SOFTWARE PEMBELAJARAN PENULISAN AKSARA JAWA DENGAN *CONVOLUTIONAL NEURAL NETWORK* SEBAGAI DETEKSI TULISAN TANGAN

SKRIPSI

Oleh :
ULFA HIDAYATI
NIM. 16650068

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

SOFTWARE PEMBELAJARAN PENULISAN AKSARA JAWA DENGAN *CONVOLUTIONAL NEURAL NETWORK* SEBAGAI DETEKSI TULISAN TANGAN

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
ULFA HIDAYATI
NIM. 16650068

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2023

HALAMAN PERSETUJUAN

SOFTWARE PEMBELAJARAN PENULISAN AKSARA JAWA DENGAN *CONVOLUTIONAL NEURAL NETWORK* SEBAGAI DETEKSI TULISAN TANGAN

SKRIPSI

Oleh :
ULFA HIDAYATI
NIM. 16650068

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 23 Mei 2023

Pembimbing I,	Pembimbing II,
A'la Syauqi, M.Kom	Dr. M. Imamudin, Lc., M.A
NIP. 19771201 200801 1 007	NIP. 19740602 200901 1 010

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT, IPM
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

SOFTWARE PEMBELAJARAN PENULISAN AKSARA JAWA DENGAN *CONVOLUTIONAL NEURAL NETWORK* SEBAGAI DETEKSI TULISAN TANGAN

SKRIPSI

Oleh :
ULFA HIDAYATI
NIM. 16650068

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 19 Juni 2023

Susunan Dewan Penguji

Ketua Penguji	: Dr. Irwan Budi Santoso, M.Kom NIP. 19770103 201101 1 004	( )
Anggota Penguji I	: Dr. Cahyo Crysdian NIP. 19740424 200901 1 008	( )
Anggota Penguji II	: A'la Syauqi, M.Kom NIP. 19771201 200801 1 007	( )
Anggota Penguji III	: Dr. M. Imamudin, Lc., M.A NIP. 19740602 200901 1 010	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT, IPM
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Ulfa Hidayati
NIM	: 16650068
Fakultas / Jurusan	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: *Software* Pembelajaran Penulisan Aksara Jawa Dengan *Convolutional Neural Network* Sebagai Deteksi Tulisan Tangan

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambilalihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 15 Juni 2023

Yang membuat pernyataan,

Ulfa Hidayati

NIM.16650068

MOTTO

"Allah Selalu Memberikan yang Terbaik untuk Hambanya"

HALAMAN PERSEMBAHAN

... Skripsi ini saya persembahkan kepada diri sendiri dan kedua orangtua saya yang selalu memberikan dukungan. ...

KATA PENGANTAR

Assalamu'alikum Wr. Wb.

Puji syukur penulis panjatkan atas kehadirat Allah SWT yang telah melimpahkan rahmat dan kasih sayang-Nya, sehingga penulis mampu menyelesaikan skripsi ini dengan baik dan tepat waktu. Sholawat serta salam semoga tercurahkan kepada Nabi besar Muhammad SAW, atas syafaatnya yang telah menuntun umat manusia dari jalan kebatilan ke jalan yang benar. Semoga kita semua masuk ke dalam golongan yang dituntun Allah SWT serta mendapatkan pertolongan Nabi Muhammad SAW. *Aamiin.*

Selanjutnya penulis mengucapkan terima kasih kepada semua pihak yang telah membantu terselesaikannya skripsi ini. Ucapan terima kasih ini penulis sampaikan kepada:

1. Prof. Dr. HM. Zainuddin MA, selaku rektor UIN Maulana Malik Ibrahim Malang.
2. Dr. Sri Harini, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Fachrul Kurniawan, M.MT., IPM selaku Kepala Prodi Teknik Informatika.
4. A'la Syauqi, M,Kom dan Dr. M. Imamuddin, Lc., M.A selaku dosen pembimbing 1 dan dosen pembimbing 2 yang selalu membimbing penyusunan skripsi ini hingga selesai.
5. Segenap civitas akademika Jurusan Teknik Informatika, terutama seluruh dosen, terima kasih atas ilmu dan bimbingan yang telah diberikan.

6. Seluruh dosen Teknik Informatika yang telah memberikan ilmu dan pengalaman yang berharga selama masa perkuliahan.
7. Bapak, ibu, dan adik penulis yang senantiasa memberikan kasih sayang, doa, serta dukungan berupa moril maupun materiil kepada penulis
8. Saudara Teknik Informatika "Andromeda" angkatan 2016 dan seluruh keluarga besar Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang yang selalu memberikan semangat dan bantuan selama menyusun skripsi
9. Semua pihak yang telah membantu dalam penyusunan skripsi ini yang tidak bisa di sebutkan satu persatu.

Malang, 15 Juni 2023

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Hidayati, Ulfa. 2023. **Software Pembelajaran Penulisan Aksara Jawa dengan Convolutional Neural Network Sebagai Deteksi Tulisan Tangan.** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) A'la Syauqi, M.Kom (II) Dr. M. Imamuddin, M.A

Kata kunci: Pengenalan Aksara Jawa, *Convolutional Neural Network*, Deteksi Tulisan Tangan.

Aksara Jawa merupakan aksara tradisional Nusantara yang digunakan untuk menulis bahasa Jawa. Dilakukan upaya pelestarian Aksara Jawa oleh pemerintah salah satunya pada dunia pendidikan. Pembuatan sitem deteksi tulisan tangan aksara jawa dapat membantu dalam pembelajaran aksara jawa. Pendeteksian tulisan tangan dilakukan dengan *deep learning* untuk mendapatkan pengenalan yang baik. Metode yang digunakan pada penelitian ini merupakan metode *Convolutional Neural Network*. Tujuan Penelitian ini untuk mengukur nilai *accuracy*, *precision*, *recall* dan *f-measure*. Metode ini digunakan karena telah banyak penelitian yang menghasilkan akurasi yang sangat baik sebagai deteksi tulisan tangan. Citra yang digunakan diambil dari tulisan tangan anak Sekolah Dasar kelas 4 dan kelas 5. Dilakukan manipulasi data pada citra untuk mendapatkan gambar yang cukup untuk dilakukan *deep learning* seperti *resize*, *grayscale*, *shear*, *zoom*, dan *rotation*. Training dilakukan dengan 350 *epoch* dan dihasikan *accuracy* sebesar 96,2% dan *validation accuracy* sebesar 96%. Dari hasil *training* yang dilakukan, diketahui bahwa semakin banyak *epoch* akan semakin baik pula hasilnya. Dilakukan Pengujian pada model dengan 100 data, dari hasil pengujian diketahui *accuracy* 98,8%, *precision* 88%, *recall* 88%, dan *f-measure* 88%.

ABSTRACT

Hidayati, Ulfa. 2023. **Javanese Alphabet Writing Learning Software Using Convolutional Neural Network as Handwriting Detection.** Thesis. Informatics Engineering Department, Faculty of Science and Technology Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisor: (I) A'la Syauqi, M.Kom (II) Dr. M. Imamuddin, M.A

The Javanese alphabet is the traditional Indonesian alphabet used to write down the Javanese language. The government conserves the alphabet by implementing it in the education world. Making the Javanese alphabet handwriting detection system can help its learning process. The detection is done using deep learning to achieve a sufficient introduction. The researcher employs the Convolutional Neural Network method. The research objective is to measure the accuracy, precision, recall, and f-measure level. The researcher chose the method since many researchers have used it to generate high-accuracy handwriting detection. He took the handwriting samples of four and five-grade elementary students. He manipulated the image data to get adequate information about the image to carry out deep learning, such as resizing, grayscale, shear, zoom, and rotation. The training was carried out for 350 epochs and resulted in an accuracy of 96.2% and a validation accuracy of 96%. The training result shows that the more epochs, the better the result. The testing on the model of 100 data shows the accuracy, precision, recall, and f-measure of 98.8%, 88%, 88%, and 88%, respectively.

Keywords: Javanese Alphabet Introduction, Convolutional Neural Network, Handwriting Detection.

الملخص

هدايتي، أولفى. ٢٠٢٣. **برنامج تعلم كتابة الأبجدية الجاوية مع الشبكات العصبونية الإلتفافية ككشف للخط اليدوي**. البحث الجامعي. قسم الهندسة المعلوماتية، كلية العلوم والتكنولوجيا بجامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف الأول: أعلى شوقي، الماجستير. المشرف الثاني: د. محمد إمام الدين، الماجستير.

الكلمات الرئيسية: التعرف على الأبجدية الجاوية، الشبكات العصبونية الإلتفافية، كشف خط اليد.

الأبجدية الجاوية هي أبجدية تقليدية للأرخبيل تستخدم لكتابة اللغة الجاوية. بذلت الحكومة جهودا للحفاظ عليها، وكان أحدها في عالم التعليم. يمكن أن يساعد إنشاء نظام الكشف عن خط اليد الجاوي في تعلم النص الجاوي. يتم الكشف عن خط اليد من خلال التعلم العميق للحصول على اعتراف جيد. الطريقة المستخدمة في هذا البحث هي طريقة الشبكات العصبونية الإلتفافية. الهدف من البحث هو قياس قيمة الدقة والضبط والاستدعاء والفاء القياسي. تم استخدام هذه الطريقة لأن هناك عدة الدراسات التي تنتج دقة ممتازة مثل الكشف عن خط اليد. الصور المستخدمة مأخوذة من خط يد طلاب المدارس الابتدائية في الصفين الرابع والخامس. تم إجراء معالجة البيانات على الصورة للحصول على صور كافية للتعلم العميق مثل تغيير الحجم وتدرج الرمادي والقص والتكبير أو التصغير والدوران. تم إجراء التدريب على ٣٥٠ حقبة وأسفر عن دقة ٩٦.٢% وضبط ٩٦%. من نتائج التدريب الذي تم إجراؤه، من المعروف أنه كلما زاد عدد العصور، كانت النتائج أفضل. أجري الاختبار على نماذج تحتوي على ١٠٠ بيانات، من نتائج ذلك الاختبار وجدت قيمة الدقة ٩٨.٨%، الضبط ٨٨%، الاستدعاء ٨٨%، و الفاء القياسي ٨٨%.

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Indonesia mempunyai kekayaan budaya yang melimpah. Keanekaragaman suku bangsa dan bahasa adalah harta yang tak ternilai. Beberapa bahasa daerah memiliki sistem tulisan khusus yang digunakan di wilayahnya, seperti aksara Jawa. Aksara Jawa tersebar di seluruh provinsi di Pulau Jawa. Aksara Jawa adalah bentuk tulisan tradisional Nusantara untuk menulis bahasa Jawa. Aksara Jawa merupakan bentuk tulisan modern dari aksara kawi dan merupakan salah satu bentuk turunan aksara brahmi yang berkembang di Jawa (Prihantono, 2011).

Aksara Jawa memiliki beberapa variasi, seperti aksara carakan, aksara pasangan, sandhangan, aksara murda, aksara rekan, aksara swara, dan angka Jawa (Lestari, 2009). Aksara carakan (abjad Jawa) yang digunakan dalam penulisan bahasa Jawa secara umum terdiri dari dua puluh aksara dasar yang bersifat suku kata (Nusantara, 2002). Sebuah karya tulis (Rizal & Karuniawan, 2018) telah menyebutkan bahwa upaya untuk melestarikan aksara Jawa sedang dilakukan oleh pemerintah. Salah satu cara yang dilakukan adalah menyertakan aksara Jawa dalam kurikulum pendidikan, agar nilai budaya Indonesia tetap terjaga.

Dalam bidang pendidikan, metode pembelajaran dapat disampaikan melalui penggunaan media pembelajaran atau alat peraga pembelajaran (Rizal & Karuniawan, 2018). Pengembangan metode pembelajaran saat ini dapat dilakukan dengan memanfaatkan teknologi komputer, khususnya dalam perangkat lunak (Widihastuti & Khosyidin, 2012). Pembuatan teknologi oleh manusia dengan

tujuan mempermudah dalam melakukan sesuatu sebagaimana yang telah dijelaskan dalam Q.S. Al-Insyirah ayat 6 sebagai berikut :

إِنَّ مَعَ ٱلْعُسْرِ يُسْرًا

"*Sesungguhnya sesudah kesulitan itu ada kemudahan*" (QS. AL-Insyirah:6)

Kemajuan software yang semakin cepat sangat membantu dalam menjaga warisan budaya. Sebuah media pembelajaran tentang penulisan aksara Jawa bisa dibuat dengan menggunakan teknologi komputer.

Perkembangan teknologi yang semakin canggih dapat memudahkan dalam berbagai aspek. Sebagaimana telah banyak *software* yang dapat mempermudah dalam berkomunikasi dan belajar berbagai bahasa. Dengan demikian dapat dibangun sebuah *software* pembelajaran aksara Jawa yang dapat mendeteksi benar tidaknya suatu tulisan tangan. Tulisan tangan merupakan suatu hal yang unik dari setiap orang. Pengenalan tulisan tangan menggunakan *deep learning* telah diimplementasikan pada tulisan tangan latin, Cina, Arab, Persia, dan Bangla (Wibowo *et al*., 2017). Terdapat berbagai penelitian mengenai pendeteksi tulisan tangan menggunakan *machine learning* (Dewa *et al*., 2018). *Machine learning* merupakan bidang ilmu baru yang mana bidang ini menggunakan kumpulan data dan menggunakan data tersebut untuk mempelajari pola sehingga mendapatkan pengetahuan tertentu. Ketika proses pembelajaran tentang pola-pola yang ada semakin dalam, itu akan menjadi *deep learning*. *Deep learning* adalah proses pembelajaran *artificial neural network* di mana pola dipelajari lebih dalam untuk menghasilkan hasil yang lebih baik. *Deep learning* meniru kemampuan otak manusia untuk mengingat sesuatu (Sudana *et al.,* 2020). Beberapa peneliti

menyebutkan bahwa pengenalan tulisan tangan menggunakan teknik *deep learning* telah menghasilkan akurasi yang lebih tinggi dari pada menggunakan teknik *conventional machine learning* (Wibowo *et al.,* 2017). Sehingga pada penelitian kali ini akan menggunakan *deep learning* sebagai teknik dalam pendeteksian tulisan tangan aksara Jawa.

Arsitektur dalam model *deep learning* memegang peranan penting dalam kemampuan jaringan untuk mempelajari data yang ada. Terdapat beberapa model dalam *deep learning*, seperti *convolutional neural network*, *deep neural network* dan *recurrent neural network* (Krizhevsky *et al.,* 2012). Telah dikatakan dengan bantuan teknik modern seperti *convolutinal neural networks* mereka dapat memindai dan memahami kata-kata dengan keakuratan yang belum pernah ada sebelumnya dalam sejarah (Shubham Sanjay Mor, 2019). Sehingga dalam penelitian ini digunakan metode CNN. Penggunaan CNN dalam penelitian ini juga didukung oleh penelitian (Rismiyati *et al.,* 2017) yang telah melakukan penerapan alogaritma *deep learning*, CNN dan DNN untuk melakukan pengenalan karakter tulisan aksara Jawa (20 karakter dasar) dan mendapatkan hasil pengujian yang menunjukkan model CNN mampu mencapai akurasi terbaik.

