DECODE: Jurnal Pendidikan Teknologi Informasi
ISSN Cetak : 2775-2984 | ISSN Online : 2775-1813
Vol. 3, No. 2, September 2023, Hal: 161-171, Doi: http://dx.doi.org/10.51454/decode.v3i2.131
http://journal.umkendari.ac.id/index.php/decode

PEMBENTUKAN MODEL RECURRENT NEURAL NETWORK DAN CONNECTIONIST TEMPORAL CLASSIFICATION PADA PENGENALAN KATA TULISAN TANGAN OFFLINE

Diana Tri Susetianingtias[1)], Rini Arianty[1)], Rodiah[1)], Eka Patriya[1)*]
[1] Universitas Gunadarma, Depok, Indonesia
Email: ekapatriya@staff.gunadarma.ac.id

Abstrak

Pengolahan informasi sebagai bentuk dari pengembangan teknologi yang semakin pesat adalah pengenalan tulisan tangan. Pengenalan tulisan tangan saat ini banyak diimplementasikan untuk melakukan identifikasi dokumen-dokumen penting berbentuk digital. Pengenalan tulisan tangan cetak dengan benar dan akurat dapat digunakan untuk menunjang kegiatan manusia baik dalam kegiatan sehari – hari, sekolah, maupun pekerjaan. Pada penelitian ini, peneliti akan melakukan pengenalan tulisan tangan cetak menggunakan *Convolutional Neural Network* pada perangkat *mobile.* Penelitian juga akan mengimplementasikan model CNN yang dihasilkan pada android dan mengidentifikasi tulisan tangan yang diambil secara langsung menggunakan kamera. Proses pelatihan menggunakan *library* tensorflow pada bahasa pemrograman python, dilakukan secara *online*. Model CNN berhasil dibentuk dengan jumlah 11 *hidden layer*, terdiri dari 5 lapisan konvolusi, 3 lapisan *max pooling*, 1 lapisan *flatten*, dan 2 lapisan *fully connected*. Model mengambil masukan berupa citra berukuran 28 x 28 piksel dan menghasilkan keluaran berupa pengenalan tulisan tangan cetak. Total parameter bobot yang dimiliki oleh model berjumlah 1.556.495 variabel. Model CNN yang sudah dilatih berhasil diimplementasikan dan dijalankan pada aplikasi android. Berdasarkan perhitungan *Global Performance Metric*, Aplikasi berhasil mengenali citra tulisan tangan 1 kata dengan akurasi 82,12%. Hasil penelitian diharapkan dapat mengenali tulisan tangan cetak dengan akurasi tinggi pada perangkat android.

Kata kunci: bobot; CNN; *global performance metric*; parameter; tulisan tangan.

DEVELOPMENT OF RECURRENT NEURAL NETWORK AND CONNECTIONIST TEMPORAL CLASSIFICATION MODEL FOR OFFLINE WORD HANDWRITING RECOGNITION

Abstract

Information processing as a form of technology development that is increasingly rapidly is handwriting recognition. Handwriting recognition is currently widely implemented to identify important documents in digital form. Recognition of printed handwriting correctly and accurately can be used to support human activities both in daily activities, school and work. In this study, researchers will perform handwriting recognition using a Convolutional Neural Network on a mobile device. The research will also implement the CNN model generated on Android and identify handwriting taken directly using the camera. All handwritten images used as input are grayscale images stored in .png format measuring 28x28. The training process uses the tensorflow library in the Python programming language, carried out online. The CNN model was successfully formed with a total of 11 hidden layers, consisting of 5 convolution layers, 3 max pooling layers, 1 flatten layer, and 2 fully connected layers. The model takes input in the form of an image measuring 28 x 28 pixels and produces output in the form of handwriting recognition. The total weight parameters owned by the model are 1,556,495 variables. The CNN model that has been trained has been successfully implemented and run on an android application. Based on Global Performance Metric calculations, the application successfully recognizes a handwritten image of 1 word

with an accuracy of 82.12%. The results of the study are expected to be able to recognize printed handwriting with high accuracy on android devices.

***Keywords**: weight; CNN; global performance metric; parameter; handwriting.*

Submitted: 27 Februari 2023	Reviewed: 28 Februari 2023	Accepted: 4 Mei 2023	Published: 5 Mei 2023

PENDAHULUAN

Pengenalan tulisan tangan (*handwriting recognition*), merupakan pengembangan dari kecerdasan buatan (Kayumov et al., 2020) yang merupakan kemajuan teknologi yang dapat mengenali sebuah tulisan atau teks layaknya seperti manusia. Saat ini, banyak individu atau kelompok yang ingin melakukan pendeteksian tulisan tangan sambung sambung (Fitrianingsih et al., 2017) dengan tujuan untuk mengubah informasi pada arsip atau dokumen dalam bentuk tulisan tangan sambung kedalam bentuk digital (Darmatasia & Fanany, 2017) dengan bantuan deteksi citra. Kesulitan dalam melakukan pengenalan tulisan tangan adalah bentuk,pola serta karakteristik tulisan (Bhunia et al., 2019) yang berbeda dari setiap individu. Pola tulisan tangan ini merupakan entitas yang didefinisikan melalui fitur ciri dalam membentuk tulisan tangan (Ji & Chen, 2019). Terdapat dua cara (Karthi et al., 2020) untuk mendeteksi tulisan tangan sambung yaitu pendeteksian secara *online* dan pendeteksian secara *offline*. Pendeteksian secara *online* dilakukan dengan cara mendeteksi tekanan (Ismael, 2020), ketebalan pada sebuah permukaan digital dengan menggunakan alat tulis atau pulpen khusus (Das et al., 2022) sedangkan pendeteksian tulisan tangan sambung secara *offline* dilakukan hanya dengan gambar biasa. Pendeteksian tulisan tangan sambung secara *offline* dapat dilakukan dengan pendeteksian citra dengan melakukan segmentasi kalimat menjadi kata (Aqab & Tariq, 2020) kemudian dilakukan pengenalan tulisan tangan.

