TUGAS AKHIR – IF184802

PENGENALAN TEKS TULISAN TANGAN MENGGUNAKAN *FULLY CONVOLUTIONAL NEURAL NETWORK*(FCN)

DEWI AYU NIRMALASARI
NRP 05111640000115

Dosen Pembimbing I
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya, 2020

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – IF184802

PENGENALAN TEKS TULISAN TANGAN MENGGUNAKAN *FULLY CONVOLUTIONAL NEURAL NETWORK* (FCN)

DEWI AYU NIRMALASARI
NRP 05111640000115

Dosen Pembimbing I
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya, 2020

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESIS – IF184802

HANDWRITING TEXT RECOGNITION USING FULLY CONVOLUTIONAL NEURAL NETWORK (FCN)

DEWI AYU NIRMALASARI
NRP 05111640000115

Supervisor I
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

Supervisor II
Dini Adni Navastara, S.Kom., M.Sc.

INFORMATICS ENGINEERING DEPARTMENT
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya, 2020

LEMBAR PENGESAHAN

PENGENALAN TEKS TULISAN TANGAN MENGGUNAKAN FULLY CONVOLUTIONAL NEURAL NETWORK (FCN)

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Teknik Informatika
Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh:
DEWI AYU NIRMALASARI
NRP: 05111640000115

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr.Eng. Nanik Suciati, S. Kom, M.Kom. .....
   (NIP. 19700213 199492 1 001) (Pembimbing 1)

2. Dini Adni Navastara, S.Kom., M.Sc. .....
   (NIP. 19851017 201504 2 001) (Pembimbing 2)

SURABAYA
Juni, 2020

[Halaman ini sengaja dikosongkan]

Pengenalan Teks Tulisan Tangan Menggunakan *Fully Convolutional Neural Network*

Nama	:	Dewi Ayu Nirmalasari
NRP	:	05111640000115
Jurusan	:	Departemen Teknik Informatika, Fakultas Teknologi Elektro dan Informatika Cerdas
Pembimbing I	:	Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Pembimbing II	:	Dini Adni Navastara, S.Kom., M.Sc.

ABSTRAK

Pengenalan tulisan tangan dari citra masih menjadi hal yang sukar karena penulisan huruf yang memiliki ukuran yang berbeda-beda. Solusi pertama yang ditawarkan dari pengenalan citra tulisan tangan ini ialah menggunakan *Convolutional Neural Network* (CNN) yang dilatih mengenali suatu rangkaian huruf berdasarkan kamus. Sayangnya solusi ini terbatas pada daftar kata yang muncul pada saat pelatihan. Solusi lain yang dapat dilakukan ialah menggunakan pengenalan rangkaian karakter. Pengenalan ini dilakukan dengan melakukan pembacaan *sliding window* pada citra dan memprediksi menggunakan *Fully Convolutional Neural Network* (FCN). Sebelum memasuki prediksi oleh FCN, prediksi banyak karakter perlu dilakukan terlebih dahulu. Hasil Prediksi banyak karakter ini berguna untuk mengubah ukuran citra untuk kemudian diprediksi tiap karakternya menggunakan *sliding windows*. Hasil dari pengenalan kata menggunakan gabungan metode ini memiliki akurasi sebesar 68,60%, presisi sebesar 83,45%, recall sebesar 68,80%, f1-score sebesar 73,97%, dan rata-rata CER sebesar 22,25%.

Kata Kunci: Convolutional Neural Network, Fully Convolutional Neural Network, pengenalan teks tulisan tangan

[Halaman ini sengaja dikosongkan]

HANDWRITTEN TEXT RECOGNITION USING FULLY CONVOLUTIONAL NEURAL NETWORK(FCN)

Name	:	Dewi Ayu Nirmalasari
Student ID	:	05111640000115
Department	:	Informatics Engineering Department, Faculty of Intelligent Electrical and Informatics Technology
Pembimbing I	:	Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Pembimbing II	:	Dini Adni Navastara, S.Kom., M.Sc.

ABSTRACT

Handwrittten text recognition from images is still a difficult task because of the writing of every character has different sizes. The first solution offered from this handwritten image recognition is to use a Convolutional Neural Network (CNN) which is trained to recognize a series of characters based on a dictionary. Unfortunately, this solution is limited to the glossary that appears during the training. Another solution that can be done is to use character sequence recognition. This recognition is done by reading using sliding window on the image and predicting it using the Fully Convolutional Neural Network (FCN). Before entering into predictions by FCN, character count prediction is need to be done first. This prediction's result is useful for resizing the image to be predicted each character using sliding windows. The results of word recognition using this combination of methods have an accuracy of 68.60%, a precision of 83.45%, a recall of 68.80%, an f1-score of 73.97%, and an average CER of 22.25%.

Keywords: Convolutional Neural Network, Fully Convolutional Neural Network, handwritten text recognition

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa. Atas rahmat dan kasih sayang-Nya, penulis dapat menyelesaikan Tugas Akhir yang berjudul:

"**Pengenalan Teks Tulisan Tangan Menggunakan *Fully Convolutional Neural Network* (FCN)**"

Pengerjaan Tugas Akhir ini menjadi suatu pengalaman yang baik bagi penulis. Penulis dapat memperoleh banyak pengalaman yang berharga dalam memperdalam dan meningkatkan keilmuan dalam bidang Informatika selama perkuliahan di Departmen Informatika ITS.

Terselesaikannya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan banyak pihak, oleh karena itu melalui lembar ini penulis ingin mengucapkan terima kasih dan penghormatan kepada:

1. Kedua orangtua penulis dan anggota keluarga lainnya yang telah memberikan dukungan doa, moral, dan material kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
2. Ibu Dr. Eng. Nanik Suciati, S.Kom., M.Kom. dan Dini Adni Navastara, S.Kom., M.Sc. selaku Pembimbing I dan Pembimbing II yang telah membimbing, memberikan motivasi, dan masukan dalam menyelesaikan Tugas Akhir ini.
3. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom., Dr. Radityo Anggoro , S.Kom, M.Sc., dan Ary Mazharuddin Shiddiqi S.Kom., M.Comp.Sc. selaku pihak manajemen Departemen Teknik Informatika ITS yang turut banyak membantu penulis dalam menyelesaikan Tugas Akhir ini.
4. Frandita Adhitama yang selalu memberikan dukungan dan nasihat yang baik pada penulis.

5. Nurlita, Tita, Ferdinand, CT, Michael, dan Nisha yang telah menemani dan membantu penulis selama perkuliahan hingga pada masa pengerjaan Tugas Akhir ini.
6. Rasyid, Dandy, Ghisa, dan Aufa yang banyak memberi saran mengenai pengerjaan Tugas Akhir ini.
7. Chintya dan Arin yang selalu siap mendukung dan membantu penulis dari jauh.
8. Firza, Evelyn, Adam, Vira, Safhira, Ayas, dan Zahra yang membantu memberikan data uji.
9. Teman-teman admin Laboratorium Pemorgraman yang selalu mendukung penulis selama berkuliah.
10. Seluruh mahasiswa Informatika ITS angkatan 2016 yang telah menjadi teman penulis selama menjalani masa kuliah di Informatika ITS.
11. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa laporan Tugas Akhir ini masih memiliki banyak kekurangan. Oleh karena itu dengan segala kerendahan hati penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan penulis kedepannya. Selain itu, penulis berharap laporan Tugas Akhir ini dapat berguna bagi pembaca secara umum.

Surabaya, Juni 2020

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

DAFTAR KODE SUMBER

DAFTAR GAMBAR

BAB I
PENDAHULUAN

Pada bab ini, akan dijelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat metodologi pengerjaan, dan sistematika penulisan tugas akhir.

1.1 Latar Belakang

Seiring berkembangnya teknologi, kegiatan menulis tangan masih belum ditinggalkan. Hal ini disebabkan kegiatan menulis tangan masih lebih mudah, efisien, dan murah. Meskipun kegiatan menulis tangan masih disenangi, banyak dokumen penting seperti cek, dokumen pajak, atau kuisioner yang masih perlu diisi manual pada beberapa isian perlu didigitalisasikan. Berbeda dengan pengenalan OCR (*Optical Character Recognition*), tulisan tangan lebih sukar untuk dikenali karena lebar antar huruf yang berbeda-beda meskipun dituliskan oleh orang yang sama.

Convolutional Neural Network (CNN) merupakan salah satu *neural network* yang sering digunakan dalam proses pengenalan citra. CNN sendiri memilliki banyak modifikasi, salah satunya ialah *Fully Convolutional Neural Network* (FCN). FCN tetap menggunakan *convolution layer* pada saat memprediksi, sehingga FCN tidak memiliki ketergantungan pada ukuran citra.

Dalam mengenali tulisan tangan dari citra, terdapat beberapa pendekatan *deep learning* yang dapat dilakukan. Salah satu pendekatan ini ialah membuat sistem pengenalan citra tulisan kata dengan melatih sistem ini dengan citra tulisan kata juga. Sistem ini sendiri nantinya mampu menghasilkan pengenalan yang akurat, tetapi terbatas pada pemilihan kata pada citra data latih. Pada pendekatan ini juga dibutuhkan dataset yang cukup besar karena ketergantungannya pada citra kata.

Pendekatan lain yang dapat dilakukan ialah pengenalan karakter demi karakter pada citra tulisan. Metode ini sendiri masih menjadi tantangan tersendiri karena seringnya terjadi kesalahan

pembacaan karakter dari suatu citra. Kesalahan yang biasa terjadi ialah adanya kata yang terlewatkan atau pembacaan karakter yang terduplikasi. Tantangan lain dari pengenalan kata ini ialah perbedaan lebar karakter pada satu citra. Dalam hal ini, penggunaan FCN akan menguntungkan karena FCN dapat menerima masukan tanpa batasan ukuran panjang. Nantinya karakter demi karakter ini akan dikenali menggunakan algoritma *sliding windows* sesuai yang telah dijabarkan pada [1].

Berdasarkan hal-hal tersebut, pada tugas akhir ini akan diimplementasikan metode-metode pengenalan karakter dari citra tulisan tangan. Adapun metode pertama yang dilakukan adalah pengenalan citra kata berdasarkan kamus kata yang telah di-tentukan sebelumnya. Pada metode ini akan digunakan CNN. Saat sistem tidak mampu memberi hasil prediksi yang meyakinkan, maka citra kata akan dikenali karakter per karakter menggunakan FCN. Sebelum sistem memasuki pengenalan karakter per karaker ini, sistem akan mengenali banyak karakter pada citra menggunakan CNN.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana cara mengenali citra tulisan tangan berdasarkan kelas-kelas kata yang telah ditentukan sebelumnya menggunakan CNN?
2. Bagaimana cara mengenali banyak karakter pada citra tulisan tangan menggunakan CNN?
3. Bagaimana cara mengenali citra tulisan tangan menggunakan FCN?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada tugas akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Implementasi program menggunakan bahasa pemro-graman Python 3.

2. Dataset pelatihan yang digunakan diambil dari NIST Special Database 19.
3. Dataset pengujian diambil dari tulisan tangan pada layar sentuh.
4. Masukan sistem berupa citra tulisan tangan berbentuk kata.
5. Sistem yang dibangun mengenali tulisan tangan yang tidak dalam bentuk tulisan latin.
6. Terdapat 62 kelas karakter yang perlu dikenali, meliputi digit angka, huruf alfabet kapital maupun huruf kecil.

1.4 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah pengenalan citra teks tulisan tangan menggunakan *Fully Convolutional Neural Network* (FCN).

1.5 Manfaat

Tugas akhir ini diharapkan dapat menghasilkan sebuah sistem yang memiliki akurasi yang tinggi dalam pengenalan teks tulisan tangan.

1.6 Metodologi

Pembuatan tugas akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1. Penyusunan Proposal Tugas Akhir

Proposal Tugas Akhir ini berisi tentang latar belakang diajukannya usulan Tugas Akhir, rumusan masalah yang akan diangkat untuk Tugas Akhir, batasan masalah yang digunakan untuk pengerjaan, tujuan pembuatan Tugas Akhir, manfaat dari pembuatan Tugas Akhir, tinjauan pustaka yang merupakan referensi untuk pengerjaan Tugas Akhir, ringkasan isi Tugas Akhir, metodologi yang merupakan tahapan-tahapan yang dilakukan pada Tugas Akhir, dan jadwal pengerjaan Tugas Akhir.

2. Studi Literatur

Pada tahap ini akan dilakukan pendalaman studi literatur yang relevan untuk dijadikan referensi dalam melakukan pengerjaan Tugas Akhir. Studi literatur didapatkan dari buku, internet, dan materi-materi kuliah yang berhubungan dengan metode yang akan digunakan.

3. Implementasi Perangkat Lunak

Pada tahap ini akan dilaksanakan implementasi metode dan algoritma yang telah direncanakan. Implementasi sistem menggunakan *Python 3* sebagai bahasa pemrograman, *TensorFlow* dan *Keras* sebagai *framework*, serta *library* pendukung lainya.

4. Pengujian dan Evaluasi

Tahap pengujian dan evaluasi akan dilakukan untuk mengetahui hasil dan performa algoritma yang telah diimplementasikan berupa data foto yang diambil manual. Selanjutnya pada tahap evaluasi, digunakan *Confusion Matrix* yang memiliki empat hasil, yaitu akurasi, presisi, recall, dan *f1-score*.

5. Penyusunan Buku

Pada tahap ini dilakukan penyusunan buku yang menjelaskan seluruh konsep, teori dasar dari metode yang digunakan, implementasi, serta hasil yang telah dikerjakan sebagai dokumentasi dari pelaksanaan Tugas Akhir.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan tugas akhir adalah sebagai berikut:

1. BABI: PENDAHULUAN
   Bab ini berisi latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi dan sistematika penulisan tugas akhir.

2. BAB II: DASAR TEORI
   Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan tugas akhir ini.

3. BAB III: DESAIN
   Bab ini berisi perancangan dari metode *Convolutinal Neural Network* dan *Fully Convolutional Neural Network* yang digunakan dalam pengenalan tulisan tangan dari citra.

4. BAB IV: IMPLEMENTASI
   Bab ini membahas implementasi dari perancangan yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode sumber yang digunakan dalam proses implementasi.

5. BAB V: PENGUJIAN DAN EVALUASI
   Bab ini berisi uji coba dan evaluasi dari hasil implementasi yang telah dilakukan pada tahap implementasi.

6. BAB VI: PENUTUP
   Bab ini berisi kesimpulan dan saran yang didapat dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses dan tertulis saat pengerjaan tugas akhir, dan saran untuk pengembangan solusi ke depannya.

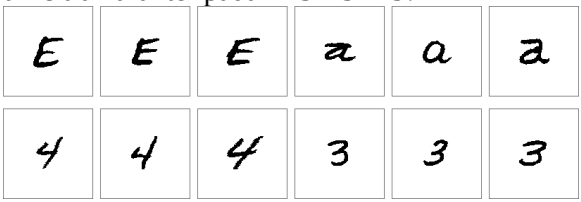
[Halaman ini sengaja dikosongkan]

BAB II
TINJAUAN PUSTAKA

Bab ini membahas mengenai teori-teori dasar yang digunakan dalam Tugas Akhir. Teori-teori tersebut mengenai citra, sintesis citra, pengolahan citra digital, *Convolutional Neural Network*, *Fully Convolutional Neural Network* dan beberapa teori lain yang mendukung pembuatan Tugas Akhir. Penjelasan ini bertujuan untuk memberikan gambaran umum dan diharapkan dapat mendukung sistem yang dibangun.

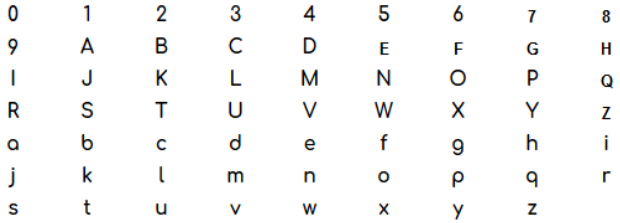
2.1 NIST Special Database 19

NIST *Special Database* 19 (SD19) merupakan dataset karakter angka dan alfabet, baik huruf besar maupun huruf kecil. Dataset ini sendiri diambil dari 3.669 formulir tulisan tangan dan memiliki total 814.225 karakter. Citra karakter disimpan dalam ekstensi *Portable Network Graphics*(.png) dan masing-masing citra memiliki ukuran 128x128. NIST sendiri pertama kali di-publikasikan pada tahun 1995 oleh *National Institute of Standards and Technology* (NIST) [2]. Gambar 2.1 menampilkan beberapa contoh dari citra karakter pada NIST SD19.



Gambar 2.1 Contoh Citra Karakter pada NIST SD19

Terdapat 62 jenis citra karakter yang disediakan NIST SD19, yang terdiri dari alfabet huruf besar dan kecil serta digit. Masing-masing kelas memiliki banyak citra yang berbeda-beda, dari yang paling sedikit sejumlah 1.920 hingga yang paling banyak sejumlah 38.184 buah.