Pada penelitian ini akan dilakukan penelitian mengenai akurasi pada 480 karakter tulisan aksara Jawa. Data sampel tulisan tangan yang digunakan merupakan tulisan tangan dari siswa/siswi kelas 3-5 SD/MI/sederajat. Data tersebut nantinya akan diproses untuk dijadikan data *training* dan data *validation.* Diharapkan pada penelitian ini akan dihasilkan nilai akurasi deteksi tulisan tangan yang lebih baik dari penelitian sebelumnya. Sehingga penelitian ini dapat

memudahkan siswa dalam mempelajari penulisan aksara Jawa. Dengan platform android juga diharap dapat dijangkau dengan mudah oleh siswa dan guru.

1.2 Pernyataan Masalah

Seberapa besar tingkat *accuracy*, *precision*, recall dan f-measure metode *deep learning* dengan *model Convolutional Neural Network* (CNN) untuk mendeteksi tulisan tangan aksara Jawa?

1.3 Batasan Masalah

Oleh karena banyaknya variasi dari aksara Jawa maka perlu adanya pembatasan masalah yang diambil sehingga penelitian dapat fokus. Adapun batasan masalah sebagai berikut :

1. Data citra yang digunakan merupakan tulisan tangan aksara siswa SD kelas 4 dan 5.
2. Subjek yang digunakan dalam penelitian sebanyak 20 subjek, yaitu aksara jawa *carakan* tanpa *pasangan, sandhangan, murda, rekan, swara*, dan angka Jawa.

1.4 Tujuan Penelitian

Mengukur tingkat *accuracy*, *precision*, recall dan f-measure metode *deep learning* dengan *model Convolutional Neural Network* (CNN) untuk mendeteksi tulisan tangan aksara Jawa.

1.5 Manfaat Penelitian

Manfaat yang diharapkan pada penelitian ini adalah :

1. Universitas dapat mengukur kemampuan mahasiswa akan ilmu yang didapatkan selama menempuh pendidikan.
2. Penelitian ini dapat memberikan kemudahan bagi yang ingin belajar aksara jawa menggunakan *software* deteksi tulisan tangan aksara jawa yang telah dibangun.

BAB II

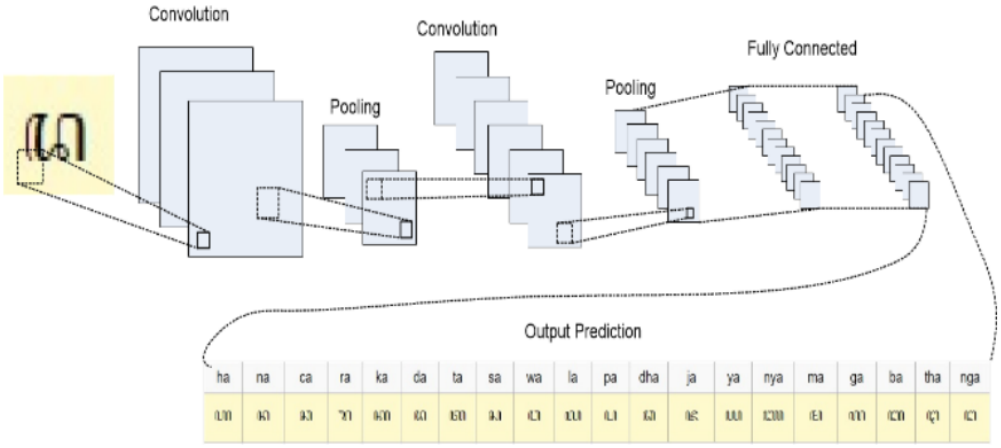
TINJAUAN PUSTAKA

Pada penelitian Wibowo et al (2017) dikatakan bahwa banyak peneliti menyebutkan pengenalan tulisan tangan menggunakan teknik *deep learning* telah menghasilkan akurasi lebih tinggi dibandingkan degan teknik *conventional machine learning*. Hal Tersebut mendorong penggunaan deep learning pada penelitian ini. Penelitian ini menggunakan objek aksara Jawa dan menggunakan *Convolutional Neural Network* (CNN).

CNN atau Convolutional Neural Network adalah metode deep learning yang sangat efektif dalam menyelesaikan tugas klasifikasi data dengan menggunakan dataset gambar sebagai input. Hal ini disebabkan karena CNN menggunakan informasi tetangga piksel dalam proses ekstraksi fitur menggunakan operasi konvolusi dan pooling antara input dan kernel. Dataset pada penelitian tersebut berjumlah total 11.000 aksara Jawa. Setiap karakter dibagi menjadi 56 data.

Peneliti menggunakan 2 model CNN. Perbedaan dari kedua model tersebut adalah jumlah total lapisan pada setiap model. Model 1 terdiri dari 3 layer *convolution* 2D, 3 layer *pooling* dan 1 layer *fully connected*. Kemudian model 2 hanya menambahkan 1 layer *fully connected* di akhir arsitektur sebelum klasifikasi. Model tersebut menggunakan algoritma *Stochastic Gradient Descent* untuk meminimalkan *error rate* dan melakukan *tuning* pada parameter *hyper* seperti *momentum, learning rate*, metode regularisasi, dan fungsi aktivasi pada kedua model CNN. Setiap model dijalankan menggunakan *hyperparameter* standar MNIST seperti 0,006 untuk *learning rate*, 0,9 untuk *momentum*, 1e-4 untuk

regularisasi L2, *ReLU* untuk fungsi aktivasi, kemungkinan *log negatif* untuk fungsi kerugian, dan *backpropagation* untuk kesalahan intervensi yang digunakan dalam pembaruan bobot. Adapun skema penelitian sebagai berikut:



Gambar 2.1 Skema Penelitian

Akurasi terbaik pengenalan karakter bahasa Jawa menggunakan CNN mencapai 94,57% dengan *learning rate* 0,01 dan regularisasi L2 0,0005 menggunakan model ke 2. Hasil kinerja yang baik diperoleh dalam penelitian ini. Hasil penelitian menunjukkan bahwa *deep learning* tipe diskriminatif seperti CNN menjadi model terbaik untuk mengenal karakter Jawa yang memiliki akurasi, *precision*, *recall*, dan F-1 dengan nilai masing-masing 94.57%, 94.75%, 94.57%, dan 94.66%.

Penelitian pengenalan tulisan tangan aksara Jawa juga dilakukan oleh (Dewa *et al.,* 2018). Penelitin ini menggunakan model *Convolutional Neural Network* (CNN) Penelitian ini membandingkan model CNN dan MLP pada sistam yang dibangun. Data yang digunakan pada penelitian ini sejumlah 2000 data yang terdiri dari 20 class data dari dasar aksara Jawa. Data yang digunakan di-*grayscale*

dan berukuran 28 x 28 piksel. Sistem yang dikembangkan memanfaatkan deteksi kontur dan deteksi tepi *canny* menggunakan *library OpenCV* terhadap citra karakter aksara Jawa untuk proses segementasi. Berikut arsitektur CNN yang digunakan pada penelitian ini :

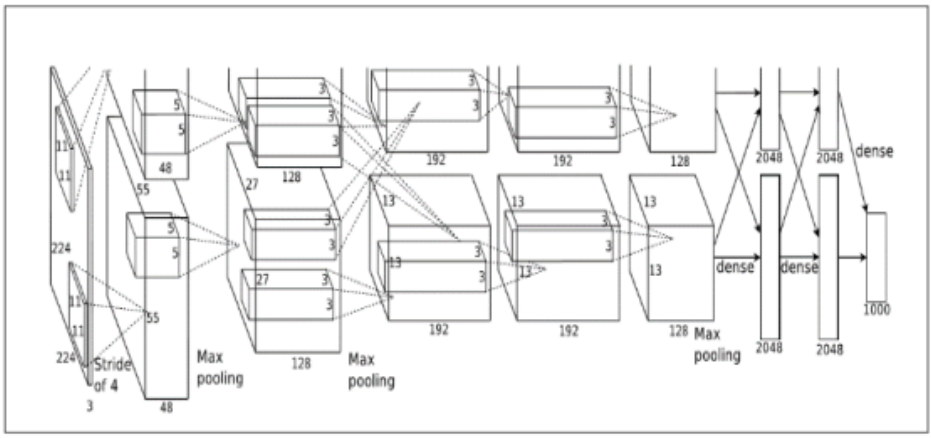
Tabel 2.1 Arsitektur CNN (Dewa *et al* ., 2018)

Layer Type	Size	Output Shape
Input	(1, 28, 28)	-
Convolution + ReLU	32 (3 x 3) filters	(32, 26, 26)
Max Pooling + Dropout	(2 x 2) filters	(32, 13, 13)
Convolution + ReLU	64 (2 x 2) filters	(64, 12, 12)
Max Pooling + Dropout	(2 x 2) filters	(64, 6, 6)
Convolution + ReLU	128 (3 x 3) filters	(128, 4, 4)
Max Pooling + Dropout	(2 x 2) filters	(128, 2, 2)
Fully-Connected + ReLU + Dropout	1,000 neurons	20
Softmax	20 way	20

Penelitian ini menghasilkan akurasi yang baik pada model CNN dibanding dengan MLP. Hasil penggunaan CNN tidak menghasilkan akurasi mencapai 90%, kemungkinan hal ini terjadi karena jumlah data yang tidak mencukupi untuk pembelajaran yang lebih mendalam pada metode ini.

Berdasarkan penelitian (Dewa *et al.,* 2018) diketahui bahwa model CNN sangat baik digunakan untuk pengenalan tulisan tangan aksara Jawa dibandingkan dengan model MLP. Dari penelitian tersebut maka penelitian ini menggunakan model CNN agar mendapatkan hasil yang memuaskan. Kemudian berdasarkan penelitian (Wibowo *et al.,* 2017) diketahui bahwa arsitektur CNN yang baik untuk pengenalan tulisan tangan aksara Jawa menggunakan 3 *convolutional layer* , *3 pooling layer* , dan *2 fully connected layer*. Arsitektur pada penelitian tersebut diadobsi pada penelitian ini untuk mendapatkan akurasi yang baik seperti pada penelitian tersebut.

Dilakukan penelitian mengenai pengenalan tulisan tangan *hangul* (Purnamawati *et al.,* 2018) dengan menggunakan metode *deep convolutional neural network.* Metode yang digunakan mempunyai beberapa tahap dimulai dari *pre-trained neural network* dengan model *inception-V3* sebagai outcome. Lalu yang kedua yaitu *retraining process* menggunakan teknik *transfer learning.* Kemudian yang terakhir yaitu proses pengenalan kata. Pada proses pengenalan kata dilakukan dengan tahapan *image acquisition, preprocessing,* dan pengenalan dengan *Deep Covolutional Neural Network.*

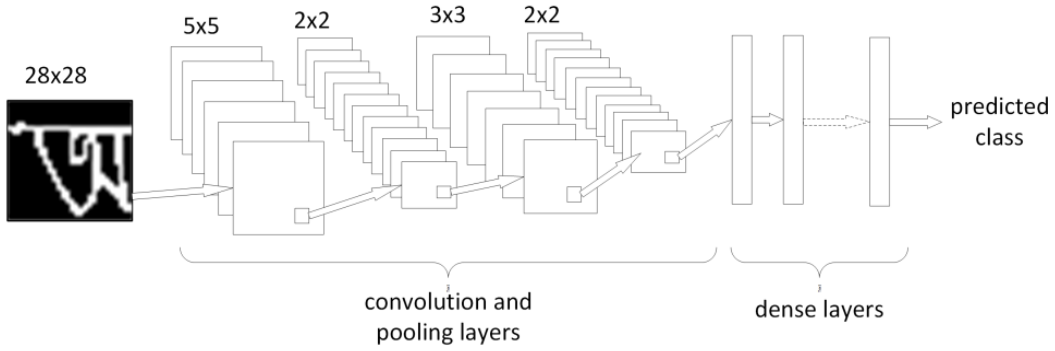


Gambar 2.2 Arsitektur *Deep Convolutional Neural Network*

Gambar diatas merupakan arsitektur *Deep Convolutional Neural Network* pada penelitian tersebut. DCNN terdiri dari 5 lapisan konvolusional dan tiga lapisan yang terhubung sepenuhnya. Pada pengenalan kata terdapat tahapan penginputan citra melalui *smartphone* android, tahapan *preprocessing* yang terdiri dari *tresholding* dan *resizing*, tahapan pengenalan menggunakan DCNN, dan tahap *ouput* yaitu hasil dari pengenalan berupa terjemahnya. Dataset pada penelitian ini menggunakan 11 kata Korea dengan 1157 citra.

Berdasarkan hasil pengujian pengenalan tulisan tangan huruf Korea menggunakan *Deep Convolutional Neural Network* pada platform android diperoleh rata-rata akurasi sebesar 86,9%. Nilai akurasi citra masukan dipengaruhi oleh bentuk guratan Hangul. 100% akurasi pelatihan diperoleh di setiap kata yang memiliki 100 citra. Jumlah citra yang dilatih di setiap kata memengaruhi akurasi pelatihan.

Telah dilakukan penelitian pengenalan tulisan tangan karakter Bengali (Purkaystha *et al.,* 2017). Penelitian ini menggunakan *Deep Convolutional Neural Network* dengan 2 layer *convolutional* , 2 layer *pooling* dan 3 layer *dense fullyconnected*. Pada akhir *dense* merupakan *softmax layer*. Telah diterapkan *dropout* pada setiap lapisan pula untuk mengurangi *overfit* data. Lapisan terakhir adalah lapisan *softmax* yang mengeluarkan distribusi probabilitas di atas kelas untuk masukan yang diberikan. Berikut gambaran dari arsitekturnya :



Gambar 2.3 CNN untuk pengenalan tulisan tangan karakter Bengali

Metode ini sangat efektif untuk mengenali tulisan tangan karakter Bengali. Pada penelitian ini menghasilkan *accuracy* 98,66% pada tipe angka dengan 10 kelas, 94,99% pada tipe vokal dengan 11 kelas, 91,60% pada tipe huruf dasar dengan 20

kelas, 91,23% pada tipe Alfabet dengan 50 kelas dan 89,93% pada tipe semua karakter dengan 80 kelas.

Pembuatan aplikasi android menggunakan Android Studio dengan menautkan sistem pengenalan teks tulisan tangan menggunakan *tensorflow* telah dilakukan oleh (Shubham Sanjay Mor, 2019). Pengenalan pada peneltian ini menggunkan Convolutinal Neural Network. Penelitian ini menggunakan kumpulan data EMNIST untuk melatih model mereka dan menguji memilih adamax sebagai optimasi. Penelitian ini memilih pengoptimalan adamax karena menghasilkan akurasi *training* yang tinggi pada setiap *epoch*-nya.

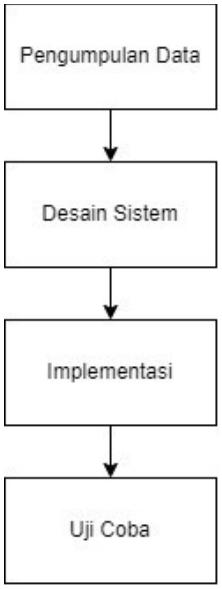
Preprocessing pada penelitian ini menggunakan normalisasi, *rotation* dan *reversing* data, dan *image filtering* (*resize* dan *grayscale*). Model CNN dibangun menggunakan KERAS dan *tensorflow*. Layer 1 dilakukan *reshape* menjadi (28,28,1). Layer 2 dan layer 3 merupakan *convolution* layer dengan filter matriks (5,5) , *stride* 1, dan menggunakan fungsi aktivasi ReLU. Layer 4 merupakan *max pooling* dengan ukuran (2,2) dan ukuran *stride* 2. Layer 5 merupakan *flatten* yang berguna untuk menyiapkan data untuk diproses pada layer setelahnya. Layer 6 merupakan layer *dense* yang berfungsi untuk aktivasi *neuron*. Layer 7 merupakan layer *dropout* yang berguna untuk menghapus *neuron* atau fitur tidak diinginkan yang dapat membuat model menjadi tebal dan menghabiskan banyak waktu *training*. Layer 8 merupakan layer terakhir yaitu *dense* layer yang sering disebut output layer. *Training* ini menghasilkan akurasi sebesar 89,53% dan menghasilkan akurasi sebesar 87,1% pada prediksinya.