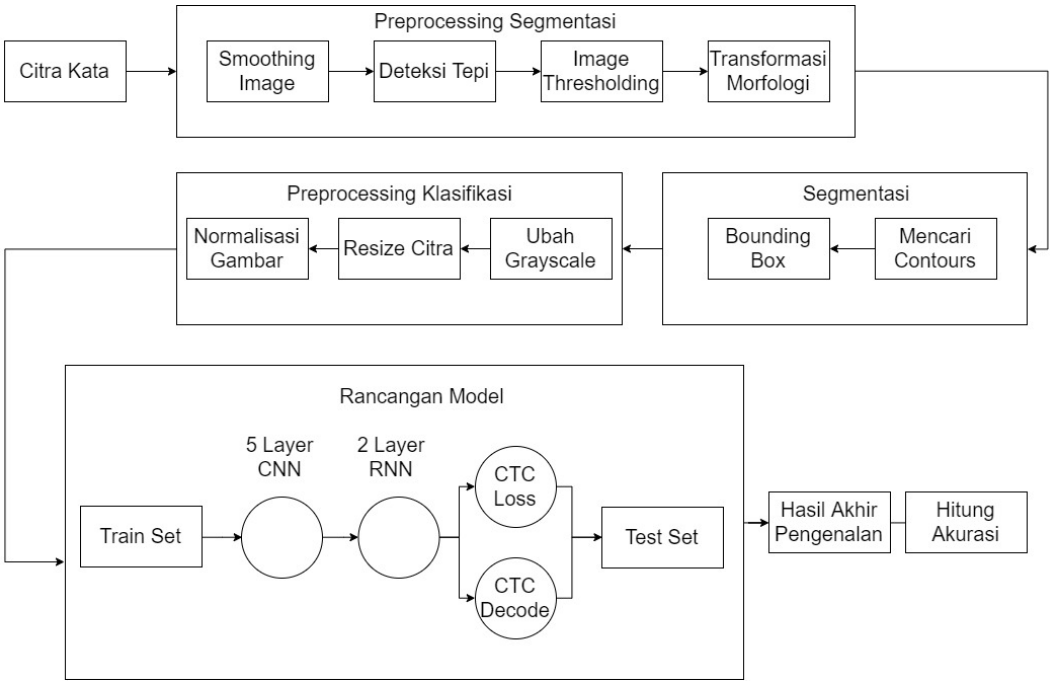
Penelitian dilakukan (Advaith et al., 2020) menggunakan LSTM (*Long Short-Term Memory)* dengan dua buah *layer Recurrent Neural Network* dan diikuti oleh penggunaan *Connectionist Temporan Network* atau *Sequence-to-Sequence Learning* dimana hasil akurasi yang didapatkan CTC lebih besar daripada Seq2Seq. Penelitian (Chammas et al., 2018) menggunakan *Convolutional Recurrent Neural Network* (CRNN) untuk dataset yang memilki sedikit label dengan melakukan normalisasi terhadap dataset sehingga menghasilkan akurasi yang lebih tinggi. Penelitian (Dwivedi et al., 2017) melakukan identifikasi tulisan tangan sambung *offline* dengan *gradient descent back propagation* menggunakan *dataset* yang bersumber dari *scanner*, kamera digital atau sumber digital *input device* lainnya. Selain *preprocessing* citra, juga dilakukan segmentasi untuk melabeli citra tulisan tangan sambung dengan ukuran 90x60 piksel. Penelitian ini melakukan ekstraksi fitur diagonal dimana setiap karakter dari gambar 90x60 pixel dibagi kedalam 54 zona yang masing-masing berukuran 10x10 piksel, tahap *classification and recognition* menggunakan *feed forward back propagation neural network* dengan 2 *hidden layer* yang digunakan untuk klasifikasi. Penelitian (Mohsin & Sadoon, 2020) mengenali tulisan tangan sambung berbahasa arab secara *offline* dengan melakukan ekstraksi karakter tulisan arab. Vektor ini kemudian digunakan di klasifier untuk mengenali pola bahasa arab. Penelitian ini menggunakan *feed forward neural network* dan berhasil mencapai akurasi 83% untuk semua karakter dan 96% untuk beberapa karakter.