Gambar 2.2 Kelas Citra Karakter yang Tersedia pada NIST SD19

2.2 Corpus Brown

Corpus Brown merupakan salah satu *corpus* yang disediakan oleh NLTK, salah satu *library* yang akan digunakan pada tugas akhir ini. *Corpus* ini dirilis oleh Brown University pada tahun 1961. *Corpus* ini sendiri memiliki 15 kategori, 500 artikel, dan 1.15 juta kata dalam bahasa Inggris [3]. Daftar kategori beserta banyak artikel Brown dapat dilihat pada Tabel 2.1.

Tabel 2.1 Kategori Artikel pada Corpus Brown

Kode	Kategori	Banyak Artikel
A	Berita	44
B	Tajuk rencana	27
C	Ulasan	17
D	Agama atau kepercayaan	17
E	Hobi	36
F	Wawasan	48
G	*Belles lettres*	75
H	Politik	30
J	Pelajaran	80
K	Fiksi	29
L	Misteri	24
M	Fiksi ilmiah	6
N	Petualangan	29
P	Roman	29
R	Humor	9

2.3 Corpus Guntenberg

Corpus Gutenberg juga merupakan salah satu *corpus* yang disediakan oleh NLTK. Berbeda dengan Brown, *corpus* ini berisikan buku elektronik. *Corpus* ini memiliki kurang lebih dua juta kata dari 18 buku elektronik. *Corpus* ini diambil dari Gutenberg *Project*, sebuah proyek pengarsipan buku *online* gratis dan memiliki kurang lebih 25.000 buku elektronik gratis [3]. Buku-buku yang tersedia pada *corpus* ini dapat dilihat pada Tabel 2.2.

Tabel 2.2 Daftar Buku pada Corpus Gutenberg

Kode	Judul Buku
auesten-emma.txt	Emma, oleh Jane Austen
austen-persuasion.txt	Persuasion, oleh Jane Austen
austen-sense.txt	Sense and Sensibility, oleh Jane Austen
bible-kjv.txt	The King James Bible
blake-poem.txt	Poems of William Blake, oleh William Blake
bryant-stories.txt	Stories to Tell Children, oleh Sarah Cone Bryant
burgess-busterbrown.txt	The Adventures of Buster Bear, oleh Thornton W. Burgess
carroll-alice.txt	Alice's Adventures in Wonderland, oleh Lewis Carroll
chesterton-ball.txt	The Ball and the Cross, oleh G.K. Chesterton
chestertown-brown.txt	The Wisdom of Father Brown, oleh G.K. Chesterton
chesterton-thursday.txt	The Man Who Was Thursday, oleh G.K. Chesterton
edgeworth-parents.txt	The Parent's Assistant, oleh Maria Edgeworth
melville-moby_dick.txt	Moby Dick. oleh Herman Melville

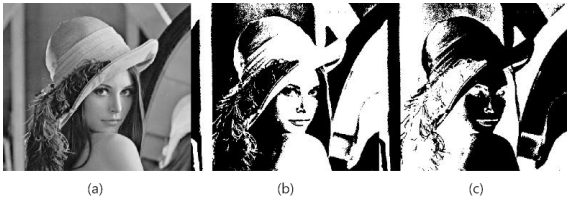
milton-paradise.txt	Paradise Lost, oleh John Milton
shakespeare-caesar.txt	The Tragedie of Julius Caesar, oleh William Shakespeare
shakespeare-hamlet.txt	The Tragedie of Hamlet, oleh William Shakespeare
shakespeare-macbeth.txt	The Tragedie of Macbeth, oleh William Shakespeare
whitman-leaves.txt	Leaves of Grass, oleh Walt Whitman

2.4 Pengolahan Citra Digital

Komputer mendefinisikan citra sebagai *array* dua dimensi, yang memiliki nilai pada lokasi-lokasi yang tersedia. Nilai ini dikenal juga sebagai elemen citra, elemen gambar, atau pixel. Pengolahan citra digital berarti kegiatan memproses suatu citra digital menggunakan komputer. Kegiatan ini ditujukan untuk menyempurnakan gambar agar dapat mengekstrak informasi yang berguna[4]. Beberapa kegiatan pengolahan citra digital yang akan digunakan pada tugas akhir ini akan dijelaskan berikut.

2.4.1 Thresholding

Thresholding merupakan salah satu pemrosesan citra dengan cara mengubah nilai warna pada *pixel* menjadi hitam atau putih yang biasa digunakan untuk segmentasi. *Thresholding* yang paling umum dikenal dengan *binary thresholding*. *Thresholding* ini memiliki nilai ambang dimana ketika nilai warna pada pixel lebih besar dari nilai ambang, *pixel* akan berubah warna putih. Sebaliknya, ketika nilai warna pada *pixel* lebih kecil, maka *pixel* akan berubah warna hitam. *Thresholding* ini sendiri memiliki variasi lain, seperti i*nverse binary thresholding*, yang bekerja berkebalikan dengan *binary thresholding* [5]. Contoh *binary thresholding* dan *inverse binary thresholding* dapat dilihat pada Gambar 2.3.



Gambar 2.3 Contoh Penggunaan *Thresholding* (a) Citra Awal; (b) Citra Hasil *Thresholding*; (c) Citra Hasil *Inverse Thresholding*

Selain dua jenis *thresholding* tersebut yang ditentukan berdasarkan nilai masukan, ada juga Otsu *thresholding* yang bekerja secara otomatis menentukan nilai ambang. Metode *thresholding* ini mengembalikan ambang intensitas yang memisahkan *pixel* ke dalam dua kelas, yakni *background* dan foreground. Contoh Otsu *thresholding* dapat dilihat pada Gambar 2.4.

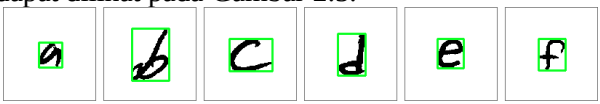
Gambar 2.4 Contoh Otsu *Thresholding* (a) Citra awal; (b) Citra Hasil Otsu *Threholding*

2.4.2 Deteksi Kontur

Kontur merupakan batas dari suatu objek pada citra. Kontur dapat digunakan untuk melakukan segmentasi pada beberapa obyek dalam satu citra. Algoritma kontur mencoba melacak batas untuk setiap objek dengan mengelompokkan *edge* dan membentuk iterasi tertutup. Setiap iterasi tertutup inilah mewakili satu kontur.

Dalam deteksi objek dengan kontur, OpenCV menyediakan berbagai macam bentuk kontur. Pada tugas akhir ini

akan digunakan bentuk kontur persegi panjang [6]. Contoh deteksi kontur dapat dilihat pada Gambar 2.5.



Gambar 2.5 Contoh Citra dengan Visualisasi Hasil Deteksi Kontur

2.4.3 Gaussian Blur

Gaussian *blur* merupakan salah satu teknik yang digunakan dalam mengaburkan citra. Pengaburan citra biasa digunakan dalam praproses citra.

Gaussian *blur* akan memindai seluruh pixel pada citra dan menghtiung ulang nilai suatu pixel berdasarkan nilai-nilai *pixel* yang berdekatan. *Pixel-pixel* berdekatan, atau dapat disebut *kernel*, ini memiliki nilai yang berbeda-beda. Semakin dekat ke *pixel* utama, semakin besar pengaruh *pixel* tersebut pada *pixel* utama. *Kernel* ini harus bernilai ganjil dan positif. Sigma pada operasi ini bekerja untuk menentukan seberapa besar standar deviasi pada fungsi ini [7]. Matriks *kernel* Gaussian ini didapat dari distribusi Gaussian, seperti yang dapat dilihat pada Persamaan (2.1).

$$G(i,j) = c.e\ \frac{(i-u)^2 + (j-v)^2}{2\sigma^2} \qquad (2.1)$$

Dimana c dan σ merupakan konstanta, (i,j) adalaj matriks kernel pada posisi (i,j), dan (u,v) indeks tengah dari matriks kernel Gaussian [8]. Ilustrasi Gaussian *blur* dapat dilihat pada Gambar 2.6.

Gambar 2.6 Contoh Gaussian Blur (a) Citra awal; (b) Citra Hasil Gaussian Blur

2.5 Regular Expression

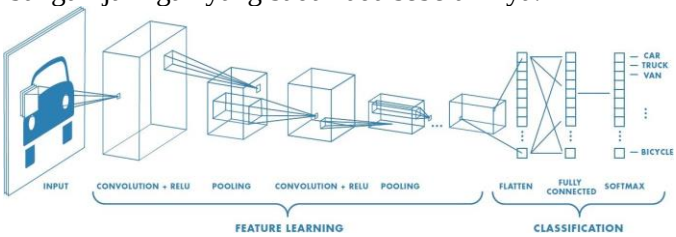
Regular Expression atau yang biasa disingkat menjadi RE, regex, ataupun regexp merupakan suatu notasi aljabar yang digunakan untuk mendefinisikan serangkaian karakter. RE sendiri sering digunakan dalam pencarian pola dari kumpulan karakter. RE dapat ditemukan pada berbagai bahasa komputer, prosesor kata, atau alat pemrosesan teks. Tabel 2.3 menampilkan beberapa contoh dari operasi RE [9].

Tabel 2.3 Contoh RE

RE	Hasil yang Didapat	Contoh
/[A-Z]/	Huruf alfabet kapital	”we should call it ‘Drenched Blossoms’”
/[12345]/	Angka 1 sampai 5	“plenty of 3 to 5”
/s	*whitespace*	”Oyfn pripetchik”

2.6 Convolutional Neural Network

Convolutional Neural Network, atau yang biasa disingkat menjadi CNN atau ConvNet, merupakan algoritma populer dalam *deep learning*. CNN sangat berguna dalam menemukan pola dalam gambar untuk mengenali objek, wajah, atau pemandangan. CNN mempelajari langsung dari data citra dan mengeliminasi fitur ekstrasi manual. Aplikasi yang menggunakan komputer visi dan pengenalan objek biasanya menggunakan CNN, seperti mobil tanpa awak atau aplikasi pengenalan wajah. CNN sendiri dapat dilatih ulang untuk kegiatan pengenalan baru, yang memungkinkan membangun jaringan yang sudah ada sebelumnya.

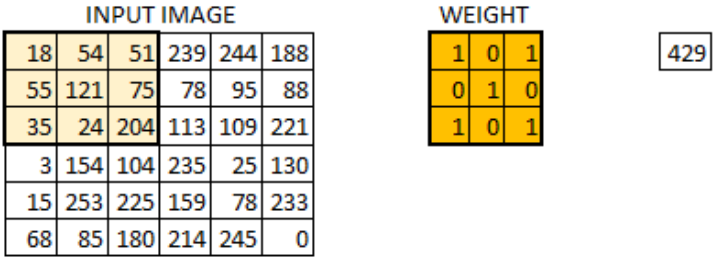


Gambar 2.7 Ilustrasi Arsitektur CNN [10]

Arsitektur dari CNN dibagi menjadi dua bagian besar, *extraction layer* dan *classification layer* [11], seperti yang dapat dilihat pada Gambar 2.7. Pada *extraction layer* terjadi proses penerjemahan dari sebuah citra menjadi *feature map* berisi angka-angka yang merepresentasikan citra tersebut. Bagian ini terdiri dari *convolutional layer* dan *pooling layer*. *Feature map* yang dihasilkan dari *convolutional layer* masih berbentuk *array* multidimensi, sehingga harus diubah menjadi sebuah *feature* vector agar bisa digunakan sebagai masukan untuk *fully connected layer* pada *classification layer*. *Fully connected layer* yang dimaksud di sini adalah MLP yang memiliki beberapa *hidden layer*, *activation function*, *output layer*, dan *loss function*.

2.6.1 Convolution Layer

Convolution layer melakukan operasi konvolusi pada masukan *layer* tersebut. Konvolusi adalah suatu istilah matematis yang berati mengaplikasikan sebuah fungsi pada output fungsi lain secara berulang. Tujuan dari operasi ini sendiri ialah untuk mengekstrak fitur dari citra masukan [12]. Ilustrasi operasi konvolusi dapat dilihat pada Gambar 2.8.

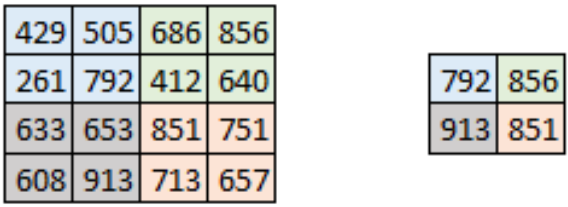


Gambar 2.8 Ilustrasi Operasi Konvolusi [13]

2.6.2 Pooling Layer

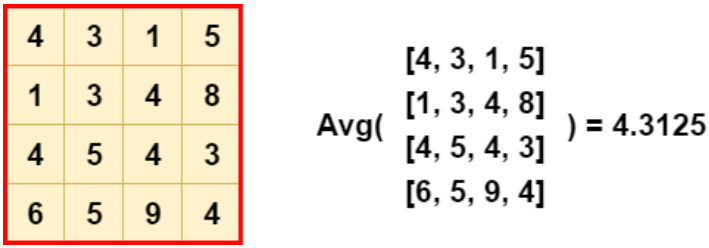
Pooling layer atau layer subsampling bekerja mereduksi ukuran sebuah data citra. Hal ini bertujuan untuk meningkatkan invariansi posisi dari fitur. Pada CNN, metode yang biasa digunakan ialah *max pooling*. *Max pooling* bekerja dengan cara

membagi citra menjadi wilayah-wilayah kecil lalu mengambil nilai tertinggi dari tiap wilayah tersebut. Ilustrasi kerja *max pooling* dapat dilihat pada Gambar 2.9.



Gambar 2.9 Ilustrasi Kerja *Max Pooling* [13]

Selain *max pooling*, metode lain yang dapat digunakan pada *layer* ini adalah *global average pooling*. Dengan metode ini, citra tidak akan dibagi menjadi wilayah kecil-kecil. Sesuai namanya, dengan metode ini akan diambil rata-rata dari seluruh masukan. Ilustrasi kerja *global average pooling* dapat dilihat pada Gambar 2.10.



Gambar 2.10 Ilustrasi Kerja *Global Average Pooling* [14]

2.6.3 Fully Connected Layer

Layer ini biasanya bekerja sebagai penerapan MLP (*Multi Layer Perceptron*) dan bertujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear. Setiap *neuron* pada *convolutional layer* perlu ditransformasi menjadi satu dimensi dulu sebelum masuk ke *fully connected layer*.

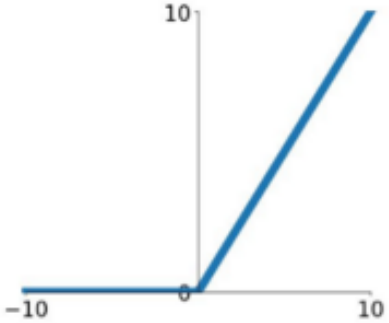
Karena hal ini menyebabkan data kehilangan informasi spasial dan tidak *reversible*, *fully connected layer* hanya dapat dimplementasikan pada akhir jaringan.

2.6.4 Fungsi Aktivasi ReLU (Rectified Linear Units)

Fungsi aktivasi bekerja menentukan suatu neuron harus aktif atau tidak. Fungsi ReLU (*Rectified Linear Units*) bersifat linear untuk semua nilai positif dan nilai nol untuk semua nilai negatif. ReLU didefinisikan pada Persamaan (2.2).

$$ReLU(x) = \begin{cases} 0 \; jika \; x \; < 0 \\ x \; jika \; x \; \geq 0 \end{cases} \tag{2.2}$$

ReLU sendiri bersifat ringan dalam komputasi karena tidak melibatkan operasi matematika yang rumit dan mudah dioptimasi. Namun karena semua nilai negatif akan langsung dibuang oleh ReLU, hal ini menyebabkan neuron yang bernilai negatif akan lebih susah untuk diperoleh kembali. ReLU sendiri hanya dapat digunakan pada *hidden layers* dari *neural network*, tidak dapat digunakn pada *output layer*. Gambar 2.11 menampilkan grafik dari fungsi aktivasi ReLU.



Gambar 2.11 Fungsi Aktivasi ReLU [15].

2.6.5 Fungsi Aktivasi Maxout

Fungsi aktivasi maxout bekerja mengatasi kekurangan ReLU dalam menentukan aktivitas neuron yang telah bernilai negative. Fungsi ini didefinisikan pada Persamaan (2.3).

$$\max (w_1^T x + b_1,\ w_2^T x + b_2 \ ) \tag{2.3}$$

Fungsi ini sendiri memakan memori dan waktu lebih banyak daripada fungsi ReLU karena maxout menggandakan jumlah parameter pada setiap neuron [15].

2.6.6 Fungsi Aktivasi Softmax

Fungsi aktivasi *softmax* bekerja dalam mengklasfikasi dengan memberikan nilai probabilitas pada setiap label kelas dari suatu data. Total dari nilai probabilitas pada data tersebut ialah 1. *Softmax* pada dasarnya adalah probabilitas eksponensial yang dinormalisasi dari nilai masukan sejumlah kelas pada model klasifikasi seperti pada Persamaan (2.4).

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \tag{2.4}$$

Variabel *y* merupakan nilai masukan. Operasi akan menghasilkan nilai probabilitas. Label dari data masukan akan ditentukan berdasarkan kelas dengan nilai probabilitas tertinggi.

