAKS	a hUndned 7eanS	0274635j498	30	81.08
Tahoma	HY HEART BkEAKS	a hundred yeaFs	02746351498	34	91.9
Times New Roman	MT XLART 8kEALS	a hunkd 7cZ	o274635I498	24	64.86
Consolas	MY HEART BREAKS	a hundned yeans	e2746351498	34	91.89
Ms Gothic	MY HEART BREAKS	a hundned yeans	02746351498	35	94.59
Lucida Console	MY HEART BREAKS	a hundned yeanS	02746351498	34	91.89
Comic Sans MS	My HEART BqEAKS	o hundned y Kns	02746351498	29	78.38
Cambria	MY HEART BkEAKS	a hundLed 7earS	027463S149B	31	83.78
Century	ZALT4L Z	a hundred 7ezrS	02746351498	21	56.75
Corbel	MY HEART BREAKS	a hundFed yeaFS	o27i635 A98	30	81.08

Dari Tabel 4 tampak bahwa untuk ketiga font yang dijadikan *data training* dapat dikenali dengan baik oleh sistem ini. Tipe font lain di luar data *data training* beberapa dapat dikenali cukup baik oleh sistem ini, terlebih untuk karakter angka yang hampir semua font di luar *database* bisa mengenalinya dengan baik. Namun, ada beberapa karakter huruf pada font di luar *database* yang cukup tidak dikenali oleh sistem ini, tampak pada prosentase hasil pengenalan yang dapat dikatakan rendah. Karakter yang salah diklasifikasi ditandai dengan karakter berwarna merah. Jika dilihat lebih spesifik, ada beberapa kategori permasalahan yang mempengaruhi akurasi pengenalan karakter pada tipe-tipe font tersebut, antara lain bentuk dasar huruf kapital dan huruf kecil yang sama, kemiripan bentuk karakter citra uji dengan bentuk huruf berlabel lain pada karakter *data training*, dan segmentasi yang belum sempurna.

3.7 Pengujian pengaruh warna tulisan dan warna kertas

Tulisan yang memiliki warna huruf kontras dengan warna kertas biasanya lebih nyaman dilihat dan dibaca oleh mata kita. Warna ternyata juga memiliki pengaruh terhadap

aktivitas yang bersifat *optical*. Maka dari itu, pengujian pada sistem dengan objek bervariasi warna dilakukan untuk mengetahui pengaruh warna tulisan dan warna kertas terhadap akurasi pengenalan pada sistem. Kertas yang digunakan adalah HVS berwarna merah, putih, kuning, dan biru. Variasi warna karakter yang diujikan adalah hitam, merah, biru, dan orange. Total karakter sampel yang dikenali pada pengujian ini sejumlah 43 karakter. Hasil pengujian pengaruh warna terhadap sistem dapat dilihat pada Tabel 5.

Tabel 5 Hasil pengujian variasi warna font dan warna kertas

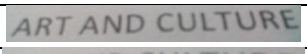
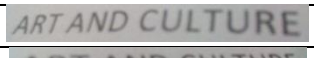
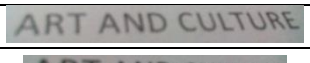
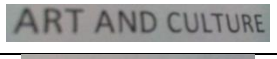
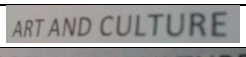
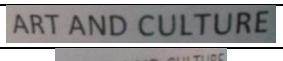
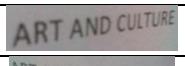
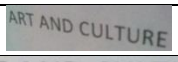
Warna Kertas	Warna Karakter	Jumlah Karakter Dikenali			Total Karakter Dikenali Benar	Rerata Akurasi Pengenalan (%)
		Sampel 1	Sampel 2	Sampel 3		
putih	hitam	15	14	14	43	100
	biru	15	14	14	43	100
	merah	15	14	14	43	100
merah	hitam	15	14	14	43	100
	biru	15	14	14	43	100
	merah	15	14	14	43	100
biru	hitam	15	14	14	43	100
	biru	15	14	14	43	100
	merah	15	14	14	43	100
kuning	hitam	15	14	14	43	100
	biru	15	14	14	43	100
	merah	15	14	14	43	100

Berdasarkan hasil pengujian pada Tabel 5 dapat dilihat bahwa sistem yang dibuat masih bisa menghasilkan akurasi pengenalan dengan baik pada warna font selain hitam dan warna kertas selain putih. Hal ini menunjukkan bahwa sistem masih bisa mensegmentasi objek karakter dengan beberapa variasi kombinasi warna kertas dengan warna font yang diujikan.