Pengenalan tulisan tangan memiliki kendala dalam proses pengenalan dikarenakan kesamaan bentuk (Samant, 2021) dan pola pada beberapa huruf alphabet, dimana hal ini dapat mempengaruhi akurasi dalam pengenalan tulisan tangan. Pada penelitian ini, kendala dalam melakukan pengenalan tulisan tangan akibat kesamaan bentuk dan pola huruf diatasi dengan mengimplementasikan penggunaan model CNN untuk ekstraksi fitur ciri huruf, RNN yang digunakan dalam prediksi label pada setiap huruf yang akan dikenali, dan *Connectionist*

Temporal Classification untuk melakukan translasi prediksi pada setiap frame karakter huruf yang akan dikenali menjadi hasil akhir. Penggunaan metode pada penelitian ini akan melakukan ekstraksi dengan menentukan hubungan ketetanggaan antar pikel ada setiap huruf yang dikenali. Proses penentuan ketetanggaan mengikuti aturan ketetanggan antar pixel, yang akan membagi citra huruf kedalam beberapa sub-area untuk dilakukan ekstraksi. Penelitian ini menggunakan IAM dataset sejumlah 115.250 citra kata huruf alfabet dengan ukuran citra yang bervariatif. *Training* dan *Testing* menggunakan rasio 8:2. Hasil penelitian dengan mengimplementasikan penggunaan model RNN dan Connectionist Temporal Classification diharapkan dapat mengenali tulisan tangan sambung *offline* dengan mengatasi kesulitan pengenalan pada kesamaan bentuk dan pola huruf untuk dapat digunakan pada sistem pengenalan tulisan tangan dengan akurasi yang tinggi.

METODE

Karakter atau huruf-huruf biasanya memiliki bentuk dasar tertentu (Memon et al., 2020). Terdapat aturan untuk mengkombinasikan huruf-huruf yang digunakan untuk merepresentasikan bentuk dari level tertinggi pada unit lingustik. Terdapat aturan untuk mengkombinasikan bentuk dari huruf-huruf sehingga membentuk kata-kata yang mempunyai arti dalam abjad latin (Impedovo et al., 2012). Metode penelitian yang dilakukan dalam melakukan pengenalan tulisan tangan sambung cetak pada penelitian terdiri atas beberapa tahapan proses seperti yang dapat dilihat pada Gambar 1.



Gambar 1. Metode Penelitian Pengenalan Tulisan Tangan Sambung Cetak

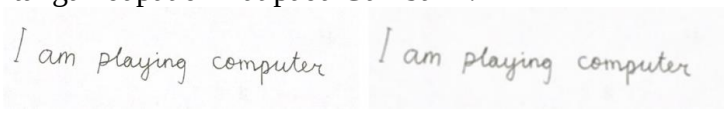
Penelitian ini menggunakan IAM Dataset (Bunke, 2002) sejumlah 115250 citra kata huruf alfabet dengan ukuran citra yang bervariatif dan berformat .png berbahasa inggris. Tahap awal *preprocessing* dilakukan segmentasi dengan langkah-langkah sebagai berikut:

1. Mengimplementasikan penggunaan *Gaussian Filtering* (Ismael, 2020) dengan ukuran kernel 15x1. Proses ini dilakukan untuk mengeliminasi informasi diluar citra kata yang

tidak dibutuhkan pada proses selanjutnya, dimana pada citra tulisan seringkali terdapat objek diluar objek tulisan (noise) yang dapat mempengaruhi hasil ekstraksi huruf. Eliminasi *noise* pada citra kata dilakukan menggunakan *Gaussian Smoothing* dengan bobot pada mask *Gaussian Smoothing* mengikuti distribusi normal menggunakan persamaan (Li et al., 2021) :

$$f_0(x,y) = \frac{f^r(x,y)+f^g(x,y)+f^b(x,y)}{3} \qquad (1)$$

Penerapan Metode *Gaussian Smoothing* antara citra f(x,y) dengan penapis $f^r(x,y) + f^g(x,y) + f^b(x,y)$. Penelusuran piksel dilakukan dari kiri ke kanan secara horisontal yang merupakan piksel *x*. Penelusuran berikutnya dilakukan dari atas ke bawah secara vertikal yang merupakan piksel *y*. Nilai indeks terkecil dari piksel *x* adalah 0 dan indeks terbesar dari piksel *x* adalah (lebar citra-1). Nilai Indeks terkecil dari piksel *y* adalah 0 dan index terbesar dari piksel *y* adalah (tinggi citra – 1). Proses eliminasi noise pada citra tulisan tangan dapat dilihat pada Gambar 1.



(a) (b)

Gambar 2 (a). Citra Asli (b). Citra Hasil Proses *Gaussian Smoothing*

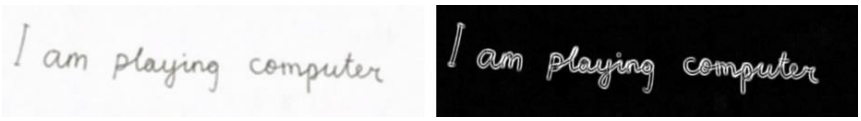
2. Setelah dilakukan eliminasi informasi untuk objek diluar citra tulisan, melakukan penelusuran piksel pada tepi vertikal dan horizontal berdasarkan bentuk dan pola huruf menggunakan kernel operator sobel (Gonzalez & Woods, n.d.)berukuran 3x3 piksel, dimana :

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \text{dan } G_y = \begin{bmatrix} -1 & 0 & 1 \\ 2 & 0 & 2 \\ -1 & 2 & 1 \end{bmatrix} \qquad (2)$$