2.6.7 Cross Entropy

Loss function merupakan fungsi yang menggambar kerugian dari suatu model. Semakin rendah *error* dihasilkan, semakin baik *loss function* pada model. Dalam klasifikasi banyak kelas, *loss function* yang biasa digunakan adalah *cross entropy*. *Cross entropy* menghitung *error* antara nilai prediksi S dengan nilai sebenarnya T. Selanjutnya, nilai *error* diambil dari rata-rata

hasil *cross entropy*. Persamaan ini dapat dilihat pada Persamaan (2.5) dan (2.6).

$$D(S_i, T_i) = -\sum_j T_{ij} \log S_{ij} \tag{2.5}$$

$$J(W, b) = \frac{1}{n} \sum_i D(S_i, T_i) \tag{2.6}$$

2.6.8 Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) merupakan salah satu algoritma pengoptimalan. Algorita ini bekerja untuk menemukan parameter yang dapat meminimalisir *error* dari *loss function*. SGD bekerja dengan memperbarui nilai bobot dan bias pada neuron di *neural network*. Pada dasarnya operasi yang dilakukan hanya mengurangi bobot awal dengan nilai *learning rate* dari nilai gradien yang sudah didapat. Persamaan ini dapat dilihat pada Persamaan (2.7) dan (2.8).

$$w_{j+1} = w_j - \alpha \frac{\partial}{\partial w_j} J(W, b) \tag{2.7}$$

$$b_{j+1} = w_j - \alpha \frac{\partial}{\partial b_j} J(W, b) \tag{2.8}$$

Dimana α merupakan *learning rate* awal dan J merupakan *cost function*.

2.6.9 RMSProp

RMSProp (*Root Mean Square Propagation*) adalah metode pengoptimalan berbasis *adaptive learning rate* yang diusulkan oleh Geoffrey Hinton [16]. RMSProp bekerja dengan memberikan bobot seacara kuadratik pada rata-rata gerak gradient. Hal ini dapat dilihat pada (2.9) dan (2.10).

$$s_j = \beta . s_{j-1} + (1 - \beta)(g_j)^2 \tag{2.9}$$

$$\theta_{j+1} = \theta_j - \frac{\alpha}{\sqrt{s_j+\varepsilon}}\, g_j \quad (2.10)$$

2.6.10 Adam

Adam (*Adaptive Moment Estimation*) merupakan salah satu algoritma pengoptimalan. Adam merupakan jenis algoritma pengoptimalan yang banyak digunakan. Adam memiliki *learning rate* pada setiap parameter dan secara terpisah beradaptasi pada saat pelatihan. Adam memperbarui nilai setiap parameter seperti RMSProp [17]. Perbedaannya Adam menggunakan gradien yang telah diperhalus dan semakin mengecil seperti yang dapat dilihat pada Persamaan (2.11). Lalu gradien tersebut akan digunakan untuk memperbarui parameter, seperti yang dapat dilihat pada Persamaan (2.12) dan (2.13).

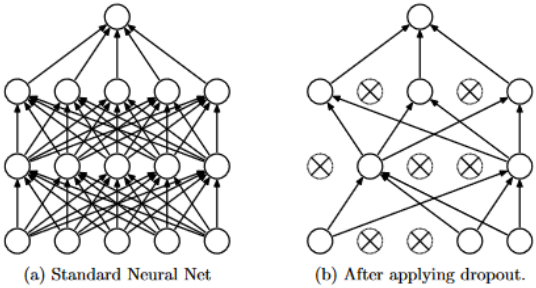
$$m_j = \beta_1 . m_{j-1} + (1-\beta_1) . g_j \quad (2.11)$$

$$s_j = \beta_2 . s_{j-1} + (1-\beta_2)(g_j)^2 \quad (2.12)$$

$$\theta_{j+1} = \theta_j - \frac{\alpha}{\sqrt{s_j+\varepsilon}}\, m_j \quad (2.13)$$

2.6.11 Dropout

Dropout merupakan salah satu cara mengatasi terjadinya *overfitting* dan juga dapat mempercepat proses *learning*. *Dropout* bekerja dengan menghilangkan *neuron* yang dapat berupa *hidden layer* maupun *visible* layer dalam jaringan. Parameter pada *dropout* mendefinisikan besaran probabilitas pada output layer dropout. Nilai umum probabilitas untuk mempertahankan output dari tiap node adalah 0,5 untuk *hidden layer*, sedangkan angka-angka mendekati 1.0 cocok untuk *visible layer* [18]. Gambar 2.12 membe-rikan ilustrasi mengenai *dropout.*



Gambar 2.12 Ilustrasi Dropout [19]

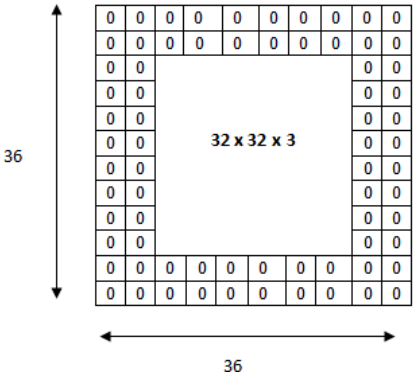
2.6.12 Batch Normalization

Batch normalization merupakan layer yang ada pada *neural network* yang bekerja melakukan normalisasi pada kumpulan data masukan, seperti pada Persamaan (2.14). Variabel x adalah nilai masukan, μb adalah rata-rata dari *batch*, σb adalah standar deviasi dari *batch*. Normalisasi dilakukan agar data memiliki mean mendekati 0 dan standar deviasi mendekati 1. Dengan begitu, normalisasi dapat meningkatkan performa dan kestabilan *neural network* [20].

$$x_{baru} = \frac{x - \mu_b}{\sigma_b} \tag{2.14}$$

2.6.13 Zero Padding

Zero padding adalah teknik penambahan nilai 0 pada sisi data masukan. Teknik ini digunakan untuk memanipulasi dimensi *output* dari *convolution layer*. Dengan menambahkan *padding* nilai 0 ini, dimensi data keluaran akan tetap memiliki ukuran yang sama atau berubah ke dalam ukuran tertentu sesuai dengan besaran *padding* yang ditentukan. Dengan adanya *padding*, *convolutional layer* akan lebih banyak mndapat fitur untuk dikelola [21]. Ilustrasi cara kerja *zero padding* dengan *padding* 2x2 dapat dilihat pada Gambar 2.13.



Gambar 2.13 Ilustrasi Zero Padding

2.6.14 Regularizer

Permasalahan yang biasanya muncul dalam melatih *neural network* ialah *overfitting*. *Overfitting* merupakan kondisi dimana suatu algoritma pembelajaran berjalan sangat baik dalam mengoptimasi *error* pada saat pelatihan, namun gagal dalam pengujian. Ketika *overfitting* terjadi, penanganan yang paling umum diberikan ialah dengan menambahkan *regularizer*. *Regularizer* bekerja dengan menambahkan pinalti pada fungsi *loss* [22]. Teknik yang umum digunakan dalam *regularization* umumnya ada dua, yakni:

a. L2 *regularizer*
   L2 *regularizer* bekerja dengan cara mengkuadratkan pinalti pada bobot. Hal ini memaksa bobot berkurang mendekati nol. Regulator L2 berguna untuk mengompress model.
b. L1 *regularizer*
   Regulator L1 bekerja dengan cara memutlakkan pinalti dan mengalikannya pada bobot. Pada regulator L1 bobot bisa jadi mencapai ke angka 0.

2.7 Fully Convolutional Neural Network

Sesuai dengan namanya, *Fully Convolutional Neural Network* (FCN) merupakan pengembangan dari CNN dimana semua *layer* yang digunakan hanyalah *convolutional layer*. Alih-alih menggunakan *fully-connected layer* seperti pada CNN dalam prediksi, FCN tetap menggunakan *convolutional layer* untuk mengklasifikasi data ke dalam kelas-kelas. Tujuan utama dari FCN ialah menghasilkan segmentasi semantik, dimana output memiliki ukuran yang sama dengan gambar input dan menyerupai gambar aslinya. Output FCN sendiri melakukan prediksi pada pixel-pixel pada citra [23].

2.8 Character Error Rate (CER)

Character Error Rate (CER) merupakan suatu cara menilai akurasi dari dengan membandingkan kata sebenarnya dengan kata hasil prediksi. Adapun persamaan CER dapat dilihat pada Persamaan (2.15) [1].

$$CER = \frac{R + D + I}{R + D + I + C} \tag{2.15}$$

dimana R merupakan banyaknya karakter yang berubah, D banyaknya karakter yang terhapus, I banyaknya karakter yang bertambah serta C banyak karakter yang benar. Dalam membangun algoritma dari CER, digunakan *dynamic programming* berbentuk *array* dua dimensi dengan $C_{0,0} = 0$ dan $CER = C_{h,l}$, dimana h merupakan banyak karakter hasil prediksi dan l merupakan banyak karakter pada kata sebenarnya. Adapun persamaan yang dilakukan pada tiap sel pada array dua dimensi dapat dilihat pada Persamaan (2.16) dan (2.17).

$$C_{i,j} = \min\left(C_{i-1,j} + 1, C_{i,j-1} + 1, Diag\right) \tag{2.16}$$

$$Diag = \begin{cases} C_{i-1,j-1} & saat\ p_i = L_j \\ C_{i-1,j-1} + 1 \end{cases} \tag{2.17}$$

Ilustrasi perhitungan CER sendiri dapat dilihat pada Tabel 2.4. Pada gambar tersebut dapat dilihat sedang dihitung CER dari kata ‘blink’ dan ‘blank.

Tabel 2.4 Ilustrasi Perhitungan CER

		b	l	i	n	k
	0	1	2	3	4	5
b	1	0	1	2	3	4
l	2	1	0	1	2	4
a	3	2	1	1	2	4
n	4	3	2	2	1	2
k	5	4	3	3	2	1

2.9 *Confusion Matrix*

Confusion Matrix merupakan cara mengevaluasi suatu model klasi-fikasi pembelajaran mesin. Tabel 2.5 menggambarkan kombinasi nilai prediksi dan aktual yang berbeda [24].

Tabel 2.5 *Confusion Matrix*

		Nilai Sebenarnya	
		Positif (1)	Negatif (0)
Nilai Prediksi	Positif (1)	TP	FP
	Negatif (0)	FN	TM

Penjelasan dari table tesebut adalah sebagai berikut.

1. *True Positive* (TP), ketika hasil prediksi dan nilai kebenarannya positif. Contohnya seorang wanita yang diprediksi halmil memang sedang hamil
2. *False Positive* (FP), ketika hasil prediksi positif sedangkan nilai kebenarannya negatif. Contohnya seorang lelaki diprediksi sedang hamil.
3. *False Negative* (FN), ketika hasil prediksi negatif sedangkan nilai kebenarannya positif. Contohnya ialah ketika seorang wanita diprediksi tidak hamil padahal sebenarnya dia sedang hamil.
4. *True Negative* (TN), ketika hasil prediksi dan nilai kebenarannya memiliki nilai negatif. Contohnya ialah lelaki diprediksi tidak hamil.

Nilai-nilai di atas dapat digunakan untuk mengukur tingkat validasi data. Beberapa jenis teknik validasi yang umum digunakan antara lain:

$$Recall = \frac{TP}{TP + FN} \tag{2.18}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.19}$$

$$F1\ score\ = \frac{2(precision * recall)}{(precision + recall)} \tag{2.20}$$

$$Akurasi = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.21}$$

2.10 Python

Python adalah bahasa pemrograman yang memiliki kode penulisan yang mudah dipahami, interaktif, dan berorientasi objek. Hal ini mencakup modul, *exception*, tipe data dinamis tingkat tinggi, dan kelas. Python juga daoat diekstensi ke Bahasa pemrograman lain, seperti C dan C++. Python sendiri dapat

berjalan di berbagai sistem operasi, seperti Unix, Mac, Linux, dan lain-lain [25].

2.11 Library

Library merupakan sekumpulan program yang dapat digunakan pada program lain tanpa terikat satu dengan yang lainnya. Terdapat beberapa library yang digunakan dalam melakukan implementasi tugas akhir ini. Library yang digunakan antara lain NLTK, Keras, Tensorflow, OpenCV, Numpy, dan Matplotlib.

2.11.1 Natural Language Toolkit (NLTK)

Natural Language Toolkit (NLTK) merupakan platform terkemuka yang dibangun pada program Python yang bekerja pada data bahasa. NLTK sendiri menyediakan lebih dari 50 arsip teks seperti Brown dan Gutenberg serta serangkaian *library* pemrosesan teks untuk klasifikasi, tokenisasi, stemming, dan kegiatan pemrosesan teks lainnya [3].

2.11.2 Keras

Keras merupakan *framework* terbaik yang digunakan dalam *deep learning*. Keras menyediakan API yang konsisten, sederhana, mudah dipelajari dan digunakan. Keras dapat dikembangkan pada berbagai platform, seperti TensorFlow dan Theano. Keras sendiri menyediakan banyak implementasi *neural network*, seperti fungsi aktivasi, optimizer dan *tools* lain yang dapat memudahkan dalam pengelolaan citra dan teks [26].

2.11.3 Tensorflow

TensorFlow merupakan *open source library* yang dimanfaatkan untuk untuk pembuatan program yang membutuhkan komputasi numerik berkinerja tinggi. TensorFlow dikembangkan oleh tim Google Brain. TensorFlow menyediakan fungsi-fungsi *machine learning* dan *deep learning*, dan dapat dijalankan dalam CPU ataupun GPU [27].

2.11.4 OpenCV

OpenCV (*Open Source Computer Vision*) adalah *library open source* yang menyediakan lebih dari 2.500 algoritma yang telah dioptimasi untuk kegiatan komputer visi dan *machine learning*. Beberapa algoritma tersebut seperti *feature & object detection, motion analysis and object tracking, image filtering, image processing*, dan lain-lain [28].

2.11.5 NumPy

NumPy adalah *library* Python yang mendukung pengolahan data pada *array* dan matriks multidimensi yang besar. NumPy menyediakan kumpulan fungsi matematika, seperti aljabar linear, transformasi Fourier, pembuatan angka acak, dan lain-lain. NumPy bersifat *open source* sehingga banyak dimanfaatkan dalam pengolahan data penelitian [29].

2.11.6 Scikit-learn

Scikit-learn merupakan *open source library* pada Bahasa Python yang berkembang untuk *machine learning*. Scikit-learn banyak menyediakan fitur terkait analisis data, seperti regresi, klasifikasi, *clustering*, dan juga menyediakan beberapa algoritma seperti *support vector machine*, *random forest*, dan *gradient boosting* [30].

2.11.7 Matplotlib

Matplotlib adalah *library* Python yang mendukung pembuatan grafik dua dimensi dalam berbagai format dan dari berbagai jenis data. Matplotlib bersifat *open source* dan banyak digunakan untuk pengolahan data dalam penelitian. Matplotlib dapat membuat plot, histogram, spektrum daya, diagram batang, diagram kesalahan, plot pencar, dan lain-lain [31].

BAB III
PERANCANGAN SISTEM

Bab ini menjelaskan tentang perancangan data dan sistem pengenalan tulisan tangan menggunakan *Fully Convolutional Network*. Bab ini juga akan menjelaskan gambaran umum sistem dalam bentuk diagram alir.

3.1 Perancangan Sistem

Sistem yang akan dibangun pada tugas akhir ini ditujukan untuk melakukan pengenalan teks pada citra kata tulisan tangan. Sistem ini sendiri memiliki tiga model yang memegang peranan penting pada sistem keseluruhan. Tiga sistem ini antara lain Lexicon CNN, Perhitungan Karakter CNN, dan Prediksi Karakter FCN.

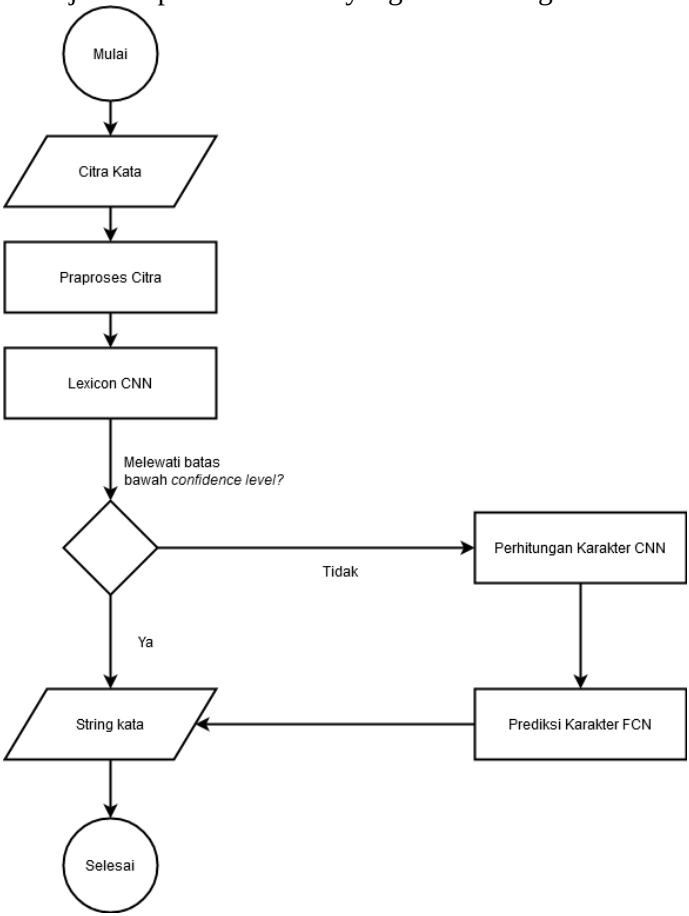
Sistem akan menerima input berupa citra kata. Input ini nantinya akan melalui tahapan praproses terlebih dahulu. Tahapan praproses ini pemotongan citra tepat pada objek kata. Sedangkan output pada akhir sistem ini ialaha *string* kata.