BAB III

DESAIN DAN IMPLEMENTASI

3.1 Alur Penelitian

Penelitian memerlukan suatu kerangka penelitian untuk memastikan penelitian tersebut terorganisir dan teratur. Berikut langkah-langkah alur penelitian ini:



Gambar 3.1 Alur Penelitian

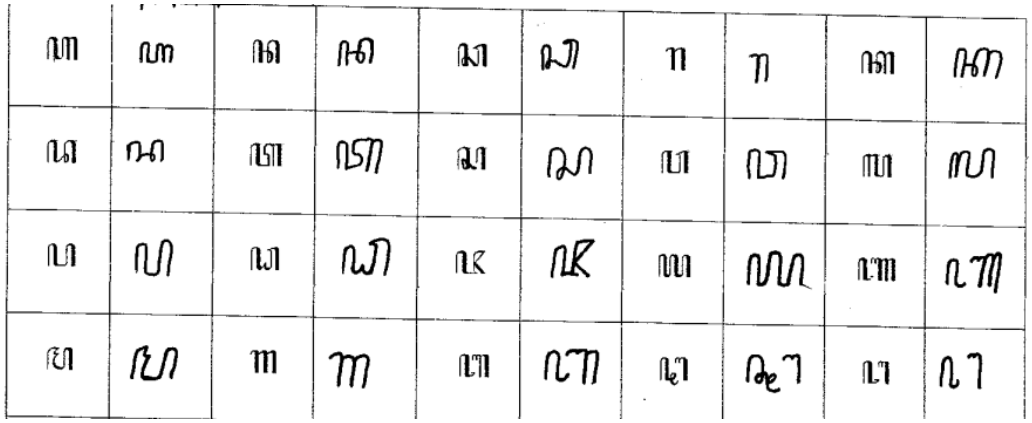
Berdasarkan gambar di atas diketahui bahwa penelitian ini diawali dengan pengumpulan data dan diakhiri dengan uji coba. Alur penelitian ini yaitu pengumpulan data, desain sistem, implementasi dan uji coba. Pada tahap pengumpulan data merupakan pengumpulan data tulisan tangan yang akan digunakan untuk implementasi sistem dan uji coba..

Kemudian langkah yang kedua adalah desain sistem yaitu langkah yang dibutuhkan untuk penggambaran sistem yang akan dibangun pada penelitian ini.

Selanjutnya langkah ketiga yaitu implementasi, pada tahap ini dilakukan pengimplementasian *Convolutional Neural Network* pada sistem *recognition* yang dibangun. Pengimplementasian CNN pada training akan menghasilkan model yang akan digunakan untuk membangun aplikasi android. Kemudian yang terakhir yaitu uji coba pada sistem yang telah dibangun dengan data yang telah disiapkan.

3.2 Pengumpulan Data

Penelitian ini membutuhkan objek berupa foto tulisan aksara Jawa yang digunakan untuk mengidentifikasi tulisan nantinya. Telah dijelaskan pada latar belakang di atas data diambil dari tulisan tangan siswa/siswi kelas 4 dan 5 SD/MI pada kertas yang telah disiapkan beserta contohnya. Data yang didapatkan sebanyak 20 dataset dengan masing-masing sebanyak 75 data. Data yang telah didapatkan dari tulisan tangan tersebut di-scan dan disimpan berupa file pdf. Berikut salah satu contoh hasil scan data tulisan tangan siswa :



Gambar 3.2 Hasil Scan Data Aksara Jawa

Setelah kertas di-scan, citra akan di-*crop* untuk memisahkan tiap karakternya. Citra hasil *crop* berukuran 112 x 112 piksel. Setiap karakter tersebut dikelompokkan sesuai dengan kelompoknya masing-masing. Setelah dataset telah

siap nantinya data akan dibagi untuk *training* dan *validation.* Berikut merupakan contoh data hasil *crop* dan pengelompokan :

Gambar 3.3 Contoh Dataset Ha

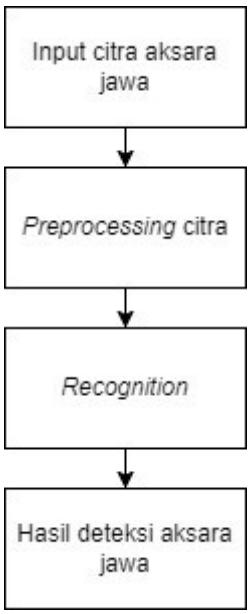
Dilakukan pula pengumpulan data yang nantinya digunakan untuk testing model. Data yang digunakan merupakan data hasil tulisan tangan dari 5 orang. Sehingga didapatkan 100 data untuk *testing* model dan aplikasi. Data ini diambil dari kamera handphone dengan ukuran 3000 x 3000 piksel. Berikut contoh citra yang digunakan untuk testing sistem dan aplikasi android.

Gambar 3.4 Data Testing

3.3 Desain Sistem

Software yang akan dibangun merupakan aplikasi android yang dapat mendeteksi tulisan tangan aksara Jawa. Dalam penelitian ini menggunakan metode *deep learning* dengan model *Convolutional Neural Network*. Pada pendeteksian citra dilakukan dengan pemindaian citra melalui kamera dan akan langsung

dideteksi tulisan tersebut. Untuk dapat memindai citra tersebut dibutuhkan pembelajaran pola dari Aksara Jawa yang disebut model. Oleh karena itu, dilakukan *training* untuk mendapatkan model yang dibutuhkan. Berikut merupakan desain sistem secara umum mengenai jalannya aplikasi :

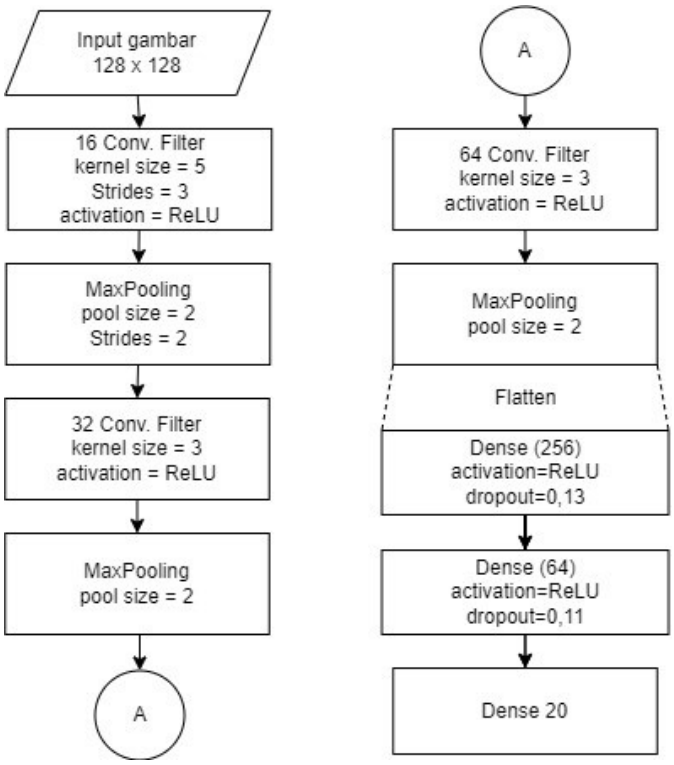


Gambar 3.5 Desain Sistem Aplikasi Android

Gambar di atas merupakan gambaran dari desain sistem android, diawali dengan memasukkan citra aksara Jawa ke sistem oleh pengguna. Setelah diinputkan, sistem akan melakukan *preprocessing* pada gambar untuk menyesuaikan gambar dengan yang dibutuhkan sistem untuk melakukan pengenalan. *Preprocessing* yang dilakukan yaitu *resize* dan *grayscale image* , sistem akan melakukan pengenalan pada citra(*recognition*) yang telah diinput untuk mengetahui hasil deteksi yang dilakukan. Pengenalan ini membutuhkan model yang dihasilkan dari implementasi *Convolutional Neural Network*. Setelah dilakukan pengenalan sistem akan menampilkan output berupa tulisan latin aksara Jawa.

3.4 Implementasi *Convolutional Neural Network*

Penelitian pengenalan tulisan tangan ini menggunakan metode *Convolutional Neural Network*. Berikut arsitektur *Convolutional Neural Network* yang digunakan :



Gambar 3.6 Arsitektur *Convolutional Neural Network*(CNN)

Diagram di atas merupakan arsitektur CNN yang telah dikembangkan dari penelitian Wibowo *et al.,* (2017). Berikut penjelasan dari diagram di atas :

a) Sebelum memasukkan citra, dilakukan *preprocecing* pada citra untuk mengoptimalkan kinerja model dengan mempersiapkan data masukan (input) sehingga mudah dikenali oleh model.

b) Citra yang telah dimanipulasi dimasukkan dan dilakukan proses konvolusi yang pertama. Saat konvolusi pertama, citra akan dikalikan dengan *kernel* berukuran 5x5 dan *filter* sebanyak 16 *filter.* Proses perkalian dilakukan dengan menggeser *kernel* sebanyak 3 *stride.* Kernel adalah matriks, yang digeser melintasi citra dan dikalikan dengan input sedemikian rupa kemudian di jumlahkan. Hasil matriks tersebut biasa disebut dengan *feature map*. Berikut rumus untuk *feature map* dan konvolusi:

$$n_{out} = \left(\frac{n_{in} - k + 2p}{s}\right) + 1 \quad (1)$$

Keterangan:

n_{out} : ukuran feature map

n_{in} : ukuran matriks masukan

k : ukuran matriks

p : ukuran padding

s : *stride*

$$FM[i]_{j,k} = \left(\sum_m \sum_n N_{[j-m,k-n)} F_{[m,n]} + bF\right) \quad (2)$$

Keterangan:

FM[i] : matriks *feature map* ke-i

N : matriks citra masukan

F : matriks filter konvolusi

bF : nilai bias pada filter

j, k : posisi piksel pada matriks citra masukan

m, n : posisi piksel pada matriks filter konvolusi

Proses Konvolusi ini menggunakan aktivasi *Rectified Linear Unit* (ReLU). Sehingga setiap piksel pada *feature map* akan dimasukkan ke dalam fungsi ReLU, berikut rumus yang digunakan :

$$F(x) = \max(0, x) \quad (3)$$

Keterangan:

F[x] : nilai keluaran

x : nilai masukan

c) Proses *pooling* merupakan proses yang bertujuan untuk mengurangi dimensi dari *feature map*. Pengurangan dimensi ini disebut juga dengan *down sampling.* Proses *pooling ini* dapat mempercepat komputasi karena parameter yang harus di-*update* semakin sedikit. Penelitian ini yang menggunakan *max pooling* dengan ukuran 2x2 dan *stride* sebanyak 2 *stride*, maka nilai yang di ambil dari setiap pergeseran filter adalah nilai terbesar dalam rentang 2x2.

d) Pada konvolusi kedua dilakukan konvolusi dengan filter sebanyak 32, dengan *kernel* berukuran 5x5 dan *stride* sebanyak 3 *stride* . Layer ini menggunakan aktivasi ReLU juga, seperti layer konvolusi pertama.

e) Dilakukan proses *pooling* kedua seperti yang dilakukan sebelumnya.

f) Pada konvolusi ketiga dilakukan konvolusi dengan filter sebanyak 64, dengan *kernel* berukuran 3x3 dan *stride* sebanyak 3 *stride* . Layer ini menggunakan aktivasi ReLU juga, seperti layer konvolusi pertama.

g) Dilakukan proses *pooling* ketiga seperti yang dilakukan sebelumnya.

h) Sebelum masuk ke tahap *fully connected layer,* dilakukan proses yang disebut *flatten*. *Flatten* merupakan operasi yang mengubah matriks menjadi vektor satu

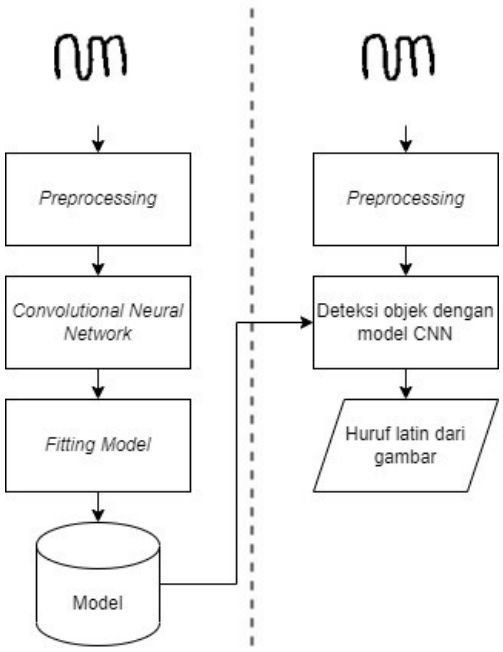
dimensi agar *feature map* yang dihasilkan dapat diklasifikasikan dengan *fully-connected* layer. Selanjutnya tahap *backpropagation* layer, layer ini mengubah vektor kembali menjadi matriks seperti dimensi semula yang selanjutnya dilakukan proses perubahan bobot filter.

i) Proses *fully connected* layer atau disebut dengan *dense* dilakukan 2 kali pada akhir proses. *Fully-connected* layer merupakan layer yang terdiri dari beberapa *node*. *Fully connected* yang pertama ditunjukkan dengan *dense* (256), maksudnya adalah terdapat neuron pada lapisan tersebut. Aktivasi dengan metode ReLu digunakan pada *fully connected* layer yang pertama ini. *Fully connected* layer kedua merupakan output layer yang ditunjukkan dengan *dense num_class*, maksudnya di sini mengikuti dengan jumlah kelas yang terdapat pada sistem. Pada penelitian ini memiliki 20 kelas dari tulisan tangan aksara Jawa, sehingga terdapat 20 *neuron*. *Fully connected* layer terakhir ini menggunakan aktivasi *softmax.*

3.4.1 Training

Untuk melakukan pembelajaran aksara Jawa pada sistem, dilakukan *training* terlebih dahulu. Pada proses ini algoritma dilatih dengan data *training* yang telah disiapkan. Pada proses *training* ini citra yang dimasukkan akan melalui proses *training* dengan metode *Convolutional Neural Network* yang akan menghasilkan suatu model yang nanti akan diuji untuk mengetahui akurasinya. Tujuan dari *training* ini adalah untuk menemukan ciri dari setiap citra kemudian menandai neuron-neuron mana yang akan diaktifkan ketika citra diklasifikasi.

Setelah melakukan proses *training* selanjutnya model yang telah didapatkan digunakan untuk proses *testing*. Pada proses *testing*, model yang digunakan adalah model yang sebelumnya didapatkan pada saat melakukan *training*. Proses *training* dan *testing* dapat dilihat pada *flowchart* berikut :



Gambar 3.7 *Training* Data Aksara Jawa

Pada gambar 3.7 di atas terdapat model, model tersebut didapatkan dengan beberapa tahapan pengolahan yang disebut *training.* Pada proses *training* dilakukan input data, *preprocecccing, learning* citra dengan *Convolutional Neural Network* dan nantinya menghasilkan model. Model ini akan digunakan untuk testing dan pembuatan aplikasi nantinya.