3.8 Pengujian pengaruh arah posisi pengambilan gambar

Arah posisi kamera saat mengambil gambar tentunya mempengaruhi bentuk suatu objek di dalam citra. Pada pengujian-pengujian lainnya dalam penelitian ini, kamera diposisikan tegak lurus pada tulisan untuk mengetahui pengaruh parameter-parameter pengujian terhadap akurasi pengenalan. Pada pengujian kali ini, parameter pengujian yang akan divariasikan adalah posisi kamera terhadap kertas tulisan saat pengambilan gambar. Tulisan akan ditangkap dari beberapa arah, antara lain dari arah bawah, atas, samping kiri, samping kanan, dan tegak lurus. Tabel hasil pengujian salah satu sampel dapat dilihat pada Tabel 6.

Tabel 6 Hasil pengujian variasi arah posisi *capture*

Kategori Posisi	Sampel Uji 1	Jumlah Karakter Pengenalan	Akurasi (%)	Waktu (ms)
Bawah		9	69,23	1086
Samping kanan bawah		9	69,23	1086
Samping kiri bawah		8	61,54	1099
Samping kiri		12	92,3	1110
Samping kanan		10	76,92	1073
Atas		12	92,3	1162
Samping kiri atas		10	76,92	1012
Samping kanan atas		9	69,23	1070
Tegak lurus		13	100	1124

Berdasarkan Tabel 6, dapat dilihat bahwa akurasi pengenalan sangat bergantung pada arah posisi pengambilan gambar. Akurasi paling baik adalah ketika tulisan dicapture dengan posisi tegak lurus oleh kamera. Ketika posisi kamera tegak lurus kertas, kontur karakter dapat dideteksi secara normal sehingga hasil pengenalan pun juga sesuai. Ketika posisi kamera menyimpang beberapa derajat ke arah sudut tangkap yang lain (tidak tegak lurus), penangkapan citra karakter menghasilkan pola yang cukup berbeda dengan pola aslinya. Ketika perbedaan pola tidak terlalu jauh, sistem masih bisa mengenali karakter yang menyimpang tersebut. Namun ketika sudut tangkap terlalu besar dan menghasilkan pola karakter yang sangat berbeda, sistem sudah tidak bisa mengenalinya dengan baik, sehingga hasil prediksi karakter pun tidak sesuai.

3.9 Pembahasan tentang performa sistem pada perangkat Android

Kompleksitas model JST yang dibentuk berpengaruh pada ukuran memori penyimpanan model tersebut. Semakin banyak komponen (*node*) yang membentuk jaringan, tentunya semakin kompleks juga jaringan yang terbentuk. Dari data Tabel 2 hasil pengujian jumlah node hidden pada pengujian yang telah dilakukan, berisi informasi akurasi yang dicapai oleh beberapa model JST, tampak bahwa beberapa hasil memberikan nilai akurasi yang tinggi. Namun dalam kepentingan pemilihan model JST yang akan diimplementasikan pada perangkat mobile, tidak hanya nilai akurasi saja yang harus diperhatikan. Faktor kompleksitas jaringan, dalam hal ini jumlah node pada hidden layer, juga harus dipertimbangkan. Untuk mengetahui pengaruh model JST yang digunakan terhadap performa sistem pada perangkat Android yang digunakan, dapat dilihat pada Tabel 7. Masing-masing jumlah *node hidden* yang terdapat pada Tabel 7 digunakan pada perangkat Android untuk mengenali sampel tulisan yang sama agar bisa dibandingkan waktu yang dibutuhkan tiap model untuk mengenali jumlah karakter yang sama.