Proses deteksi tepi dengan menggunakan operator sobel dilakukan untuk menelusuri ketetanggaan piksel pada setiap huruf untuk ketepatan dalam melakukan ekstraksi sehingga huruf dengan kesamaan bentuk dapat dikenali sebagai huruf yang berbeda. Proses deteksi tepi pada setiap layer dapat dilihat pada *pseudocode* berikut :

```
import numpy as np
import matplotlib.pyplot as plt
import cv2
from helpers import *
def edge_detect(blurred):
   return np.max(np.array([_sobel_detect(blurred[:,:, 0]),
                           _sobel_detect(blurred[:,:, 1]),
                           _sobel_detect(blurred[:,:, 2])]), axis=0)
def _sobel_detect(channel):
   """Sobel operator."""
   sobelX = cv2.Sobel(channel, cv2.CV_16S, 1, 0)
   sobelY = cv2.Sobel(channel, cv2.CV_16S, 0, 1)
   sobel = np.hypot(sobelX, sobelY)
   sobel[sobel > 255] = 255
   return np.uint8(sobel)
```

Ilustrasi proses pendeteksian tepi menggunakan operator sobel dari citra tulisan tangan hasil proses *Gaussian Smoothing* dapat dilihat pada Gambar 3.



(a) (b)

Gambar 3 (a). Citra Hasil Proses *Gaussian Smoothing*
(b). Citra Hasil Proses Pendeteksian Tepi

3. Langkah selanjutnya setelah melakukan penelusuran ketetanggaan piksel objek huruf adalah melakukan konversi citra biner untuk memisahkan objek citra tulisan (i) dari latar belakangnya (*background*) dengan mengetahui level intensitas dari *foreground* dan background, dimana jika sebuah citra biner adalah B[i,j] sama dengan sebuah *thresholded gray image* FT[i,j] yang didapat dengan cara *threshold* T untuk *original gray image* F[i,j] (Madenda & Widodo, 2016) maka:

$$B[i,j] = FT[i,j] \quad (3)$$

Objek citra tulisan yang lebih gelap dan *background* yang lebih terang menggunakan persamaan (Samsuryadi et al., 2021) :

$$FT[i,j] = \{\ 1\ if\ F[i,j] \leq T\ 0\ otherwise \quad (4)$$

Jika diketahui nilai intensitas objek citra tulisan berada didalam *range* [T1, T2] digunakan persamaan (Gonzalez & Woods, n.d.) :

$$FT[i,j] = \{\ 1\ if\ T1 \leq F[i,j] \leq T2\ 0\ otherwise \quad (5)$$

Proses ini dilakukan pada setiap piksel citra tulisan tangan menghasilkan citra seperti dapat dilihat pada Gambar 4.

(a) (b)

Gambar 4 (a). Citra Hasil Proses Pendeteksian Tepi
(b). Citra Hasil Proses *Image Smoothing*

4. Melakukan proses *Bounding box* untuk menempatkan sebuah kotak imajiner yang berada disekeliling objek citra tulisan dengan membuat kotak disekeliling objek citra tulisan untuk memotong kata agar objek citra kata tersebut memiliki area yang tidak melebar. Implementasi *bounding box* digunakan fungsi pada *OpenCV*, nama dari fungsi tersebut adalah *cv.boundingRect()* menggunakan *pseudocode* :

```
boundingBoxes = [cv.boundingRect(c) for c in contours]
```

5. Melakukan proses kontur, proses ini dilakukan untuk memisahkan setiap huruf sebelum dimasukkan kedalam model pengenalan. Implementasi kontur menggunakan fungsi pada *OpenCV* sebagai berikut :

```
contours, _ = cv.findContours(
            final_thr,
            cv.RETR_EXTERNAL,
            cv.CHAIN_APPROX_SIMPLE )
boundingBoxes = [cv.boundingRect(c) for c in contours]
        (contours, boundingBoxes) = zip( *sorted(zip(contours, boundingBoxes),
key=lambda b: b[1][0], reverse=False))
for cnt in contours:
            area = cv.contourArea(cnt)
            if area > 100000:
               print('conuntours--------', cnt)
               x,y,w,h = cv.boundingRect(cnt)
               print (x,y,w,h)
               letterBgr = txt_line[0:txt_line.shape[1],x:x+w]
               wordImgList.append(letterBgr)
               words_result_path = "./result/words/{}.jpg".format(self.words_count)
               cv.imwrite(words_result_path,letterBgr)
               self.words_count += 1
         cl=cl+1
```

Pencarian kontur pada citra tulisan tangan sambung menggunakan dengan tiga parameter. Parameter pertama berupa citra tulisan tangan sambung, parameter kedua merupakan jenis dari pengambilan kontur yaitu *cv.RETR_EXTERNAL*, dan parameter terakhir digunakan untuk menentukan point dari kontur. Setelah kontur ditemukan selanjutnya mendeklarasikan *bounding box* untuk membungkus kontur yang ditemukan pada citra tulisan tangan sambung, setelah itu dilakukan sortir terhadap kontur terdeteksi dan juga *bounding box*. Proses selanjutnya adalah melakukan perulangan terhadap kontur dan mencari area dari kontur pada citra tulisan tangan sambung dengan ketentuan jika area lebih dari 100000 maka diberikan *bounding box* untuk membungkus kontur, setelah itu dilakukan penyimpanan citra tulisan tangan sambung pada folder *words*.