Pada Gambar 3.7 dapat dilihat bahwa proses *training* terdiri dari input data, *preprocessing*, *Convolutinal Neural Network*, *fitting* model dan diakhiri dengan hasil berupa model. Berikut penjelasan dari proses pada *training* yang dilakukan:

a. Input

Pada tahapan ini merupakan input data *training* berupa citra tulisan tangan aksara Jawa. Data yang telah disiapkan dibagi menjadi 2 macam yaitu data *train*ing dan data *validation* . Data *train*ing digunakan untuk proses *learning* dan data *validation* digunakan untuk validasi data saat *learning* berlangsung. Tahapan awal dari proses ini yaitu dengan menyiapkan direktori yang digunakan untuk di inputkan. Folder yang telah disiapkan bernamakan DATA dan DATASET. Pada folder DATA terdapat 20 folder dengan 75 data pada tiap foldernya. Kemudian disiapkan fungsi untuk menelusuri direktori, pengambilan informasi seperti *path, subdirectory* dan *files* menggunakan *library* os.walk. Informasi yang didapatkan disimpan pada variabel *path_data*, *tag_dataset,* dan *nama_file.* Berikut source code yang digunakan untuk memuat citra:

```
def loadData(dataset_p, nama_file, tag_dataset, path_data):
  for path, subdirs, files in os.walk(dataset_p):
      for name in files:
          path_data.append(os.path.join(path, name))
          tag_dataset.append(path.split('/')[-1])
          nama_file.append(name)
  return(nama_file, tag_dataset, path_data)
```

Gambar 3.8 *Source Code*-Fungsi Muat Data

Setelah menjalankan fungsi di atas data akan disimpan ke *dataframe* agar lebih rapi dan mudah dibaca menggunakan fungsi yang telah disediakan oleh padas, yaitu *dataframe.* Berikut *source code* yang digunakan untuk membuat *dataframe*:

```
namaFile = []
tagData = []
pathData = []
loadData(data_path, namaFile, tagData, pathData)
df = pd.DataFrame({"Path":pathData,'Nama
File':namaFile,"Tag":tagData})
df.groupby(['Tag']).size()
```

Gambar 3.9 *Source Code*-Pembuatan *Dataframe*

Setelah data disimpan dalam datafame, data akan di-split menjadi 2 dataset yaitu data *train* dan data *validation* dengan pembagian 90% data *train* dan 10% data *validation*. Selanjutnya dilakukan inisialisasi variabel data frame untuk menyimpan data yang sudah di pisahkan. Kemudian data tersebut disimpan ke dalam folder *train* dan validation yang berisikan 20 folder berbagai macam tulisan tangan aksara jawa. Berikut kode yang digunakan:

```
X= df['Path']
y= df['Tag']
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.1, random_state=100)
df_tr = pd.DataFrame({'Path':X_train
              ,'Tag':y_train
             ,'Set':'train'})
df_val = pd.DataFrame({'Path':X_test
              ,'Tag':y_test
             ,'Set':'validation'})
```

Gambar 3.10 *Source Code-Split Dataset*

Setelah data di*split* menjadi 2, dibuat folder train dan validation untuk menyimpan data yang telah di*split.*

b. *Preprocessing*

Pada tahapan ini yaitu penyesuaian data untuk memudahkan dalam pengidentifikasian citra yang dijadikan model sistem. Selain itu dilakukan penyesuaian dan manipulasi citra untuk memperbanyak citra dikarenakan citra yang dimiliki hanya sebanyak 75 citra dan terlalu sedikit untuk dilakukan pembelajaran. Manipulasi citra ini menggunakan parameter sebagai berikut *shear_range* = 0,2, *zoom_range* = 0, *horizontal_flip* = *false*, *rotation* =40, *width_shift* = 0,2, *height_shift* = 0,2. Selain itu citra diubah ukurannya menjadi 128 x 128, kemudian dilakukan *grayscale* atau pengubahan citra menjadi hitam putih. *Grayscale* disini akan sangat membantu dalam memudahkan *learning* yang dilakukan karena dapat mengurangi kompleksitas komputasi dan waktu pelatihan model, serta meningkatkan akurasi dalam beberapa tugas klasifikasi citra. Berikut source code fungsi yang digunakan untuk manipulasi data dengan imageGenerator:

```
def imageGenerator(jml_rotasi, lebar, tinggi, shear,
zoom, horizontalFlip):
  imageGen = ImageDataGenerator(
                                rescale=1./255,
                                rotation_range=jml_rotasi
,
                                width_shift_range=lebar,
                                height_shift_range=tinggi
,
                                shear_range=shear,
                                zoom_range=zoom,
                                horizontal_flip=
horizontalFlip)
  return(imageGen)
datagen = imageGenerator(jml_rotasi, lebar, tinggi,
shear, zoom, horizontalFlip)
```

Gambar 3.11 Source Code-Memperbanyak Citra

c. Convolutional Neural Network (CNN)

Pada tahap ini citra yang telah preproceccing akan dipelajari dengan metode Convolutional Neural Network sesuai dengan arsitektur pada gambar 3.6. Berikut kode arsitektur CNN yang digunakan:

```
model = Sequential()
model.add(Conv2D(16, (5, 5), padding='same',
input_shape=inp_shape))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(256))
model.add(Activation('relu'))
model.add(Dropout(0.13))
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.11))
model.add(Dense(num_class))
model.add(Activation('softmax'))
```

Gambar 3.12 *Source Code*-Arsitektur CNN

Setelah dijalankan kode di atas, akan didapatkan arsitektur model *convolutional neural network*. Arsitektur model *convolutional neural network* tersebut dapat dilihat pada gambar 3.12 berikut:

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 128, 128, 16)	1216
activation (Activation)	(None, 128, 128, 16)	0
max_pooling2d (MaxPooling2D)	(None, 64, 64, 16)	0
conv2d_1 (Conv2D)	(None, 62, 62, 32)	4640
activation_1 (Activation)	(None, 62, 62, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	18496
activation_2 (Activation)	(None, 31, 31, 64)	0
max_pooling2d_2 (MaxPooling2D)	(None, 15, 15, 64)	0
conv2d_3 (Conv2D)	(None, 15, 15, 128)	73856
activation_3 (Activation)	(None, 15, 15, 128)	0
max_pooling2d_3 (MaxPooling2D)	(None, 7, 7, 128)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 256)	1605888
activation_4 (Activation)	(None, 256)	0
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 64)	16448
activation_5 (Activation)	(None, 64)	0
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 20)	1300
activation_6 (Activation)	(None, 20)	0

Total params: 1,721,844
Trainable params: 1,721,844
Non-trainable params: 0

Gambar 3.13 Arsitektur Model CNN

Setelah didapatkan arsitektur model *convolutional neural network*, di *compile* menggunakan optimasi *adam* dengan nilai *learning* rate sebesar 0,001. Berikut *source code* yang digunakan untuk *compile* model:

```
from tensorflow.keras.optimizers import Adam
print('Compiling Model.......')
optimize=Adam(learning_rate=0.001)
model.compile(optimizer=optimize,
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Gambar 3.14 *Source Code-Compile Model*

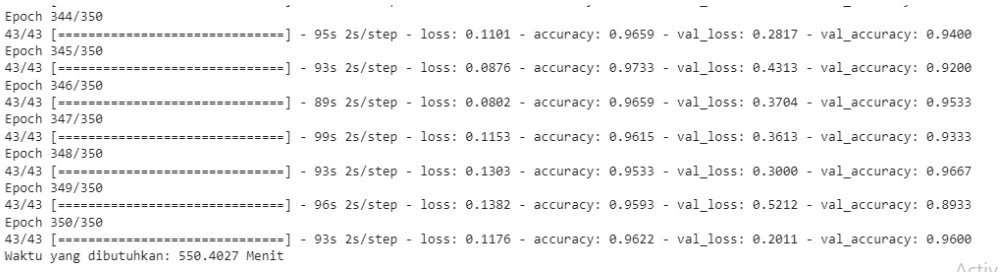
d. Fitting Model

Setelah model di-*compile*, dilakukan *fitting model* dengan *eppoch* sebanyak 350 dan *batch size* sebesar 32. *Epoch* sebanyak 350 dan batch size sebesar 32 yang berarti data akan dipelajari sebanyak 350 kali dengan 32 sampel data yang diberikan ke dalam model pada setiap iterasi atau pembaruan bobot. Berikut *source code* yang digunakan untuk *fitting* model:

```
start = time.time()
cnn_hist = model.fit(x = train_data,
          steps_per_epoch=len(train_generator),
          epochs=350,
          validation_data=val_data,
          validation_steps=len(val_generator),
          shuffle=shufle,
          verbose = 1)
end = time.time()
print("Waktu yang dibutuhkan: {:5.4f} Menit".format((end-
start)/60.0))
```

Gambar 3.15 *Source Code-Fitting Model*

Source code di atas merupakan kode yang digunakan untuk *fitting* model. Dari *fitting* model yang dilakukan didapatkan hasil sebagai berikut:



Gambar 3.16 Hasil *Training*

e. Output Model

Output yang didapatkan berupa model. Model disimpan kemudian harus diubah terlebih dahulu menjadi .tflite agar model dapat digunakan pada aplikasi android yang telah dibuat. Menyimpan model menggunakan kode seperti berikut:

```
model.save('/content/drive/MyDrive/Colab
Notebooks/model')


modelku=
tf.keras.models.load_model('/content/drive/MyDrive/Colab
Notebooks/model')


converter=tf.lite.TFLiteConverter.from_keras_model(modelk
u)
converter.experimental_new_converter=True
tflite_model=converter.convert()
open("/content/drive/MyDrive/Colab
Notebooks/converted_model.tflite","wb").write(tflite_mode
l)
```

Gambar 3.17 *Source Code*-Menyimpan dan *Convert* Model

3.4.2 Implementasi Pada Aplikasi Android

Setalah didapatkan model dari *training* yang dilakukan, model akan digunakan untuk membangun aplikasi android. Model tersebut telah dilengkapi dengan dua bahasa pemprograman. Bahasa pemprograman yang disediakan yaitu *kotlin* dan *java*. Pada penelitian ini aplikasi dibangun menggunakan bahasa pemprograman *java*. Tampilan yang dibangun sebagai berikut:

Gambar 3.18 Tampilan Aplikasi Android

Aplikasi android dibangun dengan cukup sederhana dengan inputan citra melalui kamera dan galeri ponsel. Untuk membangun aplikasi perlu dilakukan *import* model terlebih dahulu ke dalam projek pada android studio. *Tensorflow* telah mendukung pembuatan *image recognition* dengan baik, sehingga telah disediakan *source code* yang akan digunakan untuk dimasukkan pada sistem.

Source code yang telah disediakan tersebut harus disesuaikan kembali dengan model yang telah kita buat. Penyesuaian yang dilakukan yaitu resize dan grayscale image. *Source code* yang disediakan dimodifikasi seperti berikut:

```
public void classifyImage(Bitmap image){
    try {
        ConvertedModel model =
ConvertedModel.newInstance(getApplicationContext());
        // Creates inputs for reference.
        TensorBuffer inputFeature0 =
TensorBuffer.createFixedSize(new int[]{1, 128, 128, 3},
DataType.FLOAT32);
        ByteBuffer byteBuffer = ByteBuffer.allocateDirect(4 *
imageSize * imageSize * 3);
        byteBuffer.order(ByteOrder.nativeOrder());
        int[] intValues = new int[imageSize * imageSize];
        image.getPixels(intValues, 0, image.getWidth(), 0, 0,
image.getWidth(), image.getHeight());
        int pixel = 0;
        for(int i = 0; i < imageSize; i ++){
            for(int j = 0; j < imageSize; j++){
                int val = intValues[pixel++]; // RGB
                byteBuffer.putFloat(((val >> 16) & 0xFF) * (1.f
/ 255));
                byteBuffer.putFloat(((val >> 8) & 0xFF) * (1.f /
255));
                byteBuffer.putFloat((val & 0xFF) * (1.f / 255));
            }
        }
        inputFeature0.loadBuffer(byteBuffer);
        ConvertedModel.Outputs outputs =
model.process(inputFeature0);
        TensorBuffer outputFeature0 =
outputs.getOutputFeature0AsTensorBuffer();
        float[] confidences = outputFeature0.getFloatArray();
        int maxPos = 0;
        float maxConfidence = 0;
        for (int i = 0; i < confidences.length; i++) {
            if (confidences[i] > maxConfidence) {
                maxConfidence = confidences[i];
                maxPos = i;
            }
        }
        String[] classes = {"ba", "ca", "da", "dha", "ga", "ha",
"ja", "ka", "la", "ma", "na", "nga",
                "nya", "pa", "ra", "sa", "ta", "tha", "wa",
"ya"};
        result.setText(classes[maxPos]);
        model.close();
    } catch (IOException e) {
        // TODO Handle the exception
    }
}
```

Gambar 3.19 Source Code-Fungsi Klasifikasi

Setelah projek dijalankan, aplikasi android yang dibangun dapat berjalan dengan baik dan sesuai dengan model yang telah dibuat.

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Skenario Pengujian

Proses uji coba (*testing*) terdapat beberapa proses, di antaranya input citra, *preprocessing* citra, *recognition* citra, dan diakhiri dengan output. Proses ini akan dilakukan teradap 100 data uji coba yang telah disiapkan untuk mengetahui ketepatan deteksi dari model yang telah dibuat. Berikut penjelasan dari setiap proses testing:

a. Input Citra

   Citra yang telah disiapkan dimuat dan di simpan menjadi *dataframe* seperti halnya pada *training.*

b. *Preprocessing* Citra

   Preprocessing citra dilakukan untuk menyesuaikan dengan model CNN yang telah dibuat. Untuk menyesuaikan dilakukan olah citra berupa *resize* dan *grayscale*. Berikut kode yang digunakan:

```
def preprocess(img,ukuran_image):
    nimg = img.convert('RGB').resize(ukuran_image)
    img_arr = (np.array(nimg))/255
    return img_arr

def reshape(imgs_arr):
    return np.stack(imgs_arr, axis=0)
```

Gambar 4.1 *Source Code*-Fungsi Untuk *Resize* dan *Grayscale* Citra

c. *Recognition* dengan model CNN

Testing dilakukan menggunakan model yang telah didapatkan pada proses training, pada tahapan ini model dimuat dan digunakan untuk mengenali citra yang diinputkan. Berikut source code yang digunakan.