Tabel 7 Pengujian nilai akurasi model JST terhadap pengenalan karakter sampel oleh aplikasi

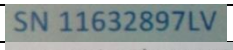
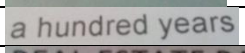
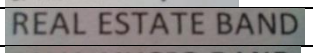
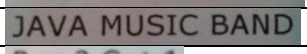
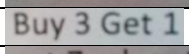
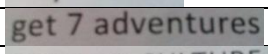
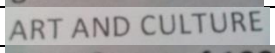
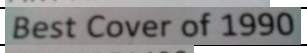
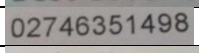
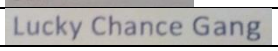
No.	Jumlah *Node Hidden*	Akurasi model JST (%)	Jumlah Karakter Uji	Jumlah Karakter Pengenalan	Waktu (ms)
1.	10	57.46	14	0	547
2.	13	36.66	14	0	648
3.	15	56.85	14	0	697
4.	20	59.41	14	12	1089
5.	23	71.85	14	12	1135
6.	30	85.75	14	12	1193
7.	32	91.49	14	13	1298
8.	37	92.08	14	12	1454
9.	40	92.67	14	13	1596
10.	46	92.37	14	12	1704
11.	48	91.79	14	12	1741
12.	50	97.58	14	14	2399
13.	70	97.31	14	14	2655
14.	150	97.58	14	14	5558
15.	200	96.68	14	14	7313
16.	300	92.96	14	14	10919

Dari Tabel 7 dapat dilihat bahwa akurasi pengenalan karakter yang dapat dicapai oleh aplikasi pada penelitian ini tidak hanya ketika menggunakan model JST dengan 50 node hidden saja yang telah diketahui memiliki akurasi model paling tinggi pada penelitian ini. Beberapa model JST dengan akurasi di bawah 97.58% juga bisa menghasilkan akurasi pengenalan karakter sampel uji yang baik ketika diimplementasikan pada aplikasi. Berdasarkan beberapa percobaan yang dilakukan, dapat dikatakan bahwa untuk mendapatkan akurasi pengenalan karakter sampel uji yang digunakan dengan baik, akurasi model JST yang digunakan pada aplikasi minimal sebesar 92.96%

Selain itu, Tabel 7 juga memberikan informasi lainnya yakni terkait lamanya waktu pemrosesan sistem. Untuk membandingkan lamanya waktu pemrosesan sistem pada perangkat yang digunakan, dapat dilakukan dengan melihat baris nomor 12 hingga 16 pada Tabel 7 yang memberikan hasil pengenalan karakter yang sama. Dari perbandingan lamanya waktu pemrosesan tersebut, dapat dilihat bahwa semakin banyak jumlah node pada hidden layer, lamanya waktu proses pengenalan karakter pun juga semakin meningkat. Hal ini berarti harus menjadi salah satu pertimbangan pada pembuatan aplikasi lainnya yang berbasis jaringan saraf tiruan (JST) untuk perangkat *mobile*.

Jumlah node hidden yang telah diimplementasikan pada penelitian ini adalah sejumlah 50 node hidden. Dari beberapa pengujian yang telah dilakukan dengan model tersebut, lamanya waktu pemrosesan pengenalan karakter pun sudah dihitung untuk mengetahui seberapa lama waktu yang diperlukan untuk mendapatkan hasil pengenalan karakter pada beberapa sampel pengujian yang digunakan pada penelitian ini. Pada dasarnya, proses pewaktuan ini erat kaitannya dengan spesifikasi perangkat keras yang digunakan (*smartphone* Android). Semakin banyak karakter yang akan terdeteksi untuk dikenali, maka semakin lama pula pemrosesannya. Tabel perhitungan rerata waktu yang dibutuhkan sistem dalam penelitian ini dapat dilihat pada pada Tabel 8.

Tabel 8 Rerata waktu pemrosesan oleh sistem pengenalan karakter pada sampel uji

No.	Sampel	Jumlah Karakter Uji	Jumlah Karakter Pengenalan	Waktu (ms)
1.		12	12	980
2.		13	12	1095
3.		14	14	1144
4.		13	12	1070
5.		8	8	682
6.		14	13	1153
7.		13	13	1093
8.		15	15	1259
9.		11	11	902
10.		15	15	1402

Rerata waktu pemrosesan = $\frac{jumlah\ waktu}{jumlah\ karakter\ dikenali} = \frac{10780}{125}$ = **86,24** ms/karakter

Dari Tabel 8, didapatkan informasi rerata lamanya proses pengenalan karakter yakni 87.52 ms per karakter. Besarnya lama waktu tersebut menjadi salah satu hal yang perlu diperhatikan dalam pengembangan sistem yang sejenis seperti pada penelitian ini. Spesifikasi perangkat yang digunakan untuk mendevelop aplikasi semacam ini harus berkapabilitas sesuai dengan beban komputasi yang dijalankan.