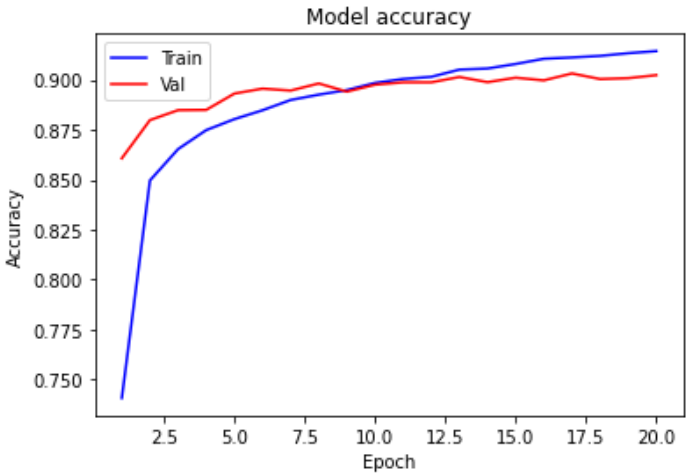
HASIL DAN PEMBAHASAN

Ujicoba dan evaluasi dilakukan pada citra tulisan tangan yang diakuisisi melalui perangkat kamera. *Split data* dilakukan untuk mengoptimalisasi proses pelatihan model CNN, dimana citra tulisan tangan yang terdapat didalam *dataset* dengan jumlah 112800 citra dibagi menjadi dua bagian, yaitu *training set* dan *validation set* dengan rasio 8:2. Pelatihan dilakukan sebanyak 20 *epoch* dengan peneliti mencatat performa pelatihan pada tiap *epoch*. Tabel 1 menunjukkan hasil pelatihan tiap 5 *epoch* beserta dengan *epoch* terbaik yang diambil dengan total kurun waktu sekitar 10 menit.

Tabel 1. Hasil Pelatihan Model CNN

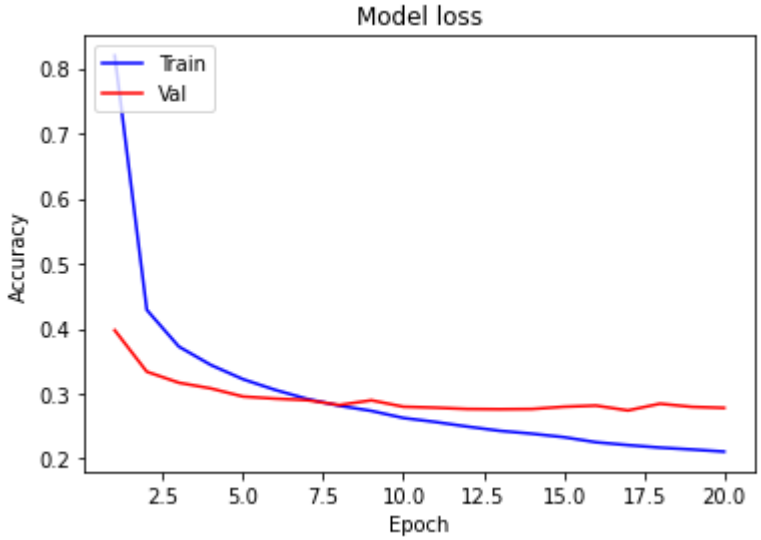
Iterasi	Kesalahan Pelatihan	Akurasi Pelatihan	Kesalahan Validasi	Akurasi Validasi
10	0.2630	0.8974	0.2819	0.8977
20	0.2110	0.9143	0.2782	0.9023
30	0.1725	0.9268	0.3027	0.9015
40	0.1524	0.9350	0.3069	0.9031
50	0.1361	0.9421	0.3395	0.9020

Berdasarkan Tabel 1 didapat bahwa hasil pelatihan pada *epoch* ke 20 merupakan hasil terbaik dari keseluruhan pelatihan yang sudah dilakukan. Hasil tersebut diambil dengan melihat akurasi terbaik dari *validation set*, yaitu mencapai angka 90.23%, dimana model CNN yang sudah dilatih dapat mengidentifikasi 20356 citra dengan benar dari 22560 citra tulisan tangan cetak pada *validation set*. *Training set* pada *epoch* tersebut juga berhasil mencapai angka 91.43%, atau dengan kata lain model CNN berhasil mengidentifikasi 82506 citra dengan benar dari 90.240 citra tulisan tangan cetak pada *training set*. Grafik akurasi model CNN dapat dilihat pada Gambar 5.



Gambar 5. Grafik Nilai Akurasi

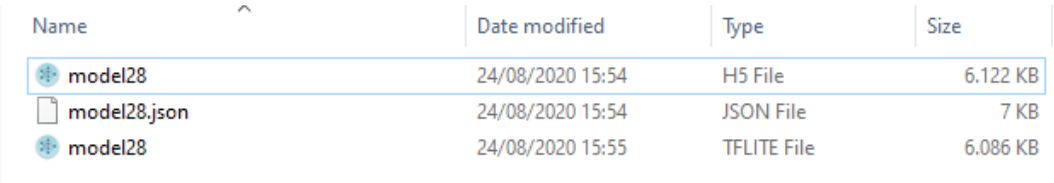
Gambar 5 menunjukkan akurasi pada *training set* dan *validation set* terhadap tiap *epoch*. Ukuran horizontal pada grafik tersebut menyatakan *epoch* dengan skalar 2.5 setiap titik, yang berarti grafik tersebut menunjukkan tingkat akurasi setiap 2.5 *epoch*. Grafik pada Gambar 5 awalnya menunjukkan nilai akurasi yang rendah. Setelah memasuki *epoch* ke-2.5 akurasi *training* mencapai 85%. Pelatihan terus dilakukan sampai *epoch* ke-20 dan didapatkan akurasi *training set* dan *validation set* tertinggi pada *epoch* ke-20. Selain akurasi, nilai *loss* juga digunakan untuk menentukan *epoch* terbaik dari pelatihan. Gambar 6 menunjukkan grafik penurunan *loss* seiring berjalannya *epoch*.