```
prediksi = []
hasil_prediksi = []
nilai = []
pred_bnr = 0

for gmb_test in range (0, len(path_file_uji)):
  gambar_uji = Image.open(path_file_uji[gmb_test])
  h = preprocess(gambar_uji,gambar_size)
  X = reshape([h])

  model_test = load_model(model_uji,compile=False)
  y = model_test.predict(X)
  prediksi.append(label[np.argmax(y)])

  if(label[np.argmax(y)] == tag_file_uji[gmb_test]):
    hasil_prediksi.append('BENAR')
    pred_bnr+=1
    nilai.append(np.max(y))

  else:
    hasil_prediksi.append('SALAH')
    nilai.append(np.max(y))
```

Gambar 4.2 Source Code-Proses Testing

d. Output

Pada beberapa proses yang telah dilakukan akan mendapatkan output berupa nilai prediksi dan hasil prediksi berupa tulisan latin dari citra yang diinputkan. Setelah nilai dan hasil uji coba didapatkan, dilakukan perhitungan dengan matriks konfusi untuk mengetahui ketepatan sistem dalam mendeteksi tulisan tangan

aksara Jawa. Matriks konfusi adalah representasi tabular untuk jumlah prediksi hasil yang benar dan salah dari model yang dievaluasi. Berikut merupakan matriks konfusi :

Tabel 4.1 Matriks Konfusi

Nilai Prediksi	Nilai Sebenarnya	
	TRUE	FALSE
TRUE	TP	FP
FALSE	FN	TN

Dari tabel 4.1 terdapat TP(*True Positive*), FP(*False Positive*),TN(*True Negative*) dan FN(*False Negative*) yang dapat dijelaskan sebagai berikut :

True Positive(TP): merupakan tulisan tangan aksara Jawa yang diprediksi sistem sebagai kelas *i* dan terverifikasi prediksi benar.

True Negative(TN): merupakan tulisan tangan aksara Jawa yang diprediksi sistem bukan kelas *i* dan terverifikasi prediksi benar.

False Positive(FP): merupakan tulisan tangan aksara Jawa yang diprediksi sistem bukan kelas *i* dan terverifikasi prediksi salah.

False Negative(FN): merupakan tulisan tangan aksara Jawa yang diprediksi sebagai kelas *i* dan terverifikasi prediksi salah.

- *Accuracy*

  Accuracy merupakan nilai yang menunjukkan tingkat kedekatan antara nilai prediksi sistem deteksi tulisan tangan aksara Jawa dengan nilai prediksi manusia. Nilai Accuracy dapat dicari dengan persamaan berikut :

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \; x \; 100\% \qquad (4)$$

- *Precision*

  Precision merupakan nilai sensitifitas atau nilai ketepatan deteksi yang diberikan oleh sistem deteksi tulisan tangan aksara jawa untuk menunjukkan tulisan tangan aksara Jawa diprediksi dengan benar atau salah. Nilai *precision* dapat dicari dengan menggunakan rumus persamaan sebagai berikut :

$$Precision = \frac{TP}{TP + FP}\ x\ 100\% \quad (5)$$

- *Recall*

  Nilai *Recall* menunjukkan tingkat keberhasilan atau spesifisitas sistem deteksi tulisan tangan aksara jawa untuk mengetahui kembali sebuah informasi tentang aksara jawa yang terprediksi secara benar dan salah. Nilai *Recall* dapat dicari dengan menggunakan rumus persamaan berikut :

$$Recall = \frac{TP}{TP + FN}\ x\ 100\% \quad (6)$$

- *F-measure*

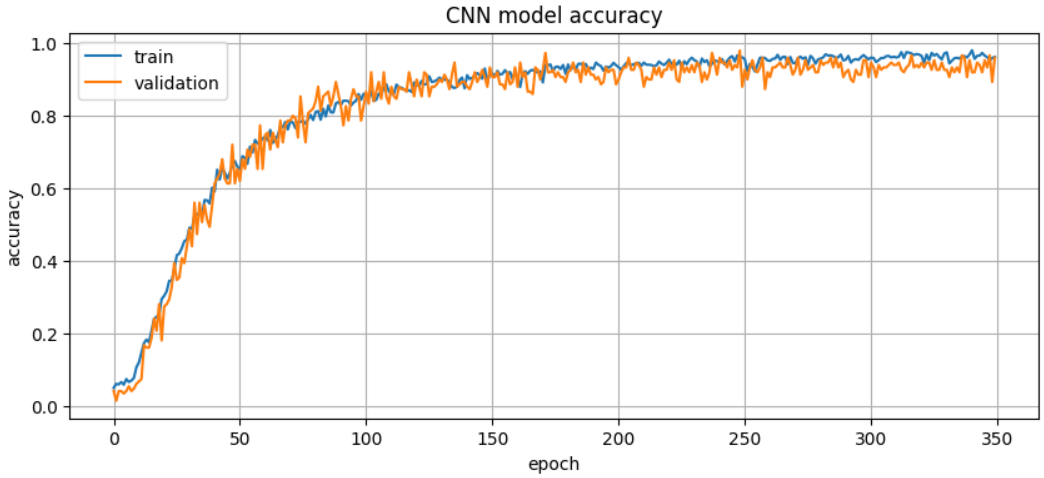
  F-measure disebut juga dengan *f1-score* yang digunakan untuk mengukur rata-rata harmonik antara *Precision* dan *Recall. F-measure* dapat dihitung dengan rumus berikut :

$$F - measure = \frac{2\ x\ Preciision\ x\ Recall}{Precision + Recall} \quad (7)$$

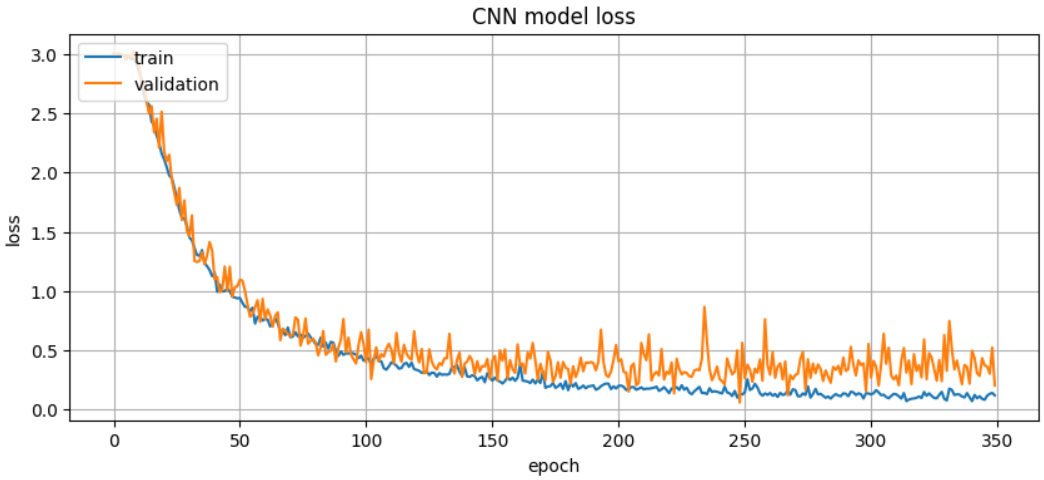
4.2 Hasil *Train*ing

Proses *train*ing pengenalan tulisan tangan aksara jawa dengan metode Convolutional Neural Network (CNN) dilakukan sesuai dengan arsitektur yang telah dijelaskan pada bab sebelumnya. Proses *train*ing ini dilakukan dengan 350

epoch, yang berarti dilakukan *learning* sebanyak 350 kali pada data yang dimiliki . Sehingga didapatkan hasil akurasi sebesar 0,96 dan validasi akurasi sebesar 0,96 dengan waktu selama 550.4027 menit (9 jam 10 menit). sebagai berikut :



Gambar 4.3 Akurasi *Train*ing dan Validation



Gambar 4.2 Loss *Train*ing dan Validation

Pada gambar 4.2 dan 4.3 di atas menunjukkan bahwa semakin banyak epoch akan semakin bagus pula hasil pembelajarannya. Selain itu, semakin banyak epoch yang dilakukan juga menunjukkan bahwa nilai loss yang dihasilkan juga mengalami penurunan (semakin kecil).

Setelah dilakukan *train*ing dan mendapatkan hasil yang cukup memuaskan, model *convert* menjadi model tflite. Model tflite ini akan digunakan untuk membangun aplikasi android, model harus berupa model tflite, karena aplikasi android akan dibangun menggunakan aplikasi open *source* android studio dengan menggunakan bahasa pemprograman java.

4.3 Hasil Uji Coba Model

Sebelum model diimplementasikan ke-dalam aplikasi android, dilakukan uji coba terhadap model yang telah didapatkan. Model akan di *test* menggunakan data *test* yang telah disiapkan. *Test* ini bertujuan untuk mengetahui akurasi yang di dapatkan dari model yang telah dibuat.

Persiapan model yang akan digunakan dalam test sudah melewati 2 tahapan yaitu tahap citra yang mengubah gambar menjadi data dan preprosesing citra yang mengubah data tersebut sesuai dengan kebutuhan. Setelah melewati kedua tahap peneliti melakukan tahapan berikutnya yaitu tahapan cnn yang mengubah kumpulan informasi tersebut menjadi hasil yang diiingkan. Hasil yang diperoleh sebagai berikut

Tabel 4.2 Hasil *Test* Model

		Prediksi																		
Aktual	Kelas	Ha	Na	Ca	Ra	Ka	Da	Ta	Sa	Wa	La	Pa	Dha	Ja	Ya	Nya	Ma	Ga	Ba	Tha	Nga
	Ha	4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	Na	0	3	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Ca	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Ra	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Ka	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Da	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Ta	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0
	Sa	0	3	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
	Wa	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0
	La	2	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0
	Pa	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0
	Dha	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0
	Ja	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0
	Ya	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0
	Nya	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0
	Ma	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
	Ga	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0
	Ba	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2
	Tha	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0
	Nga	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5

Dari tabel hasil uji coba model pengenalan tulisan tangan aksara Jawa menggunakan metode *Convolutional Neural Network* di atas diketahui bahwa uji coba model dilakukan pada 100 citra. Dari 100 citra data uji coba dihasilkan 88 citra dengan prediksi benar dan 12 citra dengan prediksi salah. Sehingga didapatkan nilai True Positive(TP), nilai *True Negative*(TN), nilai *False Positive*(FP) dan nilai *False Negative*(FN) seperti berikut:

Tabel 4.3 Nilai TP, TN, FP dan FN Pada Setiap Kelas

Kelas	TP	TN	FP	FN
Ha	4	94	1	1
Na	3	93	2	2
Ca	5	95	0	0
Ra	5	95	0	0
Ka	5	95	0	0
Da	5	95	0	0
Ta	5	95	0	0
Sa	2	92	3	3
Wa	5	95	0	0
La	3	93	2	2
Pa	5	95	0	0
Dha	5	95	0	0
Ja	5	95	0	0
Ya	5	95	0	0
Nya	5	95	0	0
Ma	4	94	1	1
Ga	5	95	0	0
Ba	2	92	3	3
Tha	5	95	0	0
Nga	5	95	0	0
Total	88	1888	12	12

Dari nilai yang didapatkan akan dilakukan perhitungan dengan konfusion matrix untuk mengetahui *accuracy*, *precision*, *recall* dan *f-measure.*

a. *Accuracy*

Untuk mengetahui *Accuracy* dari uji coba model, maka dilakukan perhitungan dengan rumus berikut :

$$Accuracy = \frac{88 + 1888}{88 + 1888 + 12 + 12}\ x\ 100\%$$

$$Accuracy = 98{,}8\%$$

b. *Precision*

Untuk mengetahui *Precision* dari uji coba model yang dilakukan, maka dilakukan perhitungan sebagai berikut :

$$Precision = \frac{88}{88 + 12} \; x \; 100\%$$

$$Precision = 88\%$$

c. *Recall*

Untuk mengetahui *Recall* dari uji coba model yang dilakukan, maka dilakukan perhitungan sebagai berikut :

$$Recall = \frac{88}{88 + 12} \; x \; 100\%$$

$$Recall = 88\%$$

d. *F-measure*

Untuk mengetahui *F-Measure* dari uji coba model yang dilakukan, maka dilakukan perhitungan sebagai berikut :

$$F - Measure = \frac{2 \; x \; 88 \; x \; 88}{88 + 88} x100 \; \%$$

$$F - Measure = 88\%$$

4.5 Pembahasan

Aplikasi Pengenalan Tulisan Tangan Aksara Jawa pada penelitian ini menggunakan Convolutional Neural Network(CNN) sebagai metode yang dilakukan dalam pembelajaran sistem. Aplikasi android yang dibangun dapat mengenali 20 aksara jawa carakan tulisan tangan dengan cara difoto maupun

memasukkan dari galeri smartphone yang dimiliki. Aplikasi ini dibangun dengan API minimal 23 atau android 8(Oreo), sehingga dapat digunakan oleh pengguna android 8 hingga pengguna android terbaru. Aplikasi ini dapat mendeteksi tulisan aksara jawa carakan dengan nilai akurasi sebesar 98,8%, nilai *Precision* sebesar 88%, nilai recall sebesar 88% dan nilai f-measure sebesar 88%. Dari penelitian ini diketahui bahwa eppoch sangat mempengaruhi hasil dari training yang dilakukan, semakin besar epoch yang dilakukan akan semakin bagus akurasi yang didapatkan. Dengan akurasi yang memuaskan ini, aplikasi dapat digunakan untuk mempelajari *aksara jawa carakan*, di mana aplikasi ini dapat membantu untuk mendeteksi tulisan latin dari huruf aksara jawa, sehingga dapat membantu dalam mempelajari aksara jawa.

Software pembelajaran aksara jawa dibangun untuk mempelajari tulisan aksara jawa dengan mudah. Mempelajari tulisan aksara jawa merupakan bentuk menuntut ilmu. Sebagai manusia kita diwajibkan untuk menuntut ilmu, yang tertuang dalam ayat Al-Qur'an surah Al-Mujadalah ayat 11 sebagai berikut :

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْٓا اِذَا قِيْلَ لَكُمْ تَفَسَّحُوْا فِى الْمَجٰلِسِ فَافْسَحُوْا يَفْسَحِ اللّٰهُ لَكُمْۚ وَاِذَا قِيْلَ انْشُزُوْا فَانْشُزُوْا يَرْفَعِ اللّٰهُ الَّذِيْنَ اٰمَنُوْا مِنْكُمْۙ وَالَّذِيْنَ اُوْتُوا الْعِلْمَ دَرَجٰتٍۗ وَاللّٰهُ بِمَا تَعْمَلُوْنَ خَبِيْرٌ

"*Hai orang-orang beriman apabila dikatakan kepadamu: "Berlapang-lapanglah dalam majlis", maka lapangkanlah niscaya Allah akan memberi kelapangan untukmu. Dan apabila dikatakan: "Berdirilah kamu", maka berdirilah, niscaya Allah akan meninggikan orang-orang yang beriman di antaramu dan orang-orang yang diberi ilmu pengetahuan beberapa derajat. Dan Allah Maha Mengetahui apa yang kamu kerjakan.*"(QS. Al-Mujadalah:11)