Setelah praproses, citra akan memasuki Lexicon CNN. Lexicon CNN bekerja mengenali kata-kata yang sering muncul pada suatu penulisan, seperti *"the"*, *"and"*, *"by"*, *"as"*, dan lain-lain. Lexicon CNN menerima citra dengan ukuran 32 x 128. Ketika hasil prediksi Lexicon CNN memiliki tingkat kepercayaan di atas Batasan yang ditentukan, hasil prediksi Lexicon CNN diakui sebagai keluaran sistem. Ketika tingkat kepercayaan ini tidak terpenuhi, maka citra kata akan diterus ke tahap Perhitungan Karakter CNN.

Perhitungan karakter CNN bekerja mengenali banyak karakter pada suatu citra. Pada sistem ini, terdapat 16 kelas yang digunakan untuk mengenali citra dengan banyka karakter satu hingga 16. Setelah sistem mengenali banyak karakter yang tersedia pada citra, citra akan diubah dulu ke dalam ukuran 32 x 16N.

Tahapan final yang dilalui citra ialah Prediksi Karakter FCN. Pada tahap ini, karakter demi karakter dalam citra akan dibaca oleh sistem. Pembacaan ini sendiri dilakukan menggunakan

sliding windows. Hasil prediksi dari Prediksi Karakter FCN ini akan menjadi output final sistem yang akan dibangun.



Gambar 3.1 Diagram Alir Sistem

3.2 Perancangan Data

Dalam Tugas Akhir ini, dataset yang akan digunakan terdapat dua jenis, yaitu dataset berupa teks dan dataset berupa gambar. Dataset teks sendiri digunakan untuk membangun kamus

kata. Kamus kata ini nantinya akan digunakan untuk membuat dataset citra yang digunakan untuk melatih model Lexicon CNN. Dataset citra yang digunakan diambil dari NIST SD19. Dari citra karakter yang dimiliki oleh NIST SD19, akan dibuat dataset citra kata sintesis dengan cara menggabungkan beberapa citra karakter. Dataset citra kata sintesis ini digunakan untuk melatih dua *neural network* pada tugas akhir ini, yakni Lexicon CNN dan Perhitungan Karakter CNN. Spesifikasi dataset untuk dua *neural network* ini dapat dilihat pada Tabel 3.1 dan

Tabel 3.2. Prediksi Karakter FCN hanya menggunakan sebagian data dari NIST SD19 dikarenakan kurang meratanya banyak data antar kelas. Spesifikasi Prediksi Karakter FCN dapat dilihat pada Tabel 3.3.

Sebelum dataset NIST SD19 digunakan, dataset ini terlebih dahulu akan dibersihkan. Pembersihan ini ditujukan untuk menghilangkan citra karakter yang tidak sesuai dengan kelasnya. Pada beberapa kelas terdapat kemunculan citra karakter yang ganjil, seperti citra karakter ‘i’ muncul pada kelas karakter ‘j’.

Tabel 3.1 Spesifikasi Dataset Citra Sintesis Lexicon CNN

Keterangan	**Spesifikasi**
Ukuran gambar	32 x 128 *pixels*
Jumlah gambar	59.000
Jumlah corpus	2
Jumlah kelas per corpus	59
Jumlah gambar per kelas	1.000
Ekstensi	.png
Ukuran file	160-1800 *bytes*
Kanal warna	1 (RGB)

Tabel 3.2 Spesifikasi Dataset Citra Sintesis Perhitungan Karakter CNN

Keterangan	**Spesifikasi**
Ukuran gambar	32 x 128 *pixels*
Jumlah gambar	16.000
Jumlah kelas	16
Jumlah gambar per kelas	1.000
Ekstensi	.png
Ukuran file	221 – 4700 *bytes*
Kanal warna	1 (RGB)

Tabel 3.3 Spesifikasi Dataset Citra Prediksi FCN

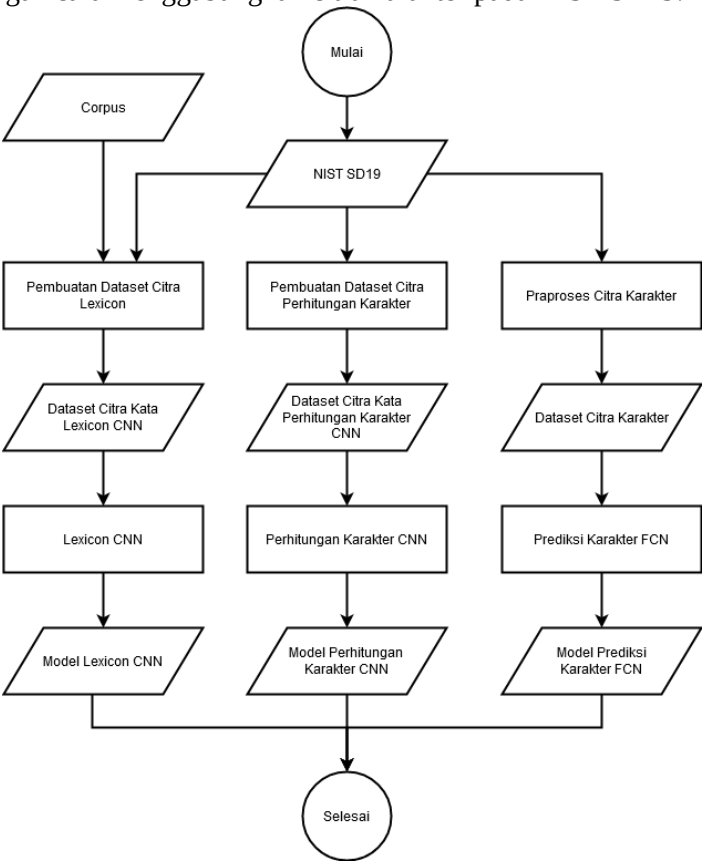
Keterangan	**Spesifikasi**
Ukuran gambar	128 x 128 *pixels*
Jumlah gambar	90,954
Jumlah kelas	62
Jumlah gambar per kelas	1.467
Ekstensi	.png
Ukuran file	400-700 *bytes*
Kanal warna	3 (RGB)

3.3 Desain Umum Sistem

Sistem pengenalan citra tulisan tangan yang dibangun ini memiliki tiga model *neural network* yang perlu melalui tahap pelatihan terlebih dahulu. Tiga model ini ialah Lexicon CNN, Perhitungan Karakter CNN, dan Prediksi Karakter FCN. Ketiga model ini memegang peranan penting pada jalannya sistem dan memiliki arsitektur *neural* yang berbeda-beda.

Lexicon CNN bekerja mengenali citra ke dalam kelas kata-kata yang sering muncul. Dalam pembuatan kamus kata yang paling sering muncul ini, digunakan dua corpus yaitu Brown dan Gutenberg. Sayangnya kumpulan kata yang terhimpun pada kamus ini tidak memiliki citra tulisan tangan. Ketidaktersediaan citra kata

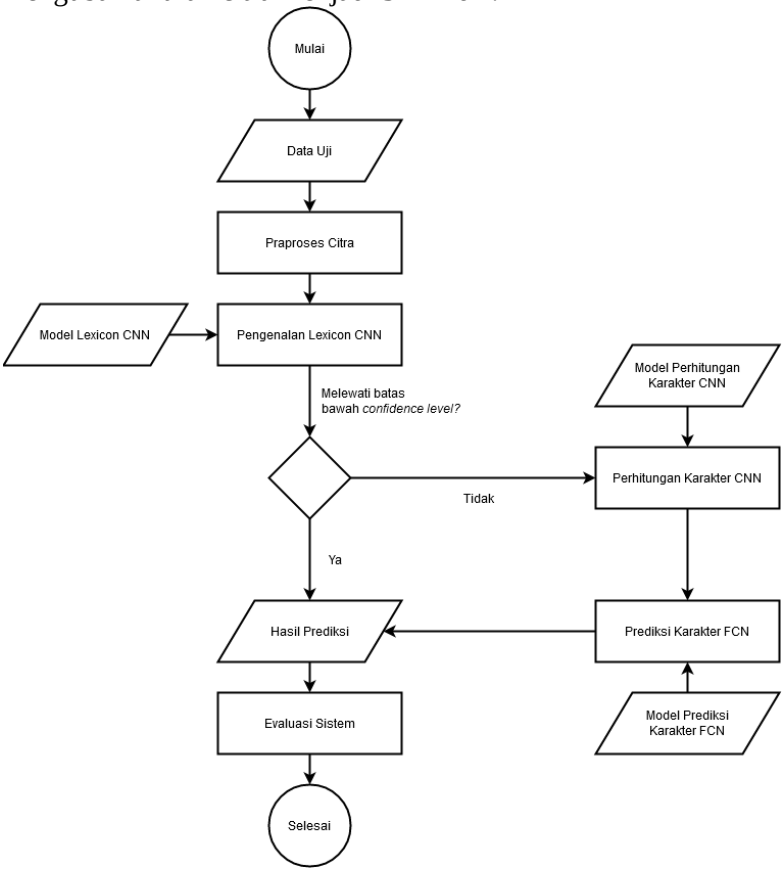
tulisan tangan ini menjadi penghalang besar dalam proses pelatihan arsitektur *neural*. Untuk mengatasi hal ini, dibuat citra kata sintesis dengan cara menggabungkan citra karakter pada NIST SD19.



Gambar 3.2 Diagram Alir Proses Pelatihan Sistem

Hasil prediksi Lexicon CNN akan diakui oleh sistem ketika memiliki tingkat kepercayaan melebihi batas yang ditentukan. Ketika hasil prediksi tidak mampu mencapai angka tersebut, citra kata akan diteruskan ke Perhitungan Karakter CNN. Sama seperti pada Lexicon CNN, data latih yang akan digunakan

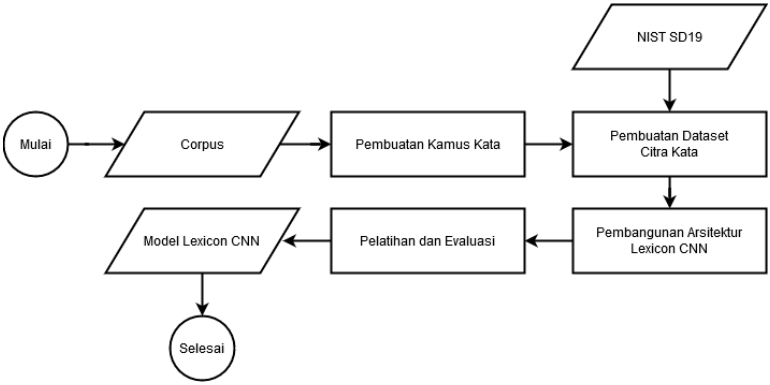
dalam melatih arsitektur *neural* merupakan citra kata sintesis dari NIST SD19. Citra kata ini dibuat untuk memiliki banyak karakter tertentu. Pada tugas akhir ini, Perhitungan Karakter CNN akan memiliki 16 kelas yang merepresentasikan kata dengan banyak karakter satu hingga 16. Hasil prediksi Perhitungan Karakter CNN berupa banyak karakter (N) pada citra akan digunakna untuk mengubah ukuran citra menjadi 32 x 16N.



Gambar 3.3 Diagram Alir Proses Pengujian Sistem

3.3.1 Tahap Pembuatan Model Lexicon CNN

Pada tugas akhir ini, Lexicon CNN merupakan arsitektur yang digunakan untuk mengenali suatu citra kata berdasarkan suatu kamus yang sebelumnya telah dibuat. Hasil prediksi Lexicon CNN hanya akan diterima ketika memiliki tingkat kepercayaan yang ditentukan.



Gambar 3.4 Diagram Alir Pembuatan Lexicon CNN

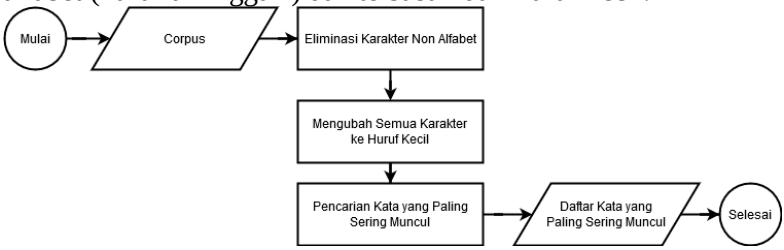
Seperti yang dapat dilihat pada Gambar 3.4, pembuatan model Lexicon CNN terbagi menjadi empat tahap utama, yaitu tahap pembuatan kamus kata, tahap pembuatan dataset citra kata, tahap pembangunan arsitektur Lexicon CNN, dan tahap pelatihan dan evaluasi.

Arsitektur *neural* Lexicon CNN akan mengenali 50 kelas kata yang sering muncul. Sistem akan menjadi lebih baik apabila memiliki lebih banyak kelas kata. Namun karena keterbatasan dari segi *hardware*, penggunaan 50 kelas kata ini dirasa cukup.

3.2.1.1 Tahap Pembuatan Kamus Kata

Pada tahap ini akan digunakan dua *corpus* yang tersedia pada NLTK, yakni Brown dan Gutenberg. Dari masing-masing *corpus* ini, akan diambil 50 kata yang memiliki frekuensi kemunculan tertinggi. Ketika dua daftar kata ini digabung, akan

terdapat 59 kata bahasa Inggris berbeda. dapun spesifikasi yang harus dimiliki kata-kata ini ialah hanya terbentuk dari karakter alfabet (huruf a hingga z) dan tersusun dari huruf kecil.



Gambar 3.5 Diagram Alir Pembuatan Kamus Kata

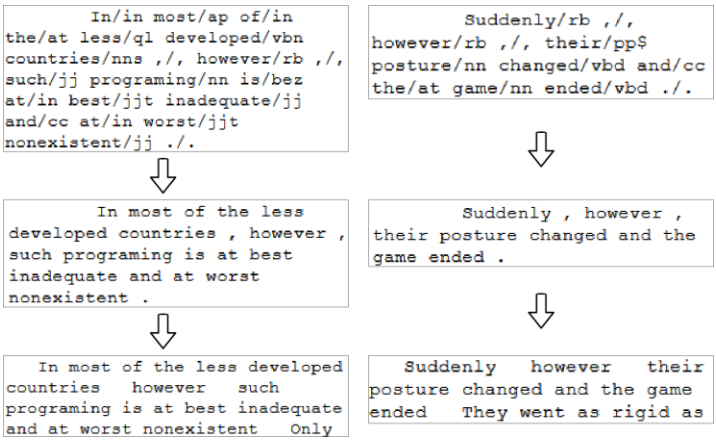
Angka dan karakter lainnya, seperti tanda titik, koma, tanda tanya, atau tanda seru, akan dieliminasi terlebih dahulu menggunakan *regular expression*. Setelah eliminasi, kata-kata yang tersisa akan diubah menjadi huruf kecil. Proses berikutnya ialah dilakukan pemeringkatan kata berdasarkan frekuensi kemunculan. Gambar 3.5 menampilkan diagram alir dari proses pembuatan kamus kata.

Dalam tahap eliminasi symbol sendiri, perlakuan yang diberikan pada *corpus* Brown dan Gutenberg berbeda. Perlakuan yang berbeda ini disebabkan susunan teks yang dimiliki oleh dua *corpus* ini berbeda jauh.

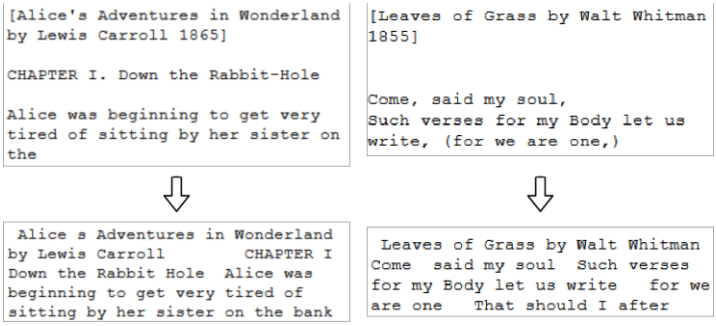
Pada Brown, setiap kata yang tertulis selalu diikuti dengan karakter '/' atau garis miring yang kemudian diikuti beberapa karakter. Karena susunan teks yang seperti ini, *regular expression* dilakukan sebanyak dua kali. *Regular expression* yang pertama digunakan untuk menghapus semua garis miring besar karakter yang muncul setelah garis miring. *Regular expression* yang kedua digunakan untuk membuang semua karakter selain huruf alfabet. Contoh proses eliminasi ini dapat dilihat pada Gambar 3.6.

Pada Gutenberg, teks yang dimiliki ditulis seperti teks pada umumnya, sehingga *regular expression* hanya dilakukan satu kali. *Regular Expression* ini digunakan untuk menghapus semua

karakter selain alfabet. Proses eliminasi karakter ini dapat dilihat pada Gambar 3.7.