Gambar 6. Grafik Nilai *Loss*

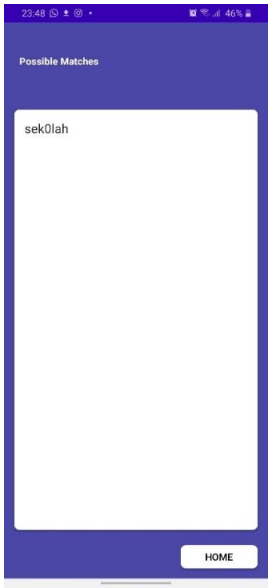
Dapat dilihat pada Gambar 6 *loss* memiliki nilai yang awalnya tinggi, namun terus menurun seiring berjalannya tiap *epoch* pada pelatihan. Dapat dilihat bahwa *epoch* ke-20 pada *training set* memiliki nilai *loss* terendah dari seluruh *epoch* yang ada. Pada *validation set*, *loss* terendah dicapai pada *epoch* ke-17, namun pada *epoch* tersebut akurasi dari *validation set* sejumlah 90,03%, sehingga *epoch* tersebut tidak dipilih sebagai *epoch* terbaik.

Proses implementasi model CNN yang telah dibuat kedalam perangkat android, yang pertama model CNN tersebut dikonversi dahulu menggunakan *library* Tensorflow Lite untuk menghasilkan model dengan format ".tflite", kemudian dilakukan proses pembuatan aplikasi android. Model CNN yang telah melalui tahap pelatihan data akan disimpan dengan pilihan 2 format, yaitu ".savedmodel" dan ".h5". Penelitian ini menggunakan format ".h5" dalam menyimpan model CNN. Baik format ".h5" maupun ".savedmodel" keduanya tidak dapat dibaca oleh Android Studio IDE, oleh sebab itu dilakukan proses konversi model menjadi format yang dapat dibaca oleh android yaitu ".tflite". Gambar 7 menunjukkan hasil dari konversi model berupa model dengan format ".tflite"



Gambar 7. Hasil dari Konversi Model CNN

Gambar 7 menunjukkan *file* dengan format TFLITE memiliki ukuran yang lebih kecil dibandingkan *file* dengan format H5. Hal itu menunjukkan bahwa hasil konversi selain untuk kompatibilitas perangkat juga untuk memperkecil ukuran *file*. Perancangan aplikasi menggunakan Android Studio IDE dengan *library* Tensorflow Lite untuk melakukan pengenalan menggunakan model CNN yang dibuat. Gambar 8 menunjukkan hasil pengenalan citra yang ditampilkan pada halaman hasil pengenalan.



Gambar 8. Hasil Pengenalan Citra Tulisan Tangan

Dapat dilihat pada Gambar 8 pengenalan yang dihasilkan adalah "sek0lah" dimana terdapat kesalahan satu huruf yaitu 'o' terbaca '0'. Kesalahan ini akan mempengaruhi akurasi

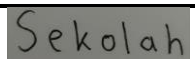
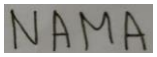
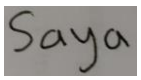
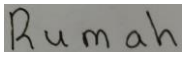
kinerja model yang akan dihitung pada tahap evaluasi kinerja model. Tabel 2 menunjukkan beberapa hasil pengenalan pada tulisan tangan cetak 1 kata.

Tabel 2. Hasil Pengenalan Tulisan Tangan Cetak

No	Tulisan	Pengenalan Huruf
1	Sekolah	sek0lah
2	NAMA	nama
3	Saya	say0
4	Rumah	rumah
5	Jakarta	jakattq
6	DEPOK	depok
7	Sayang	sayang
8	BERSIN	bqrsin
9	Keren	keren
10	Empat	empat

Berdasarkan pada hasil ujicoba pengenalan tulisan tangan cetak yang telah dilakukan, maka dapat dihitung kinerja sistem pengenalan yang telah dibangun dengan menghitung nilai *Global Performance Metric* (SM) (Tri et al., 2022) menggunakan parameter: *Detection Rate* (DR) dan *Recognition Accuracy* (RA). Tabel 3 merupakan parameter yang digunakan untuk melakukan *Global Performance Metric* Pengenalan (SM) terhadap huruf dengan menggunakan parameter *Detection Rate* Pengenalan (*DRP*) dan *Recognation Accuracy* Pengenalan (*RAP*).