Beberapa hasil akurasi yang baik pada pengujian sistem menunjukkan bahwa klasifikasi menggunakan algoritma perambatan balik ternyata juga baik ketika diimplementasikan pada perangkat mobile. Terlepas dari itu, preprocessing pengolahan citra perlu dikembangkan dengan algoritma yang lebih efisien lagi untuk pengembangan penelitian yang sejenis agar sistem tidak terbatas pada beberapa kondisi pemakaian aplikasi (jarak dan arah posisi capture). *Engine Text to Speech* yang dipanggil dalam aplikasi ini juga sudah bisa berjalan dengan baik sehingga bisa mensintesis karakter-karakter sesuai dengan hasil pengenalan yang didapatkan.

4. KESIMPULAN

Dari hasil pengamatan, pengujian, dan analisis pada hasil yang diperoleh, kesimpulan sebagai berikut:

1. Telah berhasil dirancang dan diimplementasikan pengenal karakter pada citra teks berbasis algoritma *backpropagation* untuk aplikasi *text to speech* pada perangkat Android.
2. Model jaringan saraf tiruan dengan pembelajaran algoritma perambatan balik (*back propagation*) yang diimplementasikan pada sistem terdiri dari 400 node pada input layer, 50 node pada hidden layer, dan 62 node pada output layer dengan akurasi jaringan sebesar 97,5806%.
3. Sistem ini mampu mengenali beberapa tipe font yakni Arial, Calibri, dan Verdana. Rerata akurasi pengenalan pada sampel uji yang digunakan di dalam penelitian ini sebesar 94,7% dengan kondisi jarak pengambilan gambar pada rentang jarak 3 – 8 cm dan posisi kamera tegak lurus menghadap kertas tulisan.
4. Sistem yang dibuat bisa mengenali karakter dengan baik pada beberapa variasi warna font pada beberapa variasi warna kertas. Variasi warna yang diujikan yakni warna hitam, merah, dan biru, sedangkan variasi warna kertas HVS yang diujikan yakni berwarna putih, merah, biru, dan kuning.
5. Rerata lamanya waktu proses pengenalan karakter yang dilakukan sistem yakni 86.24 ms per karakter

5. SARAN

Pada penelitian ini masih terdapat banyak hal yang harus disempurnakan. Berikut ini disampaikan saran-saran untuk menyempurnakan penelitian dan sistem yang dibuat.

1. Untuk mengurangi keterbatasan kondisi dalam proses pengambilan gambar (capture) menggunakan kamera smartphone, perlu dikaji lagi dengan tambahan metode lainnya agar proses pengambilan gambar karakter tidak terbatas pada suatu jarak optimum dan arah posisi tertentu saja.
2. Untuk meningkatkan pengenalan karakter terhadap berbagai variasi tipe font, dapat dilakukan dengan penambahan sampel dengan tipe font lain yang ingin ditambahkan.
3. Untuk meningkatkan akurasi proses deteksi karakter dapat ditambahkan algoritma segmentasi yang lebih kompleks agar dapat memisahkan secara sempurna tiap karakter yang berhimpitan.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada semua pihak yang telah memberi dukungan baik secara moril maupun materiil terhadap penelitian ini.

DAFTAR PUSTAKA

[1] Sudjadi, P., 2011, Perancangan Perangkat Lunak untuk Mempercepat Konvergensi pada Backpropagation dengan Adaptasi Laju Pembelajaran, *Thesis*, Jurusan Teknik Elektro Fakultas Teknik, Universitas Diponegoro.
[2] Cheriet, M., 2007, *Character Recognition Systems*, John Wiley & Sons, New Jersey.
[3] Prasojo, A., 2011, Pengenalan Karakter Alfabet Menggunakan Jaringan Saraf Tiruan, *Skripsi*, Jurusan Teknik Elektro Fakultas Teknik Universitas Diponegoro, Semarang.
[4] Bradski.G., Kaehler.A., 2008, *Learning OpenCV*, O'Reilly Media, Inc., California.
[5] Saleh, E.R.M., 2013, Prediksi Masa Kedaluwarsa Wafer Dengan Artificial Neural Network (ANN) Berdasarkan Parameter Nilai Kapasitansi, *Journal AGRITECH*, No.4, Vol.33, 450-457.