Gambar 3.6 Tahapan Eliminasi Karakter pada Brown



Gambar 3.7 Tahapan Eliminasi Karakter pada Gutenberg

Lima puluh kata yang paling sering muncul padai *corpus* Brown dapat dilihat pada Lampiran A, sedangkan untuk *corpus* Gutenberg dapat dilihat pada Lampiran B. Selanjutnya untuk gabungan daftar kata tersebut, dapat dilihat pada

Tabel 3.4.

Tabel 3.4 Daftar Kata yang Digunakan untuk Lexicon

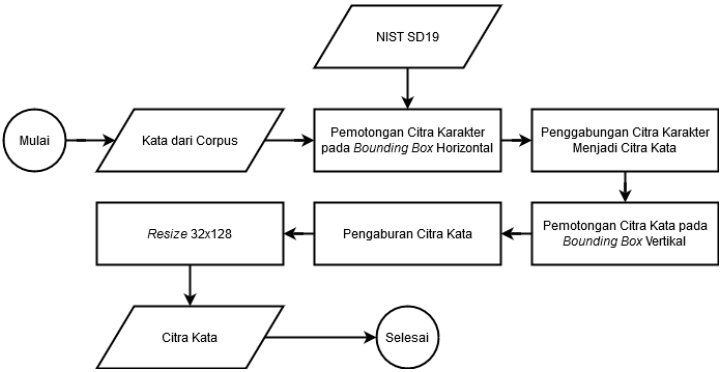
No	Kata
1	the
2	of
3	and
4	to
5	a
6	in
7	that
8	is
9	was
10	he
11	for
12	it
13	with
14	as
15	his
16	on
17	be
18	at
19	by
20	i

No	Kata
21	this
22	had
23	not
24	are
25	but
26	from
27	or
28	have
29	an
30	they
31	which
32	one
33	were
34	you
35	all
36	her
37	she
38	there
39	would
40	their

No	Kata
41	we
42	him
43	been
44	has
45	when
46	who
47	will
48	no
49	more
50	if
51	me
52	thou
53	unto
54	shall
55	said
56	so
57	lord
58	them
59	my

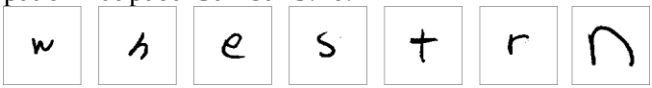
3.2.1.2 Tahap Pembuatan Dataset Citra Lexicon CNN

Pembuatan dataset citra kata ini ditujukan untuk membuat citra data latih. Pada tahapan pembuatan, dibuat 1.000 citra untuk masing-masing kelas pada masing-masing *corpus* yang digunakan. Gambar 3.8 menampilan diagram alir pembuatan citr Lexicon CNN.

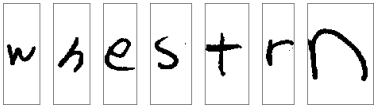


Gambar 3.8 Diagram Alir Pembuatan Citra Lexicon CNN

Tahapan pembuatan citra ini dimulai saat sistem menerima kata dari corpus. Tiap karakter pada kata ini akan diambil satu citra karakter dari NIST SD19 secara acak. Conntoh citra karakter pada NIST SD19 dapat dilihat pada Gambar 3.9. Kumpulan citra dari tiap karakter ini digabungkan menjadi satu secara horizontal. Sebelum melakukan penggabungan, citra karakter akan dipotong dulu berdasarkan *bounding box* horizontal. Citra akan memiliki tinggi yang sama, tetapi secara lebar citra akan dipotong tepat pada karakter. Pencarian *bounding box* dilakukan menggunakna deteksi kontur. Pada pemotongan *bounding box* horizontal ini, akan diberi jarak secara acak pada sisi kiri kanan karakter untuk memberikan variasi. Hasil pemotongan karakter pada *bounding box* horizontal ini dapat dilihat pada Gambar 3.10.

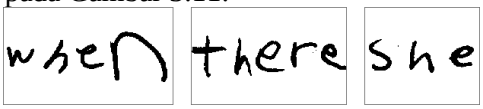


Gambar 3.9 Contoh Citra Karakter NIST SD19



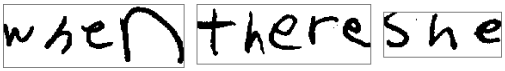
Gambar 3.10 Hasil Pemotongan *Bounding Box* Horizontal

Setelah melalui pemotongan pada *bounding box horizontal*, citra karakter tersebut akan digabungkan secara horizontal menjadi citra kata. Hasil penggabungan karakter ini dapat dilihat pada Gambar 3.11.



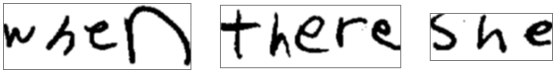
Gambar 3.11 Hasil Penggabungan Karakter

Pada citra kata ini, citra akan dipotong lagi. Pemotongan kali ini berdasarkan *bounding box* vertikal. Sama seperti sebelumnya, pencarian *bounding box* ini dilakukan menggunakan deteksi kontur. Pada pemotongan ini, akan diberi jarak secara acak pada sisi atas bawah kata untuk memberikan variasi. Hasil pemotongan citra kata pada *bounding box* vertikal ini dapat dilihat pada Gambar 3.12.



Gambar 3.12 Hasil Pemotongan *Bounding Box* Vertikal

Terakhir, citra kata akan diberikan efek pengaburan. Pengaburan yang diberikan pada citra merupakan Gaussian *blur*. Hasil pengaburan citra ini dapat dilihat pada Gambar 3.13. Sebelum disimpan, citra akan diubah ukurannya dulu ke 32x128. Pengubahan ukuran ini disesuaikan dengan masukan pada kebutuhan arsitektur *neural network*.



Gambar 3.13 Hasil Pengaburan Citra

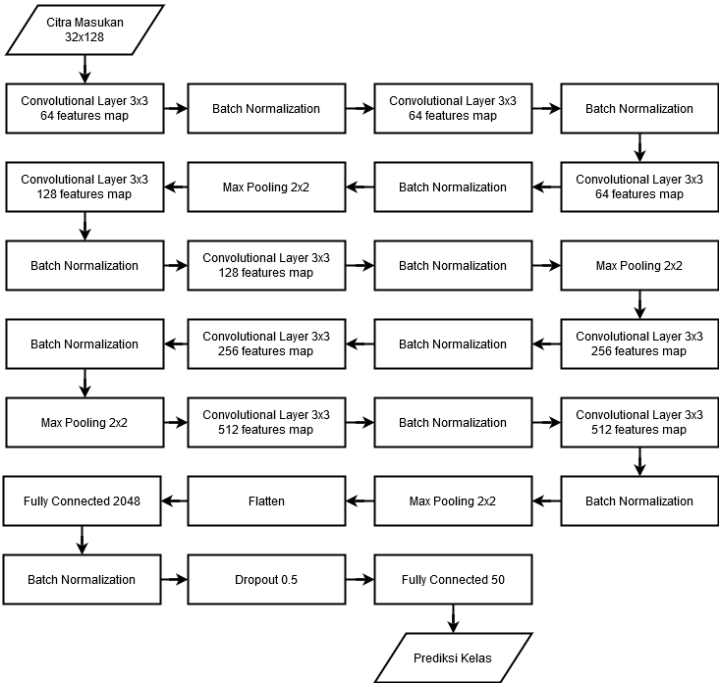
3.2.1.3 Tahap Pembangunan Arsitektur Lexicon CNN

Pembangunan arsitektur bertujuan untuk menyiapkan *layer*, fungsi aktivasi, *loss function*, dan parameter-parameter lain yang dibutuhkan. Lexicon CNN yang akan dibangun ini, terdiri dari *convolutional layer*, *batch normalization layer*, *max pooling*

layer, *fully connected layer*, dan *dropout layer*. Pembangunan arsitektur ini mengikuti arsitektur yang dibangun pada [1] dengan detail yang dapat dilihat pada Gambar 3.14 dan Tabel 3.5.

Berikut detail dari arsitektur Lexicon CNN:

1. Input untuk arsitektur Lexicon CNN berupa citra kata yang berukuran 32x128 *pixels* dengan satu kanal warna.
2. Citra masukan akan melalui 9 *convolutional layer*. Tiga *convolutional layer* memiliki *filter* dengan ukuran 64, dua *convolutional layer* memiliki *filter* 128, lalu dua *convolutional layer* memiliki *filter* 256, dan dua *convolutional layer* memiliki *filter 512*. Sembilan *layer* ini memiliki *kernel* 3x3, *stride* 1x1, dan fungsi aktivasi ReLu. Sembilan *layer* ini menggunakan *padding* bernilai *same*.
3. Citra masukan akan melalui 4 *maxpooling layer* yang memiliki *kernel* 2x2 dan *stride* 2x2.
4. Setelah melalui sembilan *convolutional layer* dan empat *maxpooling layer*, citra masukan akan melalui *flatten* untuk mengubah matriks menjadi vektor fitur.
5. Vektor fitur yang terbuat akan melalui *fully connected layer* dengan fungsi ReLU.
6. Setelahnya, vektor fitur akan melalui *dropout layer* dengan nilai probabilitas 0,5.
7. Terakhir, vektor fitur akan melalui *fully connected layer* kedua dengan fungsi aktivasi softmax.
8. *Batch normalization layer* selalu dipasang setelah *convolutional layer* yang ada pada arsitektur dan setelah *fully connected layer* yang menggunakan fungsi aktivasi ReLU.



Gambar 3.14 Arsitektur Lexicon CNN

Tabel 3.5 Arsitektur Lexicon CNN

Layer	***Input***	***Output***	**Spesifikasi**
Input Layer	(32,128,1)		-
Convolutional Layer 1	(32,128,1)	(32,128,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 1	(32,128,64)	(32,128,64)	-
Convolutional Layer 2	(32,128,64)	(32,128,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 2	(32,128,64)	(32,128,64)	-

Convolutional Layer 3	(32,128,64)	(32,128,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 3	(32,128,64)	(32,128,64)	-
Max Pooling Layer 1	(32,128,64)	(16,64,64)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 4	(16,64,64)	(16,64,128)	*Filter* : 128 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 4	(16,64,128)	(16,64,128)	-
Convolutional Layer 5	(16,64,128)	(16,64,128)	*Filter* : 128 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 5	(16,64,128)	(16,64,128)	-
Max Pooling Layer 2	(16,64,128)	(8,32,128)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 6	(8,32,128)	(8,32,256)	*Filter* : 256 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 6	(8,32,256)	(8,32,256)	-
Convolutional Layer 7	(8,32,256)	(8,32,256)	*Filter* : 256 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 7	(8,32,256)	(8,32,256)	-
Max Pooling Layer 3	(8,32,256)	(4,16,256)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 8	(4,16,256)	(4,16,512)	*Filter* : 512 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 8	(4,16,512)	(4,16,512)	-
Convolutional Layer 9	(4,16,512)	(4,16,512)	*Filter* : 512

			Kernel : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 9	(4,16,512)	(4,16,512)	-
Max Pooling Layer 4	(4,16,512)	(2,8,512)	*Kernel* : 2x2 *Stride* : 2x2
Flatten	(2,8,512)	(8192)	-
Fully Connected Layer 1	(8192)	(2048)	Fungsi : ReLU
Batch Normalization Layer 10	(2048)	(2048)	-
Dropout Layer	(2048)	(2048)	*Dropout :0,5*
Fully Connected Layer 2	(2048)	(50)	Fungsi : Softmax

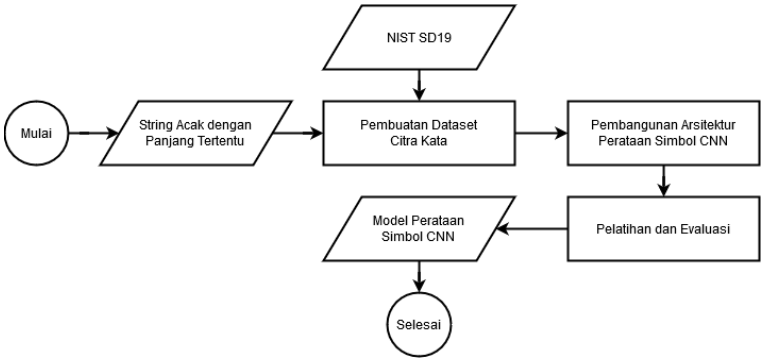
3.2.1.4 Tahap Pelatihan dan Evaluasi

Pelatihan Lexicon CNN dlilakukan dengan menggunakan citra kata sintesis sebanyak 59.000, epoch = 30, dan batch size = 64. Citra pelatihan tersebut dibagi sebanyak 90% data latih dan 10% data validasi. *Learning rate*, regulator, dan *optimizer* yang digunakan akan menyesuaikan skenario uji coba.

Akan dilakukan pengujian yang hasilnya akan dievaluasi menggunakan evaluasi CER, akurasi, presisi, *recall*, dan *f1-score*. Selain itu juga sistem akan dievaluasi menggunakna CER. Pengujian ini dilakukan pada data validasi.

3.3.2 Tahap Pembuatan Model Perhitungan Karakter CNN

Perhitungan Karakter CNN merupakan proses lanjutan setelah Lexicon CNN gagal mengenali kata. Suatu citra akan masuk ke dalam proses Perhitungan Karakter CNN ketika hasil prediksi Lexicon CNN memiliki tingkat kepercayaan di bawah nilai yang ditentukan. Perhitungan Karakter CNN bekerja untuk mengenali banyak karakter pada suatu citra kata. Hasil prediksi banyak kata ini akan membantu Prediksi Karakter FCN dalam mengenali karakter pada citra.



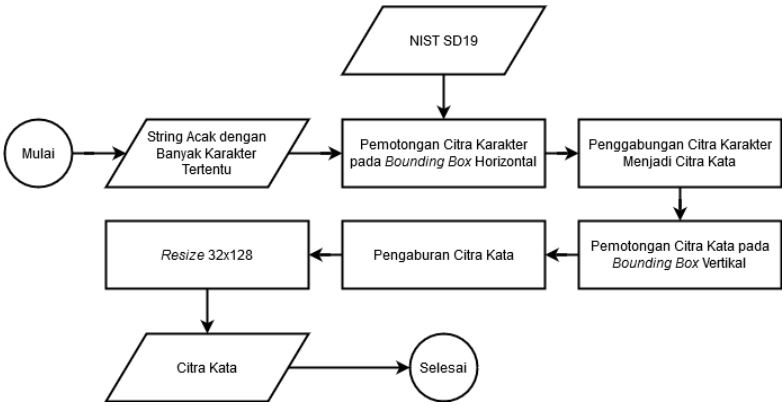
Gambar 3.15 Diagram Alir Pembuatan Model Perhitungan Karakter CNN

Perhitungan Karakter CNN memiliki 16 kelas, yang merepresentasikan banyak karakter dari satu hingga 16 pada suatu citra. Sama seperti Lexicon CNN, dalam pelatihan akan menggunakan citra kata sintesis. Pembuatan model Perhitungan Karakter CNN terdiri atas tiga tahapan, yakni tahap pembuatan dataset citra kata, tahap pembangunan arsitektur Perhitungan Karakter CNN, dan diakhiri dengan pelatihan dan evaluasi. Diagram alir pembuatan Perhitungan Karakter CNN dapat dilihat pada Gambar 3.15.

3.3.2.1 Tahap Pembuatan Dataset Citra Perhitungan Karakter CNN

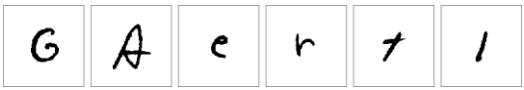
Pembuatan dataset citra kata Perhitungan Karakter CNN kurang lebih memiliki tahapan yang sama seperti pada Lexicon CNN. Pada tahap ini dibuat 1.000 citra pada masing-masing kelas. Banyak kelas pada Perhitunga Karakter FCN sendiri ialah 16. Perbedaan pembuatan dataset pada Perhitungan Karakter FCN dengan Lexicon CNN hanyalah pada masukan sistem. Pada Lexicon CNN, masukan yang dibutuhkan adalah kata yang didapat dari pencarian kata paling sering muncul. Pada Perhitungan Karakter CNN, masukan yang dibutuhkan ialah kata dengan panjang karakter tertentu. Pembuatan kata ini sendiri ialah hanya pengambilan karakter dari 62 kelas karakter pada Prediksi Karakter

FCN yang kemudian di rangkai menjadi kata. Pembuatan dataset ini dapat dilihat pada diagram alir Gambar 3.16.

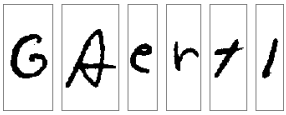


Gambar 3.16 Diagram Alir Pembuatan Dataset Citra Perhitungan Karakter CNN

Tahap pembuatan citra ini dimulai saat sistem menerima rangkaian karakter dengan panjang tertentu. Conntoh citra karakter pada NIST SD19 dapat dilihat pada Gambar 3.17 . Kumpulan citra dari tiap karakter ini digabungkan menjadi satu secara horizontal. Sebelum melakukan penggabungan, citra karakter akan dipotong dulu berdasarkan *bounding box* horizontal. Citra akan memiliki tinggi yang sama, tetapi secara lebar citra akan dipotong tepat pada karakter. Pencarian *bounding box* dilakukan menggunakna deteksi kontur. Pada pemotongan *bounding box* horizontal ini, akan diberi jarak secara acak pada sisi kiri kanan karakter untuk memberikan variasi. Hasil pemotongan karakter pada *bounding box* horizontal ini dapat dilihat pada Gambar 3.18.