Tabel 3. Nilai Parameter Pengenalan Huruf

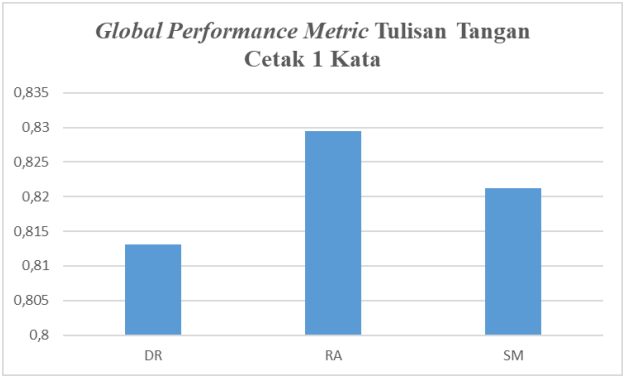
Citra Asli	Hasil Pengenalan Huruf	*GP*	*RP*	*MP*
	sek0lah	7	6	7
	nama	4	4	4
	say0	4	3	4
	rumah	5	5	5

Sebagai contoh Tabel 3 pada citra yang memiliki kata cetak “Sekolah” berdasarkan hasil ujicoba pengenalan didapatkan nilai berikut (Lampiran 2) : $GP = 7$, $RP = 6$, dan $MP = 7$, sehingga didapatkan nilai $DRP = \frac{RP}{GP} = \frac{6}{7} = 0{,}857$, dan $RAP = \frac{RP}{MP} = \frac{6}{7} = 0{,}857$. Nilai SM keseluruhan citra tulisan tangan untuk citra ini adalah $SM = \frac{2 \times DRP \times RAP}{DRP + RAP} = \frac{2 \times 0{,}857 \times 0{,}857}{0{,}857 + 0{,}857} = 0{,}857 \times 100\% = 85{,}7\%$. Tabel 4 memperlihatkan hasil kinerja sistem berdasarkan parameter DR, RA pada proses pengenalan huruf serta kinerja keseluruhan sistem SM pada citra tulisan tangan cetak 1 kata.

Tabel 4. Tabel *Global Performance* (SM)

Parameter	Pengenalan Huruf
Detection Rate	81,31%
Recognition Accuracy	82,94%
Global Performance	82,12%

Grafik *global perfomance* dari citra tulisan tangan cetak 1 kata dapat dilihat pada Gambar 9 Pengenalan huruf yang dinilai dengan parameter DR, RA menunjukkan nilai akurasi pengenalan 81,31% dan 82,94%, serta *Global Performance* menunjukkan nilai 82,12% untuk pengenalan huruf. Hasil ini menunjukkan dibutuhkan lebih banyak variasi data training mengingat pola setiap huruf juga dibedakan jenis dan tipe huruf dan hal ini dapat mempengaruhi proses pengenalan tulisan.



Gambar 9. Grafik *Global Perfomance Metric* 1 Kata

KESIMPULAN DAN SARAN

Berdasarkan hasil pengenalan dan implementasi pada android yang sudah dilakukan dengan model *Convolutional Neural Network* (CNN) terhadap citra tulisan tangan cetak pada penelitian ini, dapat diambil beberapa kesimpulan antara lain: Model CNN berhasil dibentuk dengan jumlah 11 *hidden layer*, terdiri dari 5 lapisan konvolusi, 3 lapisan *max pooling*, 1 lapisan *flatten*, dan 2 lapisan *fully connected*. Model mengambil masukan berupa citra berukuran 28 x 28 piksel dan menghasilkan keluaran berupa prediksi nama kelas dari citra tulisan tangan cetak. Total parameter bobot yang dimiliki oleh model berjumlah 1556495 variabel. Pelatihan model berhasil membentuk *training set* dan *validation set*, dengan rasio sebesar 8:2. Pelatihan dilakukan sebanyak 20 *epoch* dan diambil *epoch* dengan akurasi pada *validation set* terbaik, yaitu pada *epoch* ke-20. Model CNN yang sudah dilatih berhasil diimplementasikan dan dijalankan pada aplikasi android. Berdasarkan perhitungan *Global Performance Metric*, Aplikasi berhasil mengenali citra tulisan tangan 1 kata dengan akurasi 82,12%.

Pengembangan lebih lanjut dapat dilakukan dengan menambahkan jumlah citra yang lebih banyak lagi agar model dapat mempelajari fitur dari citra dengan lebih baik dan variasi lebih luas. Citra tulisan tangan cetak yang akan dikenali diharapkan pada penelitian selanjutnya dapat ditulis menggunakan spidol, dimana diperlukan proses skeletonisasi agar sistem dapat mengenali tulisan tangan menggunakan spidol. Pengimplementasian model CNN di perangkat android pada penelitian selanjutnya diharapkan dapat menguji lebih dari satu kata dan satu halaman dokumen tertulis dengan akurasi yang tinggi. Pengimplementasian model CNN di

perangkat android pada penelitian selanjutnya diharapkan memiliki dua opsi pengambilan citra yaitu menggunakan kamera perangkat dan mengimpor citra melalui *gallery* perangkat.