Penafsiran Tafsir Jalalain (Tafsirq, n.d) tentang ayat ini berkaitan dengan pentingnya menuntut ilmu dan partisipasi dalam majelis ilmiah atau kajian agama.

Ayat ini menekankan pentingnya keaktifan dalam mencari ilmu dan berpartisipasi dalam majelis ilmiah untuk belajar dan berdiskusi tentang agama Islam. Allah SWT akan memberikan kelapangan ilmu dan pemahaman kepada mereka yang aktif berpartisipasi dalam majelis ilmiah. Artinya, Allah akan mempermudah jalan untuk memperoleh ilmu pengetahuan bagi mereka yang sungguh-sungguh berusaha dan aktif dalam menuntut ilmu. Allah SWT akan mengangkat derajat dan kedudukan orang yang beriman dan aktif dalam menuntut ilmu. dengan mempelajari aksara Jawa akan menambah wawasan dan mempermudah mengetahui budaya Jawa.

Menuntut ilmu merupakan salah satu tuntutan agama yang sangat ditekankan. Allah SWT telah memerintahkan umat-Nya untuk senantiasa meningkatkan ilmu dan pengetahuan, serta menggunakan pengetahuan tersebut untuk berbuat kebajikan dan kebaikan. Menurut konsep *mu'amalah ma'a* Allah pada surah Al-Mujadalah ayat 11, penelitian ini berpengaruh dalam meningkatkan ilmu dan pengetahuan kita, baik dalam hal agama maupun dunia. Kita juga harus memperoleh ilmu dari sumber yang terpercaya dan mengamalkannya dengan baik untuk mendapatkan pahala dari Allah SWT dan berbuat kebaikan bagi diri sendiri dan orang lain. Mematuhi perintah Allah SWT untuk menuntut ilmu termasuk dalam *muamalah ma'a* Allah.

Allah menciptakan manusia berbangsa-bangsa dan bersuku-suku untuk saling mengenal, seperti pada Ayat Al-Qur'an surah Al-Hujurat ayat 13 seperti berikut :

يَٰٓأَيُّهَا ٱلنَّاسُ إِنَّا خَلَقۡنَٰكُم مِّن ذَكَرٖ وَأُنثَىٰ وَجَعَلۡنَٰكُمۡ شُعُوبٗا وَقَبَآئِلَ لِتَعَارَفُوٓاْۚ إِنَّ أَكۡرَمَكُمۡ عِندَ ٱللَّهِ أَتۡقَىٰكُمۡۚ إِنَّ ٱللَّهَ عَلِيمٌ خَبِيرٞ

"Hai manusia, sesungguhnya Kami menciptakan kamu dari seorang laki-laki dan seorang perempuan dan menjadikan kamu berbangsa-bangsa dan bersuku-suku supaya kamu saling kenal-mengenal. Sesungguhnya orang yang paling mulia di antara kamu di sisi Allah ialah orang yang paling takwa di antara kamu. Sesungguhnya Allah Maha Mengetahui lagi Maha Mengenal."(QS.Al-Hujurat:13)

Pada tafsir jalalain (Tafsirq, n.d) ayat ini menjelaskan tentang penciptaan manusia oleh Allah SWT dari dua jenis kelamin, yaitu laki-laki dan perempuan. Allah menciptakan manusia dalam bentuk beragam bangsa, suku, dan suku bangsa yang berbeda-beda, dengan tujuan agar mereka saling mengenal dan berinteraksi antara satu sama lain. Allah menyatakan bahwa keutamaan seseorang tidak ditentukan oleh suku, bangsa, atau warna kulit, tetapi ditentukan oleh ketakwaannya kepada Allah. Orang yang paling mulia di sisi Allah adalah mereka yang paling bertakwa. Ketakwaan kepada Allah adalah faktor penentu keutamaan manusia, bukan asal usul atau keturunan mereka. Ayat ini juga mengajarkan pentingnya toleransi, saling menghormati, dan saling mengenal antara bangsa dan suku yang berbeda dalam masyarakat. Sebagai umat manusia, kita harus menghargai perbedaan dan belajar untuk saling mengenal, memahami, dan berinteraksi dengan cara yang baik dan damai. Aplikasi pengenalan aksara Jawa ini sebagai bentuk pengenalan budaya dan suku Jawa agar lebih mudah dalam mempelajari dan memahami suku Jawa. Menurut konsep *mu'amalah ma'a An-nas* (interaksi dengan sesama manusia), ayat ini mengajarkan pentingnya saling menghormati, saling mengenal, dan saling berinteraksi dengan adab.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Telah dilakukan penelitian mengenai deteksi tulisan tangan aksara jawa menggunakan metode Convolutional Neural Network (CNN) menggunakan 1.500 data *training* dan 100 data *testing.* Penelitian ini mendapatkan hasil *Accuracy* sebesar 98,8% , *Recall* sebesar 88%, *Precision* sebesar 88% dan *F-measure* sebesar 88%. Aplikasi android yang telah dibangun berjalan dengan baik sesuai dengan model yang telah dibangun dan citra dapat diinputkan baik melalui kamera maupun dari penyimpanan ponsel.

5.2 Saran

Setelah dilakukan penelitian terhadap deteksi tulisan tangan aksara jawa menggunakan metode Convolutional Neural Network (CNN), peneliti menyadari bahwa sistem yang dibangun belum sempurna. Sehingga perlu dilakukan pengembangan terhadap sistem untuk mendapatkan hasil yang lebih baik. Berikut saran kepada para peneliti selanjutnya:

1. Penelitian ini hanya berfokus pada tulisan tangan aksara jawa carakan, sehingga diharapkan pada penelitian selanjutnya dapat melakukan pengenalan terhadap aksara jawa *carakan* dengan *sandhangan* dan pasangan-nya.
2. Penelitian ini hanya berfokus pada pendeteksian 1(satu) huruf aksara jawa saja dalam sekali input, diharapkan pada penelitian selanjutnya dapat melakukan pengenalan terhadap kata dan kalimat aksara jawa.

DAFTAR PUSTAKA

Dewa, C. K., Fadhilah, A. L., & Afiahayati. (2018). Convolutional Neural Networks for Handwritten Javanese Character Recognition. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, 83-94.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in neural information processing systems*, 1097-1105.

Lestari, E. D. (2009). *Kawruh Sapala Basa.* Klaten: Intan Pariwara.

Nusantara, Y. P. (2002). *Pedoman Penulisan Aksara Jawa.* Yogyakarta: Yayasan Pustaka Nusantara.

Prihantono, D. (2011). *Sejarah Aksara Jawa.* Yogyakarta: Java Litera.

Purkaystha, B., Datta, T., & Islam, M. S. (2017). Bengali Handwritten Character Recognition Using Deep Convolutional Neural Network. *2017 20th International Conference of Computer and Information Technology (ICCIT)* (hal. 1-5). Bangladesh: IEEE.

Purnamawati, S., Rachmawati, D., Lumanauw, G., Rahmat, R. F., & Taqyuddin, R. (2018). Korean letter handwritten recognition using deep convolutional neural network on android platform. *Journal of Physics: Conference Series*, 1-9.

Rismiyati, Khadijah, & Nurhadiyatna, A. (2017). Deep *Learning* for Handwritten Javanese Character Recognition. *2017 1st International Conference on Informatics and Computational Sciences (ICICoS)* (hal. 59-64). Semarang: IEEE.

Rizal, R. S., & Karuniawan, J. S. (2018). Perancangan Multimedia Pembelajaran Aksara Jawa Dengan Macromedia Flash MX Pada SMP Negeri 1 Balapulang Kabupaten Tegal. *Jurnal Sistem Informasi Dan Teknologi Informasi*, 132-139.

Shubham Sanjay Mor, S. S. (2019). Handwritten Text Recognition : with Deep *Learning* and Android. *International Journal of Engineering and Advanced Technology (IJEAT)*, 819-825.

Sudana, O., Gunaya, I. W., & Putra, I. K. (2020). Handwriting identification using deep convolutional neural network method. *TELKOMNIKA Telecommunication, Computing, Electronics and Control*, 1934-1941.

Wibowo, M. A., Soleh, M., Pradani, W., Hidayanto, A. N., & Arymurthy, A. M. (2017). Handwritten Javanese Character Recognition using Descriminative

Deep *Learning* Technique. *Information Systems and Electrical Engineering (ICITISEE)*, 325-330.

Widihastuti, I., & Khosyidin, M. (2012). Aplikasi Interaktif Pembelajaran Aksara Jawa. *Majalah Ilmiah Sultan Agung*, 1-8.

LAMPIRAN-LAMPIRAN

Lampiran 1

Hasil *Training*

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
1	103s 2s/step	30.031	0.0489	29.982	0.0400
2	105s 2s/step	29.960	0.0600	30.004	0.0133
3	103s 2s/step	29.946	0.0585	30.095	0.0400
4	90s 2s/step	29.924	0.0652	29.888	0.0400
5	101s 2s/step	29.854	0.0578	29.941	0.0333
6	97s 2s/step	29.831	0.0733	29.527	0.0400
7	95s 2s/step	29.799	0.0652	29.896	0.0533
8	93s 2s/step	29.608	0.0689	29.746	0.0400
9	88s 2s/step	29.597	0.0763	30.239	0.0467
10	100s 2s/step	29.306	0.1059	29.342	0.0600
11	89s 2s/step	28.516	0.1193	28.996	0.0667
12	96s 2s/step	27.945	0.1459	27.523	0.0733
13	103s 2s/step	26.589	0.1726	27.417	0.1667
14	93s 2s/step	25.971	0.1822	26.052	0.1600
15	86s 2s/step	25.734	0.1748	24.979	0.1600
16	102s 2s/step	24.318	0.2074	25.616	0.1867
17	95s 2s/step	23.983	0.2393	23.376	0.2400
18	94s 2s/step	23.152	0.2452	24.550	0.2067
19	94s 2s/step	22.497	0.2511	22.096	0.2800
20	95s 2s/step	21.600	0.2941	25.133	0.1800
21	93s 2s/step	21.116	0.3030	21.775	0.2733
22	88s 2s/step	20.459	0.3156	20.955	0.2800
23	104s 2s/step	19.751	0.3444	21.505	0.2933
24	95s 2s/step	19.556	0.3444	19.397	0.3267
25	93s 2s/step	18.744	0.3770	18.329	0.3933
26	89s 2s/step	17.826	0.4148	17.263	0.3467
27	93s 2s/step	16.837	0.4200	18.705	0.3533
28	93s 2s/step	16.198	0.4348	15.996	0.4067
29	101s 2s/step	16.128	0.4541	17.644	0.3933
30	88s 2s/step	15.508	0.4578	15.075	0.4400
31	102s 2s/step	14.516	0.4911	14.620	0.4867
32	95s 2s/step	14.275	0.4874	16.374	0.4400
33	96s 2s/step	13.631	0.5244	12.535	0.5600
34	94s 2s/step	13.046	0.5281	12.441	0.4733
35	94s 2s/step	12.954	0.5304	12.530	0.5600
36	94s 2s/step	13.447	0.5244	13.281	0.5067

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
37	96s 2s/step	12.351	0.5674	12.237	0.5533
38	89s 2s/step	12.087	0.5667	13.010	0.5133
39	100s 2s/step	11.756	0.5578	14.130	0.4933
40	94s 2s/step	11.245	0.6015	13.410	0.5533
41	95s 2s/step	11.554	0.5911	10.962	0.6133
42	94s 2s/step	0.9922	0.6511	11.173	0.6267
43	95s 2s/step	10.539	0.6237	0.9888	0.6400
44	94s 2s/step	0.9957	0.6563	10.116	0.6800
45	88s 2s/step	0.9982	0.6430	12.057	0.6267
46	99s 2s/step	10.203	0.6267	10.045	0.6133
47	94s 2s/step	0.9970	0.6422	12.042	0.6133
48	94s 2s/step	0.9525	0.6637	0.9506	0.7200
49	87s 2s/step	0.9440	0.6748	10.326	0.6133
50	101s 2s/step	0.9374	0.6593	10.386	0.6533
51	87s 2s/step	0.9406	0.6519	10.946	0.6200
52	101s 2s/step	0.8993	0.6881	10.868	0.6800
53	94s 2s/step	0.8655	0.6837	10.013	0.6533
54	94s 2s/step	0.8650	0.6674	0.8877	0.7067
55	95s 2s/step	0.8262	0.7148	0.7818	0.6867
56	93s 2s/step	0.8581	0.6985	0.8060	0.7200
57	94s 2s/step	0.7241	0.7333	0.8618	0.7200
58	87s 2s/step	0.7823	0.7230	0.9210	0.6533
59	102s 2s/step	0.7810	0.7274	0.7389	0.7733
60	94s 2s/step	0.7496	0.7370	0.9328	0.6533
61	95s 2s/step	0.7619	0.7422	0.7610	0.7467
62	94s 2s/step	0.7574	0.7319	0.8428	0.7533
63	95s 2s/step	0.6999	0.7607	0.7932	0.7067
64	88s 2s/step	0.7214	0.7252	0.6980	0.7533
65	94s 2s/step	0.7684	0.7341	0.7884	0.7333
66	97s 2s/step	0.7293	0.7459	0.8183	0.7133
67	102s 2s/step	0.6628	0.7696	0.5840	0.7867
68	93s 2s/step	0.6600	0.7681	0.6808	0.7267
69	89s 2s/step	0.6264	0.7830	0.6616	0.7733
70	100s 2s/step	0.6909	0.7622	0.6356	0.7867
71	94s 2s/step	0.6081	0.7859	0.6162	0.7867
72	93s 2s/step	0.6151	0.7807	0.6328	0.8000

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
73	94s 2s/step	0.6510	0.7659	0.7759	0.7933
74	95s 2s/step	0.6157	0.7830	0.7565	0.7400