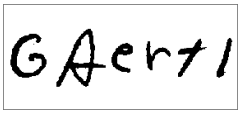


Gambar 3.17 Contoh Citra Karakter NIST SD19



Gambar 3.18 Hasil Pemotongan *Bounding Box* Horizontal

Setelah melalui pemotongan pada *bounding box horizontal*, citra karakter tersebut akan digabungkan secara horizontal menjadi citra kata. Hasil penggabungan karakter ini dapat dilihat pada Gambar 3.19.

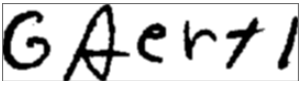


Gambar 3.19 Hasil Penggabungan Karakter

Pada citra kata ini, citra akan dipotong lagi. Pemotongan kali ini berdasarkan *bounding box* vertikal. Sama seperti sebelumnya, pencarian *bounding box* ini dilakukan menggunakan deteksi kontur. Pada pemotongan ini, akan diberi jarak secara acak pada sisi atas bawah kata untuk memberikan variasi. Hasil pemotongan citra kata pada *bounding box* vertikal ini dapat dilihat pada Gambar 3.20.

Gambar 3.20 Hasil Pemotongan *Bounding Box* Vertikal

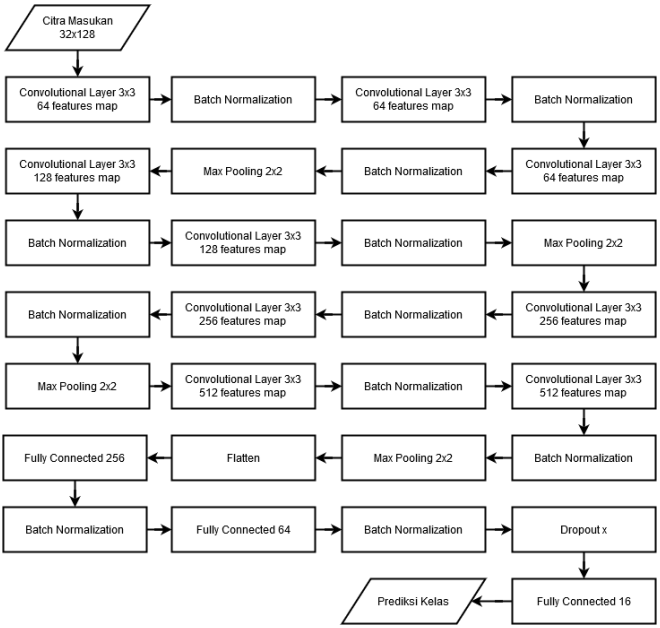
Terakhir, citra kata akan diberikan efek pengaburan. Pengaburan yang diberikan pada citra merupakan Gaussian *blur*. Hasil pengaburan citra ini dapat dilihat pada Gambar 3.21. Sebelum disimpan, citra akan diubah ukurannya dulu ke 32x128. Pengubahan ukuran ini disesuaikan dengan masukan pada kebutuhan arsitektur *neural*.



Gambar 3.21 Hasil Pengaburan Citra

3.3.2.2 Tahap Pembangunan Arsitektur Perhitungan Karakter CNN

Pembangunan Arsitektur bertujuan untuk menyiapkan layer, fungsi aktivasi, *loss function*, dan parameter-parameter lain yang dibutuhkan. Lexicon CNN yang akan dibangun ini, terdiri dari *convolutional layer*, *batch normalization layer*, *max pooling layer*, *fully connected layer*, dan *dropout layer*. Pembangunan arsitektur ini memiliki detail yang dapat dilihat pada Gambar 3.22 dan Tabel 3.6.



Gambar 3.22 Arsitektur Perhitungan Karakter CNN

Berikut detail dari arsitektur Perhitungan Karakter CNN:

1. Input untuk arsitektur Perhitungan Karakter CNN berupa citra kata yang berukuran 32x128 *pixels* dengan satu kanal warna.
2. Citra masukan akan melalui 9 *convolutional layer*. Tiga *convolutional layer* memiliki *filter* dengan ukuran 64, dilanjut dua *convolutional layer* memiliki *filter* 128, lalu dua

convolutional layer yang memiliki *filter* 256, dan dua *convolutional layer* memiliki *filter* 512. Sembilan *layer* ini memiliki *kernel* 3x3, *stride* 1x1, dan fungsi aktivasi maxout. Sembilan *layer* ini memiliki *padding* bernilai *same*.

3. Citra masukan akan melalui 4 *maxpooling layer* yang memiliki *kernel* 2x2 dan *stride* 2x2.
4. Vektor fitur yang terbuat akan melalui dua *fully connected layer* dengan fungsi ReLU dengan banyak *neuron* berbeda.
5. Setelahnya, fitur akan melalui *dropout layer* dengan nilai probabilitas x, dimana x menyesuaikan uji skenario.
6. Terakhir, fitur akan melalui *fully connected layer* kedua dengan fungsi aktivasi softmax.
7. *Batch normalization layer* selalu dipasang setelah *convolutional layer* yang ada pada arsitektur dan setelah *fully connected layer* yang menggunakan fungsi aktivasi ReLU.

Tabel 3.6 Arsitektur Perhitungan Karakter CNN

Layer	***Input***	***Output***	**Spesifikasi**
Input Layer	(32,128,1)		-
Convolutional Layer 1	(32,128,1)	(32,128,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 1	(32,128,64)	(32,128,64)	-
Convolutional Layer 2	(32,128,64)	(32,128,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 2	(32,128,64)	(32,128,64)	-
Convolutional Layer 3	(32,128,64)	(32,128,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 3	(32,128,64)	(32,128,64)	-
Max Pooling Layer 1	(32,128,64)	(16,64,64)	*Kernel* : 2x2 *Stride* : 2x2

Convolutional Layer 4	(16,64,64)	(16,64,128)	*Filter* : 128 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 4	(16,64,128)	(16,64,128)	-
Convolutional Layer 5	(16,64,128)	(16,64,128)	*Filter* : 128 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 5	(16,64,128)	(16,64,128)	-
Max Pooling Layer 2	(16,64,128)	(8,32,128)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 6	(8,32,128)	(8,32,256)	*Filter* : 256 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 6	(8,32,256)	(8,32,256)	-
Convolutional Layer 7	(8,32,256)	(8,32,256)	*Filter* : 256 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 7	(8,32,256)	(8,32,256)	-
Max Pooling Layer 3	(8,32,256)	(4,16,256)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 8	(4,16,256)	(4,16,512)	*Filter* : 512 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 8	(4,16,512)	(4,16,512)	-
Convolutional Layer 9	(4,16,512)	(4,16,512)	*Filter* : 512 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : Maxout *Padding* : *Same*
Batch Normalization Layer 9	(4,16,512)	(4,16,512)	-
Max Pooling Layer 4	(4,16,512)	(2,8,512)	*Kernel* : 2x2 *Stride* : 2x2
Flatten	(2,8,512)	(8192)	-

Fully Connected Layer 1	(8192)	(256)	Fungsi : ReLU
Batch Normalization Layer 10	(256)	(256)	-
Fully Connected Layer 2	(256)	(64)	Fungsi : ReLU
Batch Normalization Layer 11	(64)	(64)	-
Dropout Layer	(64)	(64)	*Dropout : x*
Fully Connected Layer 2	(64)	(16)	Fungsi : Softmax

3.3.2.3 Tahap Pelatihan dan Evaluasi

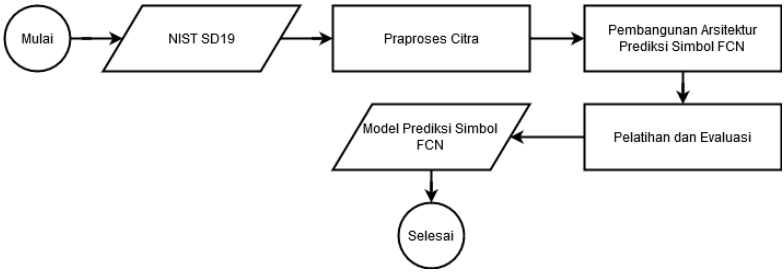
Pelatihan Perhitungan Karakter CNN dlilakukan dengan menggunakan citra kata sintesis sebanyak 16.000 dan batch size = 32. Citra pelatihan tersebut dibagi sebanyak 90% data latih dan 10% data validasi. *Learning rate*, regulator, *dropout*, dan *optimizer* yang digunakan akan menyesuaikan skenario uji coba.

Akan dilakukan pengujian yang hasilknya akan dievaluasi menggunakan evaluasi akurasi, presisi, *recall*, dan *f1-score*. Pengujian ini dilakukan pada data validasi.

3.3.3 Tahap Pembuatan Model Prediksi Karakter FCN

Prediksi Karakter FCN merupakan inti utama dari sistem pengenalan tulisan tangan yang akan dibuat. Pada Perhitungan Karakter CNN, akan didapat prediksi banyak karakter (N) pada suatu citra. Citra kemudian diubah ukurannya menjadi 32 x 16N sebelum memasuki Prediksi Karakter FCN. Pada proses pengenalannya sendiri, Prediksi Karakter FCN menggunakan metode *sliding window*.

Prediksi Karakter FCN memiliki 62 kelas, yang terdiri dari huruf alfabet baik huruf kapital maupun huruf kecil beserta angka. Pembuatan model Prediksi Karakter FCN ini menggunakan NIST SD19 sebagai data latih dan pada masing-masing kelas hanya diambil 1.467 buah citra karena ketidakseimbangan data antar kelas yang dimiliki.

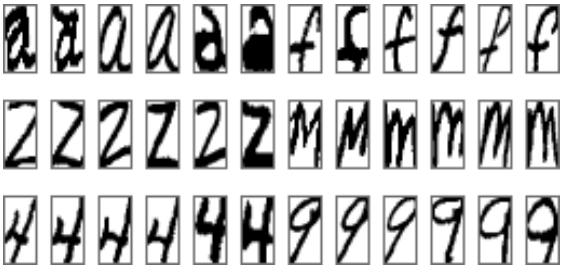


Gambar 3.23 Diagram Alir Pembuatan Model Prediksi Karakter FCN

Pembuatan model ini sendiri terdiri dari 3 tahapan, yakni tahap praproses citra data latih, tahap pembangunan arsitektur Prediksi Karakter FCN, serta tahap pelatihan dan evaluasi. Diagram alir pembuatan model Prediksi Karakter FCN dapat dilihat pada Gambar 3.23.

3.3.3.1 Tahap Praproses Citra Data Latih

Sebelum memasuki tahap pelatihan, citra karakter NIST SD19 akan melalui tahapan praproses terlebih dahulu. Praproses ini ditujukan untuk melakukan pemotongan pada citra karakter tepat pada *bounding box* karakter. Untuk melakukan pemotongan ini, digunakan deteksi kontur. Hasil pemotongan dapat dilihat pada Gambar 3.24.



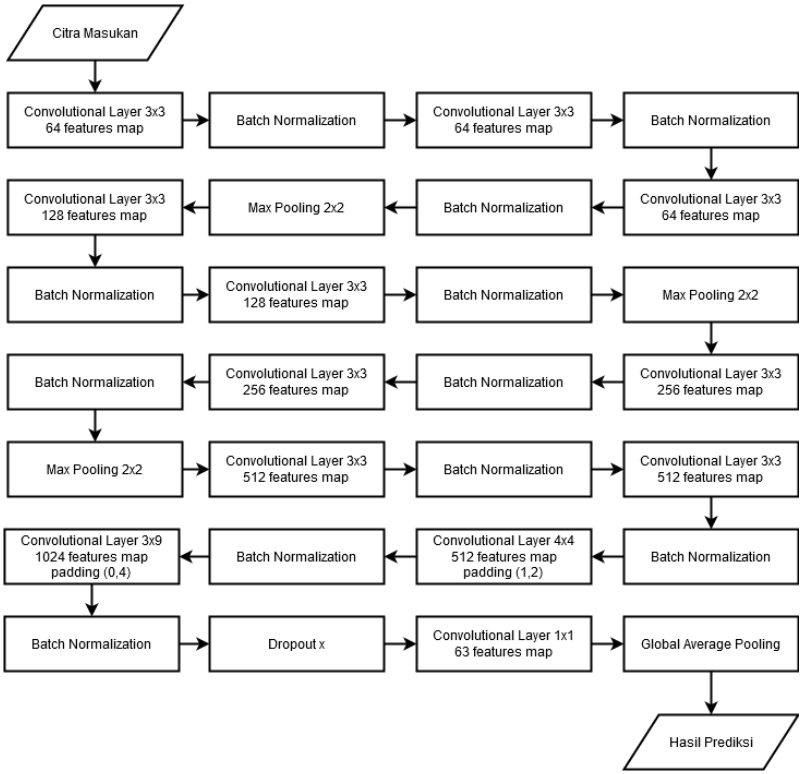
Gambar 3.24 Hasil Pemotongan Citra NIST SD19

3.3.3.2 Tahap Pembangunan Arsitektur Prediksi Karakter FCN

Pembangunan arsitektur bertujuan untuk menyiapkan *layer*, fungsi aktivasi, *loss function*, dan parameter-parameter lain yang dibutuhkan. Prediksi Karakter FCN yang akan dibangun ini, terdiri dari *convolutional layer*, *batch normalization layer*, *max pooling layer*, dan *dropout layer*. Pembangunan arsitektur ini mengikuti arsitektur yang dibangun pada [1] dengan detail yang dapat dilihat pada Gambar 3.25 dan Tabel 3.7.

Berikut detail dari arsitektur Prediksi Karakter FCN:

1. Input untuk arsitektur Prediksi Karakter FCN berupa citra kata yang berukuran lebar 32 dengan satu kanal warna.
2. Citra masukan akan melalui 9 *convolutional layer* awal. Tiga *convolutional layer* memiliki *filter* dengan ukuran 64, dilanjut dua *convolutional layer* memiliki *filter* 128, lalu dua *convolutional layer* yang memiliki *filter* 256, dan dua *convolutional layer* memiliki *filter* 512. Sembilan *layer* ini memiliki *kernel* 3x3, *stride* 1x1, dan fungsi aktivasi maxout. Sembilan *layer* ini memiliki *padding* bernilai *same.*
3. Setelahnya akan melalui *dropout layer* dengan nilai probabilitas x, dimana x menyesuaikan uji skenario.
4. Berbeda dengan CNN, FCN tetap menggunakan *convolutional layer* pada bagian prediksi. Bagian prediksi untuk arsitektur Prediksi Karakter FCN terdiri dari 3 *convolutional layer*. Tiga *convolutional layer* ini memiliki filter, padding, dan kernel yang berbeda. Pada *convolutional layer* terakhir digunakan filter jumlah kelas ditambah satu, dimana satu kelas terakhir ini dianggap sebagai *blank prediction*.
5. Setelah melalui semua *convolutional layer*, akan melalui layer *global average pooling*, yang kemudian akan menjadi masukan pada fungsi aktivasi *softmax*.
6. *Batch normalization layer* selalu dipasang setelah *convolutional layer* yang ada pada arsitektur.



Gambar 3.25 Arsitektur Prediksi Karakter FCN

Tabel 3.7 Arsitektur Prediksi Karakter FCN

Layer	***Input***	***Output***	**Spesifikasi**
Input Layer	(32,None,1)		-
Convolutional Layer 1	(32,None,1)	(32,None,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 1	(32,None,64)	(32,None,64)	-
Convolutional Layer 2	(32,None,64)	(32,None,64)	*Filter* : 64 *Kernel* : 3x3

			Stride : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 2	(32,None,64)	(32,None,64)	-
Convolutional Layer 3	(32,None,64)	(32,None,64)	*Filter* : 64 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 3	(32,None,64)	(32,None,64)	-
Max Pooling Layer 1	(32,None,64)	(16,None,64)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 4	(16,None,64)	(16,None,128)	*Filter* : 128 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 4	(16,None,128)	(16,None,128)	-
Convolutional Layer 5	(16,None,128)	(16,None,128)	*Filter* : 128 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 5	(16,None,128)	(16,None,128)	-
Max Pooling Layer 2	(16,None,128)	(8,None,128)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 6	(8,None,128)	(8,None,256)	*Filter* : 256 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 6	(8,None,256)	(8,32,256)	-
Convolutional Layer 7	(8,None,256)	(8,None,256)	*Filter* : 256 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*

Batch Normalization Layer 7	(8,None,256)	(8,None,256)	-
Max Pooling Layer 3	(8,None,256)	(4,None,256)	*Kernel* : 2x2 *Stride* : 2x2
Convolutional Layer 8	(4,None,256)	(4,None,512)	*Filter* : 512 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 8	(4,None,512)	(4,None,512)	-
Convolutional Layer 9	(4,None,512)	(4,None,512)	*Filter* : 512 *Kernel* : 3x3 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Same*
Batch Normalization Layer 9	(4,None,512)	(4,None,512)	-
Dropout	(1,None,1024)	(1,None,1024)	-
Convolutional Layer 10	(4,None,512)	(3,None,512)	*Filter* : 512 *Kernel* : 4x4 *Stride* : 1x1 Fungsi : ReLU *Padding* : (1,2)
Batch Normalization Layer 10	(3,None,512)	(3,None,512)	-
Convolutional Layer 11	(3,None,512)	(1,None,1024)	*Filter* : 1024 *Kernel* : 3x9 *Stride* : 1x1 Fungsi : ReLU *Padding* : (0,4)
Batch Normalization Layer 11	(1,None,1024)	(1,None,1024)	-
Convolutional Layer 12	(1,None,1024)	(1,None,63)	*Filter* : 63 *Kernel* : 1x1 *Stride* : 1x1 Fungsi : ReLU *Padding* : *Valid*
Global Average Pooling	(1,None,63)	(63)	
Activation	(63)	(63)	Fungsi: Softmax

3.3.3.3 Tahap Pelatihan dan Evaluasi

Pelatihan Prediksi Karakter FCN dilakukan dengan menggunakan citra karakter sebanyak 90,954, epoch = 50, dan batch size = 64. Citra pelatihan tersebut dibagi sebanyak 90% data latih dan 10% data validasi. *Learning rate*, regulator, *dropout*, dan *optimizer* yang digunakan akan menyesuaikan skenario uji coba.