DAFTAR PUSTAKA

Aqab, S., & Tariq, M. U. (2020). Handwriting recognition using artificial intelligence neural network and image processing. *International Journal of Advanced Computer Science and Applications*, *11*(7), 137-146. https://doi.org/10.14569/IJACSA.2020.0110719

Bhunia, A. K., Das, A., Bhunia, A. K., Kishore, P. S. R., & Roy, P. P. (2019). Handwriting recognition in low-resource scripts using adversarial learning. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2019-June*(November), 4762-4771. https://doi.org/10.1109/CVPR.2019.00490

Chammas, E., Mokbel, C., & Likforman-Sulem, L. (2018). Handwriting recognition of historical documents with few labeled data. *Proceedings - 13th IAPR International Workshop on Document Analysis Systems, DAS 2018*, 43-48. https://doi.org/10.1109/DAS.2018.15

Darmatasia, & Fanany, M. I. (2017). Handwriting recognition on form document using convolutional neural network and support vector machines (CNN-SVM). *2017 5th International Conference on Information and Communication Technology, ICoIC7 2017*, *April*. https://doi.org/10.1109/ICoICT.2017.8074699

Das, M., Panda, M., & Dash, S. (2022). Enhancing the Power of CNN Using Data Augmentation Techniques for Odia Handwritten Character Recognition. *Advances in Multimedia*, *2022*. https://doi.org/10.1155/2022/6180701

Dwivedi, U., Rajput, P., Sharma, M. K., & Noida, G. (2017). Cursive handwriting recognition system using feature extraction and artificial neural network. *International Research Journal of Engineering and Technology*, *4*(03), 2202-2206.

Fitrianingsih, F., Madenda, S., Ernastuti, E., Widodo, S., & Rodiah, R. (2017). Cursive handwriting segmentation using ideal distance approach. *International Journal of Electrical and Computer Engineering*, *7*(5), 2863-2872. https://doi.org/10.11591/ijece.v7i5.pp2863-2872

Fitrianingsih, F., Susetianingtias, D. T., Pernadi, D., Patriya, E., & Arianty, R. (2022). The Implementation of Artificial Neural Network (ANN) on Offline Cursive Handwriting Image Recognition. *ILKOM Jurnal Ilmiah*, *14*(1), 63-73.

Fitrianingsih, F., Widodo, S., Madenda, S., & Rodiah, R. (2016). Slant Correction and Detection for Offline Cursive Handwriting using 2D Affine Transform. *International Journal of Engineering Research & Technology (IJERT)*, *5*(8), 568-572.

Gonzalez, R. C., & Woods, R. E. (n.d.). (2009). *Digital Image Processing, 4e*. Pearson education India.

Impedovo, D., Pirlo, G., & Plamondon, R. (2012, September). Handwritten signature verification: new advancements and open issues. In *2012 International Conference on Frontiers in Handwriting Recognition*, 367-372. https://doi.org/10.1109/ICFHR.2012.211

Ismael, S. O. (2020). *Deep Learning and Image Processing for Handwritten Style Recognition*.

J, R., & Kumar, D. P. (2020). Gesture Recognition using CNN and RNN. *International Journal of Recent Technology and Engineering (IJRTE)*, *9*(2), 230-233. https://doi.org/10.35940/ijrte.b3417.079220

Ji, B., & Chen, T. (2019). *Generative Adversarial Network for Handwritten Text*. https://doi.org/10.48550/arXiv.1907.11845

Karthi, M., Priscilla, R., & Syed, K. J. (2020). A novel content detection approach for handwritten english letters. *Procedia Computer Science*, *172*(2012), 1016-1025. https://doi.org/10.1016/j.procs.2020.05.149

Kayumov, Z., Tumakov, D., & Mosin, S. (2020). Hierarchical Convolutional Neural Network for Handwritten Digits Recognition. *Procedia Computer Science*, *171*(2019), 1927-1934. https://doi.org/10.1016/j.procs.2020.04.206

Li, P., Wang, H., Yu, M., & Li, Y. (2021, April). Overview of image smoothing algorithms. In *Journal of Physics: Conference Series*, 12-24. https://doi.org/10.1088/1742-6596/1883/1/012024

Marti, U. V., & Bunke, H. (2002). The IAM-database: an English sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, *5*, 39-46.

Memon, J., Sami, M., Khan, R. A., & Uddin, M. (2020). Handwritten optical character recognition (OCR): A comprehensive systematic literature review (SLR). *IEEE Access*, *8*, 142642-142668. https://doi.org/10.1109/ACCESS.2020.3012542

Mohsin, A., & Sadoon, M. (2020). *Developing an Arabic Handwritten Recognition System by Means of Artificial Neural Network Developing an Arabic Handwritten Recognition System by Means of Artificial Neural Network*. *June*, 1-4. https://doi.org/10.36478/jeasci.2020.1.3

Samant, Y. S. (2021). Handwritten Text Recognition Using Machine Learning. *International Journal of Advanced Research in Science, Communication and Technology*, *14*(02), 106-108. https://doi.org/10.48175/ijarsct-2006

Samsuryadi, S., Kurniawan, R., & Susilawati Mohamad, F. (2021). Automated handwriting analysis based on pattern recognition: a survey. *Indonesian Journal of Electrical Engineering and Computer Science*, *22*(1), 196-206. https://doi.org/10.11591/ijeecs.v22.i1.pp196-206

How to cite:

Susetianingtias, D. T., Arianty, R., Rodiah, R. Patriya, E. (2023). Pembentukan Model Recurrent Neural Network dan Connectionist Temporal Classification Pada Pengenalan Kata Tulisan Tangan Offline. *DECODE: Jurnal Pendidikan Teknologi Informasi, 3(2),* 172-183. http://dx.doi.org/10.51454/decode.v3i2.151