75	94s 2s/step	0.6316	0.7830	0.5373	0.8533
76	94s 2s/step	0.6029	0.7867	0.6141	0.7800
77	93s 2s/step	0.6202	0.7778	0.7666	0.7267
78	94s 2s/step	0.6402	0.7904	0.5541	0.8067
79	93s 2s/step	0.6078	0.8022	0.5782	0.8133
80	94s 2s/step	0.5834	0.7881	0.5968	0.8200
81	87s 2s/step	0.5497	0.8104	0.5645	0.8400
82	100s 2s/step	0.5439	0.8126	0.4553	0.8800
83	95s 2s/step	0.6050	0.7889	0.5227	0.8133
84	93s 2s/step	0.5293	0.8193	0.6608	0.8533
85	87s 2s/step	0.5636	0.7978	0.4598	0.8600
86	101s 2s/step	0.5115	0.8281	0.4820	0.8667
87	94s 2s/step	0.5697	0.8096	0.4940	0.8267
88	95s 2s/step	0.5621	0.8089	0.5491	0.8533
89	94s 2s/step	0.4895	0.8341	0.4041	0.8933
90	88s 2s/step	0.4508	0.8378	0.5204	0.8600
91	93s 2s/step	0.4886	0.8296	0.5885	0.8267
92	98s 2s/step	0.4598	0.8415	0.7620	0.7733
93	98s 2s/step	0.4718	0.8415	0.4805	0.8267
94	94s 2s/step	0.4716	0.8400	0.6051	0.7867
95	94s 2s/step	0.4755	0.8281	0.4599	0.8467
96	95s 2s/step	0.4718	0.8363	0.4241	0.8733
97	93s 2s/step	0.4603	0.8415	0.3867	0.8533
98	87s 2s/step	0.4316	0.8600	0.5420	0.8533
99	101s 2s/step	0.4532	0.8437	0.6515	0.7867
100	86s 2s/step	0.4108	0.8607	0.5481	0.8267
101	96s 2s/step	0.4212	0.8519	0.3954	0.8667
102	99s 2s/step	0.4312	0.8504	0.6724	0.8333
103	89s 2s/step	0.3770	0.8674	0.2547	0.9200
104	98s 2s/step	0.4293	0.8407	0.4120	0.8400
105	94s 2s/step	0.4540	0.8437	0.5227	0.8667
106	94s 2s/step	0.4074	0.8615	0.4040	0.8933
107	93s 2s/step	0.4053	0.8563	0.4662	0.8467
108	95s 2s/step	0.3471	0.8785	0.4456	0.9200
109	94s 2s/step	0.3347	0.8867	0.5113	0.8533

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
110	95s 2s/step	0.3623	0.8674	0.5530	0.8333
111	94s 2s/step	0.3968	0.8607	0.5019	0.8800
112	95s 2s/step	0.3856	0.8607	0.3856	0.8533
113	87s 2s/step	0.3691	0.8859	0.6455	0.8467
114	101s 2s/step	0.3432	0.8785	0.4769	0.8800
115	95s 2s/step	0.3491	0.8770	0.4005	0.8733
116	93s 2s/step	0.3833	0.8726	0.5444	0.8733
117	88s 2s/step	0.3921	0.8681	0.4707	0.8667
118	100s 2s/step	0.3580	0.8704	0.4353	0.9200
119	89s 2s/step	0.3943	0.8659	0.4183	0.8867
120	98s 2s/step	0.3394	0.8919	0.6605	0.8533
121	93s 2s/step	0.3333	0.8852	0.4789	0.9000
122	94s 2s/step	0.3247	0.8822	0.3952	0.8800
123	94s 2s/step	0.3081	0.8963	0.5088	0.8533
124	96s 2s/step	0.3133	0.8859	0.3043	0.9200
125	96s 2s/step	0.3209	0.8881	0.3113	0.9133
126	95s 2s/step	0.2891	0.9089	0.5059	0.8533
127	94s 2s/step	0.3097	0.8933	0.4021	0.8800
128	87s 2s/step	0.3066	0.8948	0.3576	0.8800
129	101s 2s/step	0.2759	0.8985	0.3366	0.8867
130	87s 2s/step	0.3054	0.9052	0.3902	0.9000
131	101s 2s/step	0.2909	0.8993	0.3735	0.8933
132	94s 2s/step	0.2947	0.8993	0.4299	0.9133
133	94s 2s/step	0.2918	0.9000	0.4318	0.9000
134	88s 2s/step	0.2998	0.8926	0.6373	0.8733
135	101s 2s/step	0.3591	0.8793	0.3546	0.9133
136	93s 2s/step	0.3563	0.8763	0.3026	0.9467
137	89s 2s/step	0.3737	0.8800	0.4179	0.8800
138	99s 2s/step	0.3167	0.8985	0.4787	0.8867
139	93s 2s/step	0.2812	0.9052	0.2743	0.8933
140	95s 2s/step	0.3570	0.8756	0.3864	0.8933
141	93s 2s/step	0.2785	0.9119	0.4019	0.8800
142	89s 2s/step	0.2733	0.9015	0.4504	0.8733
143	100s 2s/step	0.2875	0.9044	0.4117	0.8933
144	95s 2s/step	0.2791	0.9096	0.2933	0.9200
145	93s 2s/step	0.2443	0.9178	0.4064	0.8933
146	93s 2s/step	0.2632	0.9119	0.3190	0.9067

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
147	95s 2s/step	0.2743	0.9022	0.3668	0.9267
148	94s 2s/step	0.2284	0.9304	0.3613	0.9067
149	95s 2s/step	0.3044	0.8896	0.4249	0.8800
150	86s 2s/step	0.2676	0.9185	0.2563	0.9333
151	102s 2s/step	0.2420	0.9163	0.4098	0.9067
152	88s 2s/step	0.2677	0.9089	0.4511	0.9067
153	101s 2s/step	0.2441	0.9237	0.2436	0.9200
154	94s 2s/step	0.2337	0.9133	0.4974	0.8867
155	94s 2s/step	0.2180	0.9156	0.4107	0.8867
156	93s 2s/step	0.2383	0.9259	0.5054	0.8733
157	94s 2s/step	0.2717	0.9163	0.2596	0.9200
158	95s 2s/step	0.2471	0.9207	0.4150	0.8867
159	94s 2s/step	0.2673	0.9015	0.3392	0.8933
160	95s 2s/step	0.2402	0.9215	0.3073	0.9333
161	93s 2s/step	0.2669	0.9052	0.5478	0.8800
162	89s 2s/step	0.3781	0.8793	0.3709	0.9467
163	99s 2s/step	0.2420	0.9156	0.3864	0.9067
164	94s 2s/step	0.2391	0.9230	0.3018	0.9200
165	94s 2s/step	0.2335	0.9156	0.4795	0.8667
166	94s 2s/step	0.2218	0.9244	0.4285	0.8667
167	95s 2s/step	0.2506	0.9267	0.5600	0.8600
168	93s 2s/step	0.2183	0.9200	0.4137	0.9333
169	95s 2s/step	0.2516	0.9207	0.2503	0.9267
170	93s 2s/step	0.2216	0.9222	0.3840	0.9000
171	94s 2s/step	0.3200	0.8919	0.3135	0.9267
172	94s 2s/step	0.1817	0.9400	0.2452	0.9733
173	93s 2s/step	0.1922	0.9393	0.4851	0.9200
174	94s 2s/step	0.1955	0.9415	0.4128	0.9200
175	94s 2s/step	0.2223	0.9222	0.2313	0.9267
176	94s 2s/step	0.1812	0.9348	0.2807	0.9067
177	87s 2s/step	0.1941	0.9356	0.3974	0.9200
178	91s 2s/step	0.2166	0.9289	0.2667	0.9067
179	102s 2s/step	0.1717	0.9400	0.4000	0.9067
180	94s 2s/step	0.2383	0.9156	0.3510	0.8933
181	94s 2s/step	0.1612	0.9415	0.3414	0.9200
182	93s 2s/step	0.2210	0.9259	0.2153	0.9267
183	86s 2s/step	0.1917	0.9385	0.3634	0.9467

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
184	96s 2s/step	0.2209	0.9274	0.2723	0.9200
185	99s 2s/step	0.1724	0.9430	0.3331	0.9267
186	95s 2s/step	0.1826	0.9437	0.4364	0.8933
187	93s 2s/step	0.2006	0.9385	0.3208	0.9467
188	95s 2s/step	0.1771	0.9430	0.3939	0.9133
189	93s 2s/step	0.1938	0.9289	0.3926	0.9333
190	86s 2s/step	0.1962	0.9326	0.3989	0.9133
191	102s 2s/step	0.2016	0.9289	0.3898	0.9133
192	94s 2s/step	0.1653	0.9467	0.3295	0.9000
193	89s 2s/step	0.1752	0.9407	0.4019	0.9067
194	94s 2s/step	0.1949	0.9370	0.6723	0.8867
195	94s 2s/step	0.2086	0.9244	0.4021	0.8933
196	100s 2s/step	0.2169	0.9356	0.2917	0.9333
197	90s 2s/step	0.2158	0.9326	0.2723	0.9200
198	99s 2s/step	0.1707	0.9370	0.3199	0.9267
199	94s 2s/step	0.1938	0.9459	0.4177	0.9267
200	86s 2s/step	0.1761	0.9393	0.5422	0.8867
201	100s 2s/step	0.1980	0.9341	0.4045	0.9000
202	95s 2s/step	0.2002	0.9267	0.4227	0.9200
203	87s 2s/step	0.1894	0.9363	0.3158	0.9333
204	101s 2s/step	0.1631	0.9452	0.3102	0.9333
205	94s 2s/step	0.1688	0.9444	0.1498	0.9600
206	93s 2s/step	0.1872	0.9393	0.3631	0.9267
207	95s 2s/step	0.1873	0.9422	0.3855	0.9267
208	94s 2s/step	0.1651	0.9444	0.2034	0.9467
209	88s 2s/step	0.1744	0.9459	0.2239	0.9400
210	99s 2s/step	0.1851	0.9378	0.5599	0.9267
211	94s 2s/step	0.1681	0.9467	0.4665	0.8800
212	88s 2s/step	0.1743	0.9407	0.4146	0.9067
213	99s 2s/step	0.1736	0.9407	0.6327	0.9000
214	94s 2s/step	0.1783	0.9400	0.2438	0.9000
215	93s 2s/step	0.1824	0.9363	0.2952	0.9333
216	94s 2s/step	0.1909	0.9393	0.3161	0.9133
217	93s 2s/step	0.1552	0.9496	0.2858	0.9333
218	94s 2s/step	0.1679	0.9437	0.5081	0.9067
219	95s 2s/step	0.1875	0.9400	0.2517	0.9267
220	94s 2s/step	0.1631	0.9459	0.3170	0.9267

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
221	87s 2s/step	0.1360	0.9548	0.2989	0.9467
222	100s 2s/step	0.1761	0.9422	0.3943	0.9267
223	94s 2s/step	0.1762	0.9459	0.1348	0.9533
224	94s 2s/step	0.1870	0.9407	0.4255	0.9000
225	94s 2s/step	0.1635	0.9519	0.3409	0.8933
226	95s 2s/step	0.2031	0.9333	0.2948	0.9467
227	94s 2s/step	0.1652	0.9504	0.3144	0.9200
228	88s 2s/step	0.1817	0.9422	0.3063	0.9333
229	100s 2s/step	0.1409	0.9541	0.2789	0.9200
230	87s 2s/step	0.1276	0.9600	0.2746	0.9400
231	103s 2s/step	0.1526	0.9533	0.3277	0.9200
232	89s 2s/step	0.1708	0.9496	0.3362	0.9467
233	101s 2s/step	0.1895	0.9341	0.3294	0.9400
234	93s 2s/step	0.1376	0.9578	0.4291	0.8933
235	95s 2s/step	0.1392	0.9481	0.8652	0.9200
236	93s 2s/step	0.1278	0.9533	0.5712	0.9000
237	90s 2s/step	0.1782	0.9363	0.3404	0.9200
238	91s 2s/step	0.1466	0.9437	0.2452	0.9733
239	92s 2s/step	0.1528	0.9526	0.2933	0.9467
240	104s 2s/step	0.1479	0.9504	0.3603	0.9133
241	94s 2s/step	0.1411	0.9481	0.2561	0.9600
242	95s 2s/step	0.1335	0.9563	0.2358	0.9467
243	94s 2s/step	0.1911	0.9415	0.2103	0.9467
244	85s 2s/step	0.1424	0.9511	0.4284	0.9333
245	101s 2s/step	0.1345	0.9600	0.3539	0.9533
246	96s 2s/step	0.1122	0.9644	0.2949	0.9267
247	93s 2s/step	0.1550	0.9496	0.3188	0.9400
248	94s 2s/step	0.0981	0.9667	0.4967	0.9333
249	89s 2s/step	0.1315	0.9600	0.0564	0.9800
250	99s 2s/step	0.1249	0.9615	0.5615	0.8800
251	94s 2s/step	0.1440	0.9556	0.2635	0.9267
252	87s 2s/step	0.2535	0.9222	0.3777	0.9000
253	102s 2s/step	0.1631	0.9430	0.3355	0.9333
254	87s 2s/step	0.1754	0.9348	0.2315	0.9533
255	101s 2s/step	0.2146	0.9222	0.3426	0.9267
256	94s 2s/step	0.1839	0.9415	0.3092	0.9200
257	93s 2s/step	0.1344	0.9600	0.4206	0.9333

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
258	94s 2s/step	0.1151	0.9600	0.2417	0.9600
259	94s 2s/step	0.1340	0.9593	0.7600	0.8733
260	95s 2s/step	0.1216	0.9526	0.3723	0.9200
261	93s 2s/step	0.1375	0.9511	0.2923	0.9067
262	89s 2s/step	0.1146	0.9570	0.4826	0.9333
263	95s 2s/step	0.1288	0.9578	0.2663	0.9333
264	93s 2s/step	0.1044	0.9681	0.3610	0.9400
265	102s 2s/step	0.1386	0.9541	0.3882	0.9467
266	94s 2s/step	0.1205	0.9622	0.2521	0.9333
267	95s 2s/step	0.1278	0.9630	0.4042	0.9333
268	94s 2s/step	0.1606	0.9422	0.1185	0.9600
269	96s 2s/step	0.1230	0.9622	0.2875	0.9400
270	94s 2s/step	0.1692	0.9459	0.2540	0.9133
271	95s 2s/step	0.1625	0.9444	0.2907	0.9267
272	94s 2s/step	0.1099	0.9585	0.2996	0.9600
273	95s 2s/step	0.1102	0.9630	0.4257	0.9133
274	88s 2s/step	0.1345	0.9570	0.4847	0.9267
275	101s 2s/step	0.1206	0.9615	0.1678	0.9533
276	94s 2s/step	0.1933	0.9400	0.4225	0.9267
277	94s 2s/step	0.0989	0.9652	0.2659	0.9333
278	94s 2s/step	0.1292	0.9585	0.3858	0.9400
279	94s 2s/step	0.1720	0.9437	0.3048	0.9200
280	89s 2s/step	0.1252	0.9556	0.3788	0.9200
281	100s 2s/step	0.1309	0.9578	0.3399	0.9533
282	96s 2s/step	0.1133	0.9607	0.2386	0.9400
283	93s 2s/step	0.0971	0.9696	0.3280	0.9467
284	87s 2s/step	0.1080	0.9659	0.2758	0.9467
285	103s 2s/step	0.0903	0.9667	0.2240	0.9600
286	95s 2s/step	0.1118	0.9644	0.3584	0.9333
287	94s 2s/step	0.1280	0.9600	0.3335	0.9600
288	87s 2s/step	0.1163	0.9615	0.3670	0.9333
289	102s 2s/step	0.1107	0.9689	0.2952	0.9467
290	95s 2s/step	0.1348	0.9526	0.4365	0.9067
291	95s 2s/step	0.1306	0.9548	0.3239	0.9200
292	94s 2s/step	0.1102	0.9652	0.3395	0.9267
293	95s 2s/step	0.1484	0.9467	0.5262	0.9000
294	94s 2s/step	0.1429	0.9541	0.3766	0.8933

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
295	96s 2s/step	0.1205	0.9563	0.2857	0.9267
296	93s 2s/step	0.1351	0.9541	0.4126	0.9133
297	89s 2s/step	0.1272	0.9615	0.3727	0.9133
298	93s 2s/step	0.0916	0.9674	0.3881	0.9533
299	101s 2s/step	0.1357	0.9615	0.1530	0.9533
300	94s 2s/step	0.1337	0.9556	0.5521	0.9067
301	88s 2s/step	0.1253	0.9607	0.2686	0.9467