Akan dilakukan pengujian yang hasilknya akan dievaluasi menggunakan variabel evaluasi akurasi, presisi, *recall*, dan *f1-score*. Selain itu juga sistem akan dievaluasi menggunakna CER. Pengujian ini dilakukan menggunakan data validasi.

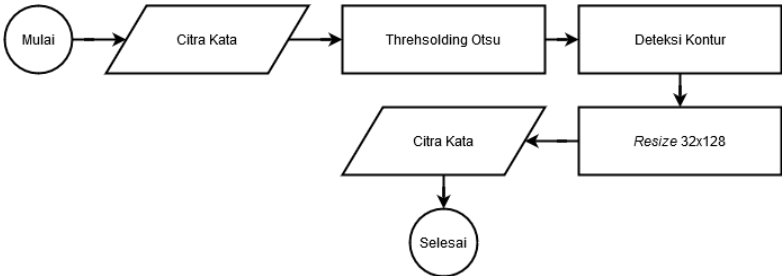
3.3.4 Tahap Pengujian Sistem

Pada subbab ini akan membahas tahapan yang dilakukan selama evaluasi sistem. Pada tahap ini, awalnya citra masukan akan dilakukan praproses agar sesuai dengan masukan yang dapat dikelola sistem. Setelah praproses, citra akan melalui proses pengenalan menggunakan model-model yang sebelumnya telah dibuat.

3.3.4.1 Tahap Praproses Citra Masukan

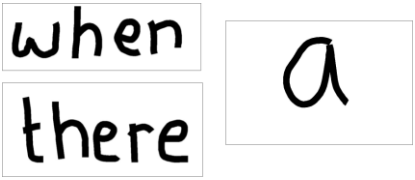
Citra data uji perlu melalui beberapa tahapan praproses agar citra lebih mudah dikonsumsi sistem. Diagram alir tahapana praproses ini dapat dilihat pada Gambar 3.26.

Pada citra data uji ini, tahapan praproses pertama yang dilalui adalah *Thresholding* Otsu. Seperti yang dapat dilihat pada Gambar 3.27, tinta yang digunakan untuk menulis memiliki ketebalan yang berbeda-beda dan pada beberapa titik tulisan terlihat transparan. Kondisi ini menghambat tahapan praproses berikutnya, yaitu pemotongan, dalam mendeteksi titik tepian objek tulisan tangan. Keunggulan dari *thresholding* Otsu dibanding *thresholding* normal ialah tidak diperlukan memasang nilai ambang. Hasil dari *thresholding* Otsu dapat dilihat pada Gambar 3.28.



Gambar 3.26 Diagram Alir Praproses Citra Uji

Gambar 3.27 Contoh Citra Uji yang Perlu *Thresholding* Otsu



Gambar 3.28 Hasil *Thresholding* Otsu

Setelah itu akan dilakukan pemotongan citra. Pemotongan citra dilakukan berdasarkan *bounding box* pada keempat sisi objek tulisan tangan. Pembuatan *bounding box* ini dilakukan menggunakan deteksi kontur. Pemotongan ini perlu dilakukan karena citra data latih yang digunakan bisa dikatakan memiliki *margin* tepian yang kecil, sedangkan pada data uji kebanyakan citra memiliki *margin* tepian yang sangat besar. Contoh hasil pemotongan dapat dilihat pada Gambar 3.29. Setelah pemotongan citra, kata akan diubah ukurannya menjadi 32x128 agar dapat dikonsumsi sistem.

Gambar 3.29 Hasil Pemotongan Citra

3.3.4.2 Tahap Pengujian Sistem Keseluruhan

Pada tahap pengujian ini, semua model yang dibuat sebelumnya akan dipakai untuk membuat sistem sepenuhnya. Pembuatan sistem ini sendiri mengacu pada Gambar 3.3.

Citra yang telah melalui akan masuk ke dalam subsistem Lexicon CNN. Lexicon CNN bekerja mengenali tulisan pada citra berdasarkan kamus kata yang ditentukan pada saat pelatihan. FCN pada sistem ini memiliki ketergantungan pada kamus kata tersebut, sehingga tidak adaptif saat menemukan rangkaian karakter di luar kamus tersebut. Ketika hasil prediksi dari Lexicon CNN memiliki nilai tingkat kepercayaan melebihi batas yang ditentukan, sistem akan menganggap proses keseluruhan telah berakhir dan hasil prediksi Lexicon CNN dianggap sebagai keluaran sistem. Sebaliknya, ketika nilai tingkat kepercayaannya tidak melebihi batas yang ditentukan maka citra akan diteruskan ke Perhitungan Karakter CNN.

Perhitungan Karakter FCN bekerja untuk mengenali banyak karakter pada suatu citra (N). Sebelum memasuki Prediksi Karakter FCN, citra akan diubah terlebih dahulu ukurannya menjadi 32x16N. Pada saat Prediksi Karakter FCN, karakter-karakter pada citra akan dikenali satu per satu menggunakan algoritma *sliding windows* dengan ukuran tertentu.

Hasil prediksi dari sistem ini akan dibandingkan dengan label kelas sebenarnya. Selanjutkan akan dilakukan evaluasi dengan nilai akurasi, *f1-score*, presisi, dan *recall*. Sistem ini juga akan dievaluasi menggunakan CER.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Bab ini menjelaskan mengenai implementasi perangkat lunak dari rancangan sistem yang telah dibahas pada Bab 3 meliputi kode program dalam perangkat lunak. Selain itu, implementasi dari tiap proses, parameter masukan, keluaran, dan beberapa keterangan yang berhubungan dengan program juga dijelaskan.

4.1 Lingkungan Implementasi

Dalam mengimplementasikan aplikasi pengenalan ekspresi manusia diperlukan beberapa perangkat pendukung sebagai berikut.

4.1.1 Perangkat Keras

Implementasi tugas akhir ini mengggunakan fasilitas Google Colaboratory yang memiliki sepesifikasi 4 buah CPU Intel(R) Xeon(R) CPU @2.20GHz, *Random Access Memory* (RAM) sebesar 16GB, dan mempunyai *Graphics Processing Unit* (GPU) dengan spesifikasi Tesla P100-PCIE sebesar 16GB.

4.1.2 Perangkat Lunak

Perangkat lunak yang digunakan memiliki spesifikasi antara lain menggunakan bahasa pemrograman Python 3.7 untuk laptop dan Python 3.6 untuk Google Colaboratory. Meskipun memiliki versi yang berbeda, kedua program ini tidak memiliki perbedaan yang terlalu signifikan untuk pengerjaan tugas akhir ini. Perangkat lunak lain yang digunakan ialah beberapa *library* yang tersedia pada Python, antara lain NLTK, OpenCV, Tensorflow, Keras, Numpy, Matplotlib dan Scikit-learn.

4.2 Implementasi Pembuatan Model Lexicon CNN

Pada subbab ini akan dijelaksan mengenai implementasi tahap pembuatan Lexicon CNN.

4.2.1 Implementasi Pembuatan Kamus Kata

Dalam pembuatan kamus kata, data artikel yang tersedia perlu dibersihkan terlebih dahulu dengan mengeliminasi karakter selain alfabet. Kemudian semua karakter diubah ke huruf kecil.

```
1.  brown_word = []
2.  for fileid in brown.fileids():
3.    text_slashword_removed = re.sub('(/\S+)', '',
      brown.raw(fileid))
4.    text_cleaned = re.sub('[^a-zA-Z\']', ' ',
      text_slashword_removed)
5.    text_cleaned = re.sub('[\'\',]', '',text_cleaned)
6.    brown_word+= string_to_list(text_cleaned.lower())
```

Kode Sumber 4.1 Eliminasi Karakter Non Alfabet Corpus Brown

Kode Sumber 4.1 dijalankan untuk mengeliminasi karakter non alfabet kemudian disimpan dalam list. Pada baris tiga, teks akan dibersihkan terlebih dahulu dari *string* yang muncul setelah tanda garis miring (/) menggunakan *regular expression* hingga *whitespace* ditemukan. Baris empat menjalankan operasi pencarian karakter di luar karakter a hingga z, baik huruf kapital maupun huruf kecil, serta tanda petik (‘) lalu menggantinya dengan spasi. Pada baris ini tanda petik tetap dipertahankan untuk mencegah timbulnya pemotongan kata yang aneh, seperti “*let’s”* menjadi “*let s*”. Pada baris lima barulah semua tanda petik dihilangkan. Hal ini untuk mencegah munculnya kata yang memiliki non alfanumerik yang muncul pada kamus. Pada baris 6 semua karakter yang tersisa diubah ke huruf kecil dan disimpan dalam *list*.

Kode Sumber 4.2 mengimplementasikan hal yang sama pada *corpus* Gutenberg. Pada baris 3 menjalankan operasi pencarian karakter di luar karakter a hingga z, baik huruf kapital maupun huruf kecil, beserta tanda petik lalu menggantinya dengan

spasi. Selanjutnya, kata-kata yang tersisa diubah ke huruf kecil dan disimpan pada *list*. Pada baris keempat, tanda petik dihilangkan semua unutk mencegah munculnya kata dengan karakter non alfanumerik.

```
1.  gutenberg_word = []
2.  for fileid in gutenberg.fileids():
3.    text_cleaned = re.sub('[^a-zA-Z\']', ' ',
      gutenberg.raw(fileid))
4.    text_cleaned = re.sub('[\'\',]','', text_cleaned)
5.    gutenberg_word+=string_to_list(text_cleaned.lower(
      ))
```

Kode Sumber 4.2 Eliminasi Karakter Non Alfabet Corpus Gutenberg

Pada baris 4 Kode Sumber 4.1 dan Kode Sumber 4.2 terdapat fungsi *string_to_list* yang akan didefinisikan pada Kode Sumber 4.3, Baris 2 berjalan untuk memisahkan kata dengan spasi dan menyimpannya dalam list. Karena pada teks memungkinkan adanya *whitespace* berurutan lebih dari satu kali, baris 3 digunakan untuk menghilangkan spasi yang ikut masuk dalam list.

```
1.  def string_to_list(string):
2.    li = list(string.split(" "))
3.    li = [string for string in li if string != ""]
4.    return li
```

Kode Sumber 4.3 Fungsi Mengubah String ke List

Setelah list siap, akan dilakukan pencarian 50 kata yang paling sering muncul menggunakan fungsi most_common dari *library* Collections. Fungsi ini bekerja dengan menyimpan list ke dalam bentuk Counter terlebih dahulu. Counter sendiri merupakan *subclass* dari Collections dimana dia menyimpan data beserta frekuensi kemunculan tanpa diurutkan. Sistem pemeringkatan ini diimplementasikan pada Kode Sumber 4.4.

```
1.  brown_commonword = Counter(brown_word).most_common(
    50)
2.  gutenberg_commonword=Counter(gutenberg_word)most_co
    mmon(50)
```

Kode Sumber 4.4 Mencari 50 Kata yang Paling Sering Muncul

4.2.2 Impelementasi Pembuatan Dataset Citra Lexicon CNN

Dataset citra Lexicon CNN dibuat berdasarkan kata yang paling sering muncul. Masing-masing kata dibuatkan citra sebanya 1.000 buah. Fungsi utama pembuatan dataset ini dapat dilihat pada Kode Sumber 4.5.

Pada baris dua hingga 14 bekerja membaca karakter demi karakter untuk menggabungkannya menjadi kata. Pada baris 4, akan dijalankan fungsi untuk mengambil citra karakter secara acak dari dataset. Selanjutnyap ada baris 6 citra akan diberi *binary inverse thresholding* untuk kemudian dideteksi tepian objek menggunakan deteksi kontur. Deteksi kontur ini sendiri menggunakan fungsi *boundingRect* dari openCV.

Setelah tepiannya ditemukan, pada baris 9 citra dipotong secara horizontal dan mempertahankan tinggi citra. Pemotongan citra horizontal ini sendiri diberi jarak secara acak antara 0 hingga 10 untuk memberi jarak antar karakter nantinya pada gambar. Dengan mempertahankan tinggi citra pun memudahkan proses penggabungan gambar secara horizontal karena semua citra karakter potongan akan memiliki tinggi yang sama. Proses penggabungan karakter-karakter ini dilakukan pada baris 11 hingga 14.

Pada baris 16 hingga 25, citra karakter telah berubah menjadi citra kata. Pada mulanya, citra akan melalui *binary inverse thresholding* kembali untuk dilakukan pemotongan secara vertikal. Sama seperti sebelumnya, pemotongan dilakukan dengan mencari tepian objek tulisan tangan dengan menggunakan deteksi kontur. Pada sisi atas dan bawah citra juga akan diberi jarak secara acak antara 0 hingga 10. Pada baris 20 citra akan diberikan *Gaussian*

blur dengan kernel 3x3. Setelah mengalami pengaburan, citra akan diubah ukurannya menjadi 32x128 sebelum kemudian disimpan.

```
1.  def generate_image(word):
2.     for i in range(len(word)):
3.        character = word[i]
4.        character_image = random_image(character)
5.
6.        character_thresh =
          cv2.threshold(character_image, 0, 255,
          cv2.THRESH_BINARY_INV)[1]
7.        x,y,w,h = cv2.boundingRect(character_thresh)
8.
9.        character_cropped = character_image[0:
          character_image.shape[0], x-randint(0,10)
          :x+w+randint(0,10)]
10.
11.       if(i==0):
12.          word_image = character_cropped
13.       else:
14.          word_image = np.hstack((word_image,
             character_cropped))
15.
16.    word_thresh = cv2.threshold(word_image, 0, 255,
       cv2.THRESH_BINARY_INV)[1]
17.    x,y,w,h = cv2.boundingRect(word_thresh)
18.
19.    word_cropped = word_image[y-randint(0,10)
       :y+h+randint(0,10), x:x+w]
20.    blur = cv2.GaussianBlur(word_cropped, (3,3),0)
21.    final_image = cv2.resize(blur, (128,32),
       interpolation=cv2.INTER_AREA)
22.    return final_image
```

Kode Sumber 4.5 Fungsi Pembuatan Citra Kata

Kode Sumber 4.6 mengimplementasikan pemanggilan fungsi pada tiap kata dari *list* kata yang sering muncul. Pada baris 2 dilakukan iterasi 1000 kali untuk membuat 1.000 buah citra pada tiap kelas.

```
for word in list_commonword:
   for i in range(1000):
      final_image = generate_image(word)
```

Kode Sumber 4.6 Implementasi Pembuatan Dataset Lexicon CNN

4.2.3 Implementasi Pembangunan Arsitektur Lexicon CNN

Dalam pembangunan arsitektur Lexicon CNN, fungsi Kode Sumber 4.7 perlu dipanggil. Pada fungsi tersebut, arsitektur CNN yang dibangun memiliki 6 macam layer, yakni *Convolution2D*, *MaxPooling2D*, *Flatten*, *Dropout*, dan *Dense*. *Convolution2D* merupakan implementasi *convolutional layer* dalam bentuk dua dimensi.

Conv2D merupakan implementasi *convolution* layer dalam bentuk dua dimensi. Pada *layer* Conv2D terdapat lima parameter sebagai berikut:

1. Parameter *filters* menunjukkan banyak *filter* digunakan.
2. Parameter *kernel_size* menunjukkan ukuran *kerne l*(h, w).
3. Parameter *strides* menunjukkan ukuran *strides* (h,w).
4. Parameter *acitivation* adalah fungsi aktivasi. Fungsi aktivasi yang digunakan pada *convolutional layer* ialah ReLU (Rectified Linear Unit).
5. Parameter *padding* menentukan jenis *padding* pada citra sebelum memasukin *convolutional layer*. Jika *padding* memiliki nilai *'same'* maka keluaran dari *convolutional layer* akan memiliki ukuran yang sama dengan ukuran masukan. Secara *default*, *padding* pada Conv2D terpasang dengan nilai '*valid*.'