302	100s 2s/step	0.1364	0.9593	0.4085	0.9333
303	93s 2s/step	0.1614	0.9481	0.3584	0.9067
304	95s 2s/step	0.1429	0.9533	0.2007	0.9400
305	93s 2s/step	0.1191	0.9593	0.3814	0.9267
306	93s 2s/step	0.1234	0.9593	0.6373	0.9333
307	95s 2s/step	0.1032	0.9644	0.3847	0.9333
308	93s 2s/step	0.0929	0.9652	0.5203	0.9400
309	94s 2s/step	0.1132	0.9607	0.2827	0.9467
310	86s 2s/step	0.1308	0.9630	0.2523	0.9400
311	102s 2s/step	0.1259	0.9607	0.2847	0.9400
312	94s 2s/step	0.1099	0.9630	0.2028	0.9467
313	94s 2s/step	0.0906	0.9756	0.4143	0.9267
314	95s 2s/step	0.1369	0.9593	0.5175	0.9200
315	95s 2s/step	0.0685	0.9756	0.3098	0.9333
316	94s 2s/step	0.0821	0.9741	0.4124	0.9400
317	89s 2s/step	0.0886	0.9719	0.2133	0.9667
318	96s 2s/step	0.0918	0.9674	0.4657	0.9333
319	96s 2s/step	0.0970	0.9704	0.3245	0.9400
320	101s 2s/step	0.1081	0.9689	0.3730	0.9333
321	94s 2s/step	0.0952	0.9652	0.2700	0.9467
322	95s 2s/step	0.1462	0.9459	0.5896	0.9333
323	87s 2s/step	0.1141	0.9689	0.2544	0.9467
324	95s 2s/step	0.1345	0.9533	0.4745	0.9467
325	100s 2s/step	0.1134	0.9733	0.4357	0.9267
326	95s 2s/step	0.0953	0.9659	0.3373	0.9533
327	93s 2s/step	0.0932	0.9704	0.2412	0.9267
328	96s 2s/step	0.1220	0.9622	0.4154	0.9267
329	94s 2s/step	0.1471	0.9556	0.2374	0.9467
330	85s 2s/step	0.0881	0.9719	0.6260	0.9267
331	102s 2s/step	0.0747	0.9748	0.3248	0.9267

Lampiran I. Lanjutan

Epoch	*Time*	*Loss*	*Accuracy*	*Validasi Loss*	*Validasi Accuracy*
332	93s 2s/step	0.1739	0.9444	0.7462	0.9067
333	95s 2s/step	0.1640	0.9519	0.5111	0.9333
334	94s 2s/step	0.1203	0.9585	0.3372	0.9200
335	88s 2s/step	0.1268	0.9600	0.4110	0.9333
336	100s 2s/step	0.1248	0.9578	0.3130	0.9200
337	95s 2s/step	0.1077	0.9637	0.2117	0.9667
338	86s 2s/step	0.1004	0.9644	0.3745	0.9600
339	102s 2s/step	0.1368	0.9607	0.3583	0.9067
340	94s 2s/step	0.1027	0.9689	0.2166	0.9533
341	87s 2s/step	0.0688	0.9807	0.4949	0.9200
342	101s 2s/step	0.1217	0.9570	0.4306	0.9200
343	94s 2s/step	0.0929	0.9652	0.3051	0.9467
344	95s 2s/step	0.1101	0.9659	0.2817	0.9400
345	93s 2s/step	0.0876	0.9733	0.4313	0.9200
346	89s 2s/step	0.0802	0.9659	0.3704	0.9533
347	99s 2s/step	0.1153	0.9615	0.3613	0.9333
348	93s 2s/step	0.1303	0.9533	0.3000	0.9667
349	96s 2s/step	0.1382	0.9593	0.5212	0.8933
350	93s 2s/step	0.1176	0.9622	0.2011	0.9600

Lampiran II

Hasil *Testing* Model

No.	Nama File	Data Asal	Prediksi Sistem	Hasil Prediksi	Prediksi
1	IMG20221118132907.jpg	da	da	BENAR	0.999976396560669
2	IMG20221118132932.jpg	da	da	BENAR	0.9996436834335327
3	IMG20221118132925.jpg	da	da	BENAR	0.9999693632125854
4	IMG20221118133312.jpg	da	da	BENAR	0.9990567564964294
5	IMG20221118132903.jpg	da	da	BENAR	0.9758199453353882
6	IMG20230412094141.jpg	ba	ba	BENAR	0.997661828994751
7	IMG20230412094108.jpg	ba	nga	SALAH	0.771274983882904
8	IMG20230412094121.jpg	ba	ba	BENAR	0.997627317905426
9	IMG20230412094101.jpg	ba	ha	SALAH	0.935520589351654
10	IMG20230412094104.jpg	ba	nga	SALAH	0.6011412739753723
11	IMG20221117203401.jpg	ha	ha	BENAR	0.7049440741539001
12	IMG20221117203351.jpg	ha	nya	SALAH	0.6782740354537964
13	IMG20221117203408.jpg	ha	ha	BENAR	0.9969947338104248

Lampiran II. Lanjutan

No.	Nama File	Data Asal	Prediksi Sistem	Hasil Prediksi	Prediksi
14	IMG20221117203315.jpg	ha	ha	BENAR	0.999049723148346
15	IMG20221117203322.jpg	ha	ha	BENAR	0.9613637328147888
16	IMG20221123191154.jpg	ja	ja	BENAR	0.9999923706054688
17	IMG20221123191151.jpg	ja	ja	BENAR	0.9999871253967285
18	IMG20221123191140.jpg	ja	ja	BENAR	0.9999916553497314
19	IMG20221123191134.jpg	ja	ja	BENAR	0.9999985694885254
20	IMG20221123191144.jpg	ja	ja	BENAR	0.9999768733978271
21	IMG20221118145633.jpg	ca	ca	BENAR	0.9998979568481445
22	IMG20221118145645.jpg	ca	ca	BENAR	0.9999872446060181
23	IMG20221118145630.jpg	ca	ca	BENAR	0.9692810773849487
24	IMG20221118145638.jpg	ca	ca	BENAR	0.9909063577651978
25	IMG20221118145642.jpg	ca	ca	BENAR	0.9998843669891357
26	IMG20221123185212.jpg	la	ha	SALAH	0.812364935874939
27	IMG20221123185222.jpg	la	la	BENAR	0.7512533068656921
28	IMG20221123185217.jpg	la	ha	SALAH	0.712183952331543
29	IMG20221123185235.jpg	la	la	BENAR	0.6772825717926025
30	IMG20221123185208.jpg	la	la	BENAR	0.721623420715332
31	IMG20230412092853.jpg	ma	ca	SALAH	0.711208164691925
32	IMG20230412092900.jpg	ma	ma	BENAR	0.9999836683273315
33	IMG20230412092904.jpg	ma	ma	BENAR	0.9941987991333008
34	IMG20230412092855.jpg	ma	ma	BENAR	0.9999575614929199
35	IMG20230412092849.jpg	ma	ma	BENAR	0.999932050704956
36	IMG20230412093418.jpg	ga	ga	BENAR	0.9999983310699463
37	IMG20230412093422.jpg	ga	ga	BENAR	0.9999752044677734
38	IMG20230412093407.jpg	ga	ga	BENAR	0.9999184608459473
39	IMG20230412093409.jpg	ga	ga	BENAR	0.75059574842453
40	IMG20230412093416.jpg	ga	ga	BENAR	0.9999856948852539
41	IMG20221118132108.jpg	ka	ka	BENAR	0.9983835220336914
42	IMG20221118132158.jpg	ka	ka	BENAR	0.9999171495437622
43	IMG20221118132049.jpg	ka	ka	BENAR	0.9998376369476318
44	IMG20221118132041.jpg	ka	ka	BENAR	0.9981725215911865
45	IMG20221118132037.jpg	ka	ka	BENAR	0.9999518394470215
46	IMG20221123190822.jpg	dha	dha	BENAR	0.9996649026870728
47	IMG20221123190826.jpg	dha	dha	BENAR	0.9999873638153076
48	IMG20221123190815.jpg	dha	dha	BENAR	0.9999983310699463
49	IMG20221123190811.jpg	dha	dha	BENAR	0.9999436140060425
50	IMG20221123190820.jpg	dha	dha	BENAR	0.9999935626983643
51	IMG20221123190214.jpg	pa	pa	BENAR	0.9927592277526855

Lampiran II. Lanjutan

No.	Nama File	Data Asal	Prediksi Sistem	Hasil Prediksi	Prediksi
52	IMG20221123190211.jpg	pa	pa	BENAR	0.9801946878433228
53	IMG20221123190223.jpg	pa	pa	BENAR	0.9886119961738586
54	IMG20221123190226.jpg	pa	pa	BENAR	0.9825427532196045
55	IMG20221123190242.jpg	pa	pa	BENAR	0.9797820448875427
56	IMG20221117233035.jpg	ra	ra	BENAR	01.00
57	IMG20221117233115.jpg	ra	ra	BENAR	0.9999903440475464
58	IMG20221117233121.jpg	ra	ra	BENAR	0.9999984502792358
59	IMG20221117233118.jpg	ra	ra	BENAR	0.9999945163726807
60	IMG20221117233125.jpg	ra	ra	BENAR	0.9958786964416504
61	IMG20221123191726.jpg	ya	ya	BENAR	0.9999839067459106
62	IMG20221123191719.jpg	ya	ya	BENAR	0.9999963045120239
63	IMG20221123191723.jpg	ya	ya	BENAR	01.00
64	IMG20221123191716.jpg	ya	ya	BENAR	0.9999704360961914
65	IMG20221123191710.jpg	ya	ya	BENAR	0.9999969005584717
66	IMG20230412095253.jpg	nga	nga	BENAR	0.9999991655349731
67	IMG20230412095259.jpg	nga	nga	BENAR	0.9998886585235596
68	IMG20230412095240.jpg	nga	nga	BENAR	0.9999421834945679
69	IMG20230412095249.jpg	nga	nga	BENAR	0.9998657703399658
70	IMG20230412095245.jpg	nga	nga	BENAR	0.9999996423721313
71	IMG20230412094513.jpg	tha	tha	BENAR	0.9999781847000122
72	IMG20230412094515.jpg	tha	tha	BENAR	0.9998887777328491
73	IMG20230412094502.jpg	tha	tha	BENAR	0.9999760389328003
74	IMG20230412094457.jpg	tha	tha	BENAR	0.9999628067016602
75	IMG20230412094504.jpg	tha	tha	BENAR	0.9999853372573853
76	IMG20221118150040.jpg	wa	wa	BENAR	0.9993496537208557
77	IMG20221118150050.jpg	wa	wa	BENAR	0.9988742470741272
78	IMG20221118150057.jpg	wa	wa	BENAR	0.9956687688827515
79	IMG20221118150045.jpg	wa	wa	BENAR	0.9974915981292725
80	IMG20221118150101.jpg	wa	wa	BENAR	0.9916862845420837
81	IMG_20221117_212900.jpg	na	na	BENAR	0.5194545388221741
82	IMG20221117212206.jpg	na	na	BENAR	0.9539977312088013
83	IMG20221117212241.jpg	na	na	BENAR	0.5130056738853455
84	IMG20221117212229.jpg	na	da	SALAH	0.6944449543952942
85	IMG20221117212237.jpg	na	da	SALAH	0.8253216743469238
86	IMG20230412092416.jpg	nya	nya	BENAR	0.9999470710754395
87	IMG20230412092424.jpg	nya	nya	BENAR	0.9996178150177002
88	IMG20230412092412.jpg	nya	nya	BENAR	0.9996756315231323
89	IMG20230412092439.jpg	nya	nya	BENAR	0.9645365476608276

Lampiran II. Lanjutan

No.	Nama File	Data Asal	Prediksi Sistem	Hasil Prediksi	Prediksi
90	IMG20230412092420.jpg	nya	nya	BENAR	0.999854564666748
91	IMG20221118133701.jpg	ta	ta	BENAR	0.9999995231628418
92	IMG20221118133652.jpg	ta	ta	BENAR	0.979332447052002
93	IMG20221118133656.jpg	ta	ta	BENAR	0.9999760389328003
94	IMG20221118133710.jpg	ta	ta	BENAR	0.999995231628418
95	IMG20221118133716.jpg	ta	ta	BENAR	01.00
96	IMG20221117232141.jpg	sa	na	SALAH	0.5345481038093567
97	IMG20221117232248.jpg	sa	sa	BENAR	0.9977908134460449
98	IMG20221117232128.jpg	sa	na	SALAH	0.5620614886283875
99	IMG20221117232256.jpg	sa	sa	BENAR	0.9998921155929565
100	IMG20221117232041.jpg	sa	na	SALAH	0.9260716438293457

Lampiran III

Hasil Klasifikasi Pengujian

No	Data Asal	Prediksi Sistem	TP	TN	FP	FN
1	da	da	1	19	0	0
2	da	da	1	19	0	0
3	da	da	1	19	0	0
4	da	da	1	19	0	0
5	da	da	1	19	0	0
6	ba	ba	1	19	0	0
7	ba	nga	0	18	1	1
8	ba	ba	1	19	0	0
9	ba	ha	0	18	1	1
10	ba	nga	0	18	1	1
11	ha	ha	1	19	0	0
12	ha	nya	0	18	1	1
13	ha	ha	1	19	0	0
14	ha	ha	1	19	0	0
15	ha	ha	1	19	0	0
16	ja	ja	1	19	0	0
17	ja	ja	1	19	0	0
18	ja	ja	1	19	0	0
19	ja	ja	1	19	0	0
20	ja	ja	1	19	0	0
21	ca	ca	1	19	0	0
22	ca	ca	1	19	0	0

Lampiran III. Lanjutan

No	Data Asal	Prediksi Sistem	TP	TN	FP	FN
23	ca	ca	1	19	0	0
24	ca	ca	1	19	0	0
25	ca	ca	1	19	0	0
26	la	ha	0	18	1	1
27	la	la	1	19	0	0
28	la	ha	0	18	1	1
29	la	la	1	19	0	0
30	la	la	1	19	0	0
31	ma	ca	0	18	1	1
32	ma	ma	1	19	0	0
33	ma	ma	1	19	0	0
34	ma	ma	1	19	0	0
35	ma	ma	1	19	0	0
36	ga	ga	1	19	0	0
37	ga	ga	1	19	0	0
38	ga	ga	1	19	0	0
39	ga	ga	1	19	0	0
40	ga	ga	1	19	0	0
41	ka	ka	1	19	0	0
42	ka	ka	1	19	0	0
43	ka	ka	1	19	0	0
44	ka	ka	1	19	0	0
45	ka	ka	1	19	0	0
46	dha	dha	1	19	0	0
47	dha	dha	1	19	0	0
48	dha	dha	1	19	0	0
49	dha	dha	1	19	0	0
50	dha	dha	1	19	0	0
51	pa	pa	1	19	0	0
52	pa	pa	1	19	0	0
53	pa	pa	1	19	0	0
54	pa	pa	1	19	0	0
55	pa	pa	1	19	0	0
56	ra	ra	1	19	0	0
57	ra	ra	1	19	0	0
58	ra	ra	1	19	0	0
59	ra	ra	1	19	0	0
60	ra	ra	1	19	0	0
61	ya	ya	1	19	0	0

Lampiran III. Lanjutan

No	Data Asal	Prediksi Sistem	TP	TN	FP	FN
62	ya	ya	1	19	0	0
63	ya	ya	1	19	0	0
64	ya	ya	1	19	0	0
65	ya	ya	1	19	0	0
66	nga	nga	1	19	0	0
67	nga	nga	1	19	0	0
68	nga	nga	1	19	0	0
69	nga	nga	1	19	0	0
70	nga	nga	1	19	0	0
71	tha	tha	1	19	0	0
72	tha	tha	1	19	0	0
73	tha	tha	1	19	0	0
74	tha	tha	1	19	0	0
75	tha	tha	1	19	0	0
76	wa	wa	1	19	0	0
77	wa	wa	1	19	0	0
78	wa	wa	1	19	0	0
79	wa	wa	1	19	0	0
80	wa	wa	1	19	0	0
81	na	na	1	19	0	0
82	na	na	1	19	0	0
83	na	na	1	19	0	0
84	na	da	0	18	1	1
85	na	da	0	18	1	1
86	nya	nya	1	19	0	0
87	nya	nya	1	19	0	0
88	nya	nya	1	19	0	0
89	nya	nya	1	19	0	0
90	nya	nya	1	19	0	0
91	ta	ta	1	19	0	0
92	ta	ta	1	19	0	0
93	ta	ta	1	19	0	0
94	ta	ta	1	19	0	0
95	ta	ta	1	19	0	0
96	sa	na	0	18	1	1
97	sa	sa	1	19	0	0
98	sa	na	0	18	1	1
99	sa	sa	1	19	0	0
100	sa	na	0	18	1	1

Lampiran III. Lanjutan

No	Data Asal	Prediksi Sistem	TP	TN	FP	FN
Total			88	1888	12	12