MaxPooling2D adalah implementasi *Max Pooling Layer* dalam bentuk dua dimensi. Terdapat dua parameter yang digunakan pada *MaxPooling2D* dengan penjelasan sebagai berikut:

1. Parameter *pool_size* menunjukkan ukuran kernel *Pooling Layer*.

2. Parameter *strides* menunjukkan ukuran *strides* yang digunakan.

Batch Normalization digunakan untuk menormalisasi keluaran dari *convolutional layer* dan *fully connected* layer. Flatten pada arsitektur CNN digunakan untuk mengubah peta fitur menjadi vektor fitur yang memiliki satu dimensi. Vektor fitur ini yang nantinya akn menjadi masukan pada *fully connected layer*. *Dropout* adalah implementasi teknik *dropout* yang berfungsi membuat beberapa *neuron* yang dipilih secara acak untuk tidak dipakai selama proses pelatihan. *Dropout* memiliki parameter *rate* yakni nilai probabilitas yang dipakai dalam menentukan secara acak *neuron*.

```
1.  def build_lexicon_model(regularization_type):
2.    model = Sequential()
3.    model.add(Activation(activation='relu',
      input_shape=(32, 128, 1)))
4.
5.    model.add(Conv2D(64, (3, 3), padding='same',
      activation='relu'))
6.    model.add(BatchNormalization())
7.    model.add(Conv2D(64, (3, 3), padding='same',
      activation='relu'))
8.    model.add(BatchNormalization())
9.    model.add(Conv2D(64, (3, 3), padding='same',
      activation='relu'))
10.   model.add(BatchNormalization())
11.   model.add(MaxPooling2D(pool_size=(2, 2),
      strides=(2,2)))
12.   model.add(Conv2D(128, (3, 3), padding='same',
      activation='relu'))
13.   model.add(BatchNormalization())
14.   model.add(Conv2D(128, (3, 3), padding='same',
      activation='relu'))
15.   model.add(BatchNormalization())
16.   model.add(MaxPooling2D(pool_size=(2, 2),
      strides=(2,2)))
```

```
17.   model.add(Conv2D(256, (3, 3),  padding='same',
      activation='relu'))
18.   model.add(BatchNormalization())
19.   model.add(Conv2D(256, (3, 3),  padding='same',
      activation='relu'))
20.   model.add(BatchNormalization())
21.   model.add(MaxPooling2D(pool_size=(2, 2),
      strides=(2,2)))
22.
23.   model.add(Conv2D(512, (3, 3),  padding='same',
      activation='relu'))
24.   model.add(BatchNormalization())
25.   model.add(Conv2D(512, (3, 3),  padding='same',
      activation='relu'))
26.   model.add(BatchNormalization())
27.   model.add(MaxPooling2D(pool_size=(2, 2),
      strides=(2,2)))
28.
29.   model.add(Flatten())
30.   model.add(Dense(2048, activation='relu'))
31.   model.add(BatchNormalization())
32.   model.add(Dropout(0,5))
33.
34.   if(regularization_type == 'l1'):
35.     regularization = regularizers.l1(0.025)
36.   if(regularization_type == 'l2'):
37.     regularization = regularizers.l2(0.025)
38.   model.add(Dense(50, activity_regularizer =
      regularization, activation ='softmax'))
39.   return model
```

Kode Sumber 4.7 Fungsi Pembangunan Arsitektur Lexicon CNN

Dense adalah implementasi *Fully Connected Layer*. *Dense* menggunakan 2 parameter, antara lain:

1. Parameter *units* adalah jumlah neuron.
2. Parameter *activation* adalah fungsi aktivasi apa yang digunakan. Terdapat banyak pilihan fungsi aktivasi yang disediakan oleh Keras, salah satunya fungsi ReLU dan fungsi

softmax, yang biasa digunakan pada akhir arsitektur untuk mengklasifikasikan label kelas.

Fungsi pembangunan Lexicon CNN dipanggil pada Kode Sumber 4.7. Fungsi ini membutuhkan parameter *string* jenis regulator. Masukan 'l1' akan dibaca sebagai regulator L1 dan 'l2' dibaca sebagai regulator L2. Pemanggilan fungsi pembangunan Lexicon CNN diimplementasikan pada Kode Sumber 4.8.

```
1.  model = build_lexicon_model(regularization_type)
```

Kode Sumber 4.8 Implementasi Pembangunan Lexicon CNN

4.2.4 Implementasi Pelatihan dan Evaluasi

Proses pelatihan Lexicon CNN menggunakan *loss function* berupa *cross entropy* dan terdapat 3 macam *optimizer* yang nantinya akan dibandingkan kinerjanya, yakni RMSProp, Adam, dan SGD. Selain itu *learning rate* juga akan dibandingkan untuk mengetahui kinerja masing. Pada proses pelatihan ini, dilakukan epoch sebanyak 30 kali dan ukuran *batch* yang digunakan ialah 64. Adapun implementasi pelatihan ini dapat dilihat pada Kode Sumber 4.9.

```
1.  model.compile(optimizer=opt,loss='categorical_cross
    entropy', metrics=['accuracy'])
2.  model_lexicon = model.fit(X_train, y_train,
    validation_data=(X_test, y_test),epochs=30,
    batch_size=64,shuffle=True,verbose=2)
```

Kode Sumber 4.9 Implementasi Pelatihan Model Lexicon CNN

Proses evaluasi model dilakukan menggunakan *Confusion Matrix* dan CER. *Confusion Matrix* dibangun menggunakan *library* Scikit-learn. *Confusion Matrix* ini dapat dilihat pada Kode Sumber 4.10. Evaluasi CER dibangun secara manual. Fungsi CER dapat dilihat pada Kode Sumber 4.11..

```
y_pred = lexicon_model.predict(data)
report = classification_report(y_true, y_pred)
```

Kode Sumber 4.10 Implementasi Evaluasi Menggunakan *Confusion Matrix* pada Lexicon CNN

```
def cer(prediction,label):
    w, h = len(prediction)+1, len(label)+1
    Matrix= [[0 for i in range(w)]for j in range(h)]
    for i in range(h):
       for j in range(w):
          if(i==0):
             Matrix[0][j]=j
          elif(j==0):
             Matrix[i][0]=i
          else:
             if(prediction[j-1]==label[i-1]):
                diag = Matrix[i-1][j-1]
             else:
                diag = Matrix[i-1][j-1]+1
          Matrix[i][j]=min(diag,Matrix[i-1][j]+1,
          Matrix[i][j-1]+1)
    n=0
    if(w>h):
       n=w-1
    else:
       n=h-1
    return float(Matrix[h-1][w-1]/n)
```

Kode Sumber 4.11 Fungsi CER

```
cer_total = 0
for i in range(len(y_pred)):
  cer_value = cer(y_pred[i], y_true[i])
  cer_total +=cer_value
cer_avg = cer_total/len(y_pred)
```

Kode Sumber 4.12 Implemetasi Perhitungan Rata-Rata CER

Implementasi perhitungan nilai CER pada evaluasi Lexicon CNN dapat dilihat pada Kode Sumber 4.12. Pencarian rata-rata didapat dengan mendapat nilai total CER untuk kemudian dibagi dengan jumlah data uji.

4.3 Implementasi Pembuatan Perhitungan Karakter CNN

Pada subbab ini akan dijelaskan mengenai implementasi pembuatan Perhitungan Karakter CNN.

4.3.1 Implementasi Pembuatan Dataset Citra Perhitungan Karakter CNN

Pada Perhitungan Karakter CNN terdapat 16 kelas, masing-masing merepresentasikan banyak karakter pada kata. Masing-masing kelas ini akan dibuat citra sebanyak 1.000 citra. Pada bagian pembuatan citra, Perhitungan Karakter CNN dan Lexicon CNN hanya memiliki perbedaan pada kata masukan.

Fungsi yang digunakan untuk membuat dataset citra ini diimplementasikan pada Kode Sumber 4.5. Untuk pembuatan kata dengan banyak karakter tertentu, dapat dilihat pada Kode Sumber 4.16. Pada baris 3 akan diinisiasi *string* yang tidak memiliki karaker apapun. Kemudian dilakukan iterasi sesuai panjang karakter kelas untuk mengacak karakter apa yang nantinya dirangkai. Setelah karakter demi karakter dirangkai membentuk kata, fungsi *generate_image* akan dipanggil dengan masukan kata hasil rangkaian tersebut. Proses ini dilakukan iterasi 1.000 kali untuk 16 kelas.

```
1.  for length in range(16):
2.     for i in range(1000):
3.        word = ‘’
4.        for j in range(length+1):
5.           word+= random.choice(symbol)
6.           final_image = generate_image(word)
```

Kode Sumber 4.13 Implementasi Pembuatan Dataset Perhitungan Karakter CNN

4.3.2 Implemenetasi Pembangunan Arsitektur Perhitungan Karakter CNN

Sedikit berbeda dengan arsitektur Lexicon CNN, pada Perhitungan Karakter CNN *convolutional layer* menggunakan fungsi *maxout*. Fungsi membangun *convolutional layer* menggunakan *maxout* dapat dilihat pada Kode Sumber 4.14.

Pada baris 9 hingga baris 18 merupakan kumpulan perhitungan dari fungsi aktivasi *maxout*. Sedamglam baris 20 hingga baris 31 merupakan implementasi untuk menghasilkan output yang sesuai dengan keinginan.

Implementasi pembangunan arsitektur Perhitungan Karakter CNN dapat dilihat pada Kode Sumber 4.15

```
1.  class MaxoutConv2D(Layer):
2.    def __init__(self, kernel_size, output_dim, nb_fea
      tures=4, padding='valid', **kwargs):
3.       self.kernel_size = kernel_size
4.       self.output_dim = output_dim
5.       self.nb_features = nb_features
6.       self.padding = padding
7.       super(MaxoutConv2D, self).__init__(**kwargs)
8.
9.    def call(self, x):
10.      output = None
11.      for _ in range(self.output_dim):
12.         conv_out = Conv2D(self.nb_features, self.ker
            nel_size, padding=self.padding)(x)
13.         maxout_out = K.max(conv_out, axis=-1,
            keepdims=True)
14.         if output is not None:
15.            output = K.concatenate([output, maxout_ou
               t], axis=-1)
16.         else:
17.            output = maxout_out
18.      return output
```

```
19.  def compute_output_shape(self, input_shape):
20.     input_height= input_shape[1]
21.     input_width = input_shape[2]
22.     if(self.padding == 'same'):
23.        output_height = input_height
24.        output_width = input_width
25.     elif(input_height==None or input_width== None):
26.        return (input_shape[0], None, None, self.out
           put_dim)
27.     else:
28.        output_height = input_height - self.kernel_s
           ize[0] + 1
29.     output_width=input_width-self.kernel_size[1]+1
30.     return (input_shape[0], output_height, output_w
        idth, self.output_dim)
```

Kode Sumber 4.14 Fungsi Membangun Layer *Conv2D* dengan Fungsi Aktivasi *Softmax* [32]

```
1.  def build_countchar_model(dropout_const
    regularization_type):
2.   model = Sequential()
3.   model.add(Activation(activation='relu', input_shap
     e=(32, 128, 1)))
4.   model.add(MaxoutConv2D((3,3),64,padding='same'))
5.   model.add(BatchNormalization())
6.   model.add(MaxoutConv2D((3,3),64,padding='same'))
7.   model.add(BatchNormalization())
8.   model.add(MaxoutConv2D((3,3),64,padding='same'))
9.   model.add(BatchNormalization())
10.  model.add(MaxPooling2D(pool_size=(2, 2),strides=(2
     , 2)))
11.  model.add(MaxoutConv2D((3,3),128, padding='same'))
12.  model.add(BatchNormalization())
13.  model.add(MaxoutConv2D((3,3),128, padding='same'))
14.  model.add(BatchNormalization())
15.  model.add(MaxPooling2D(pool_size=(2, 2),strides=(2
     , 2)))
```

```
16.  model.add(MaxoutConv2D((3,3),256, padding='same'))
17.  model.add(BatchNormalization())
18.  model.add(MaxoutConv2D((3,3),256, padding='same'))
19.  model.add(BatchNormalization())
20.  model.add(MaxPooling2D(pool_size=(2, 2),strides=(2
     , 2)))
21.
22.  model.add(MaxoutConv2D((3,3),512, padding='same'))
23.  model.add(BatchNormalization())
24.  model.add(MaxoutConv2D((3,3),512, padding='same'))
25.  model.add(BatchNormalization())
26.  model.add(MaxPooling2D(pool_size=(2, 2),strides=(2
     , 2)))
27.
28.  model.add(Flatten())
29.  model.add(Dense(256, activation='relu'))
30.  model.add(BatchNormalization())
31.  model.add(Dense(64, activation='relu'))
32.  model.add(BatchNormalization())
33.  model.add(Dropout(dropout_const))
34.
35.  if(regularization_type == 'l1'):
36.    regularization = regularizers.l1(0.025)
37.  if(regularization_type == 'l2'):
38.    regularization = regularizers.l2(0.025)
39.
40.  model.add(Dense(16, activity_regularizer=regulariz
     ation, activation='softmax'))
41.  return model
```

Kode Sumber 4.15 Fungsi Pembangunan Arsitektur Perhitungan Karakter CNN

Fungsi pembangunan Perhitungan Karakter CNN dipanggil pada Kode Sumber 4.15. Fungsi ini membutuhkan parameter *string* jenis regulator dan *dropout*. Masukan ‘l1’ akan dibaca sebagai regulator L1 dan ‘l2’ dibaca sebagai regulator L2. Pemanggilan fungsi pembangunan Lexicon CNN diimplementasikan pada Kode Sumber 4.16.

```
1.  model = build_countchar_model(dropout_const,
    regularization_type)
```

Kode Sumber 4.16 Implementasi Pembangunan Perhitungan Karakter CNN

4.3.3 Implementasi Pelatihan dan Evaluasi

Proses pelatihan Perhitungan Karakter CNN menggunakan *loss function* berupa *cross entropy* dan terdapat 3 macam *optimizer* yang nantinya akan dibandingkan kinerjanya, yakni RMSProp, Adam, dan SGD. Selain itu *learning rate* dan dropout juga akan dibandingkan untuk mengetahui kinerja masing. Pada proses pelatihan ini, dilakukan epoch sebanyak 50 kali dan ukuran *batch* yang digunakan ialah 64. Adapun implementasi pelatihan ini dapat dilihat pada Kode Sumber 4.17.

```
1.  model.compile(optimizer=opt,loss='categorical_cross
    entropy', metrics=['accuracy'])
2.  model_countchar = model.fit(X_train, y_train,
    validation_data=(X_test, y_test),epochs=50,
    batch_size=32,shuffle=True,verbose=2)
```

Kode Sumber 4.17 Implementasi Pelatihan Model Perhitungan Karakter CNN

Proses evaluasi model dilakukan hanya menggunakan *Confusion Matrix*. *Confusion Matrix* dibangun menggunakan *library* Scikit-learn. *Confusion Matrix* ini dapat dilihat pada Kode Sumber 4.18Kode Sumber 4.10 .

```
1.  y_pred = lexicon_model.predict(data)
2.  report = classification_report(y_true, y_pred)
```

Kode Sumber 4.18 Implementasi Evaluasi Menggunakan *Confusion Matrix* pada Perhitungan Karakter CNN

4.4 Implementasi Pembuatan Prediksi Karakter FCN

Pada subbab ini akan dijelaskan mengenai implementasi pembuatan Prediksi Karakter FCN.

4.4.1 Implementasi Praproses Citra Data Latih

Pada praproses pelatihan Prediksi Karakter FCN, citra latih akan dipotong dan diubah ukuran dulu. Fungsi praproses ini dapat dilihat pada Kode Sumber 4.19.

Pada baris dua hingga baris lima dilakukan proses pemotongan berdasarkan *bounding box*. Pada baris enam citra akan diubah ukurannya ke dalam 16x32.

```
1.  def preproc_fcn(word_image):
2.      word_thresh = cv2.threshold(word_image, 0, 255,
        cv2.THRESH_BINARY_INV)[1]
3.      x,y,w,h = cv2.boundingRect(word_thresh)
4.      word_cropped = word_image[y:y+h, x:x+w]
5.
6.      final_image = cv2.resize(blur,
        (16,32),interpolation=cv2.INTER_AREA)
7.      return final_image
```

Kode Sumber 4.19 Fungsi Praproses Citra Data Latih Prediksi FCN

4.4.2 Implementasi Pembangunan Arsitektur

Fungsi pembangunan Prediksi Karakter FCN dipanggil pada Kode Sumber 4.20. Berbeda dengan CNN, pada bagian prediksi FCN tetap menggunakan *layer Conv2D*. Bagian prediksi ini dapat dilihat mulai baris 32 hingga 41.

```
1.  def get_predict_model():
2.      model = Sequential()
3.      model.add(Activation(activation='relu', input_sh
        ape=(32, None, 1)))
4.      model.add(Conv2D(64, (3, 3), padding='same',
        activation='relu'))
```

```
5.     model.add(BatchNormalization())
6.     model.add(Conv2D(64, (3, 3), padding='same',
       activation='relu'))
7.     model.add(BatchNormalization())
8.     model.add(Conv2D(64, (3, 3), padding='same',
       activation='relu'))
9.     model.add(BatchNormalization())
10.    model.add(MaxPooling2D(pool_size=(2, 2), strides
       =(2,2)))
11.
12.    model.add(Conv2D(128, (3, 3), padding='same',
       activation='relu'))
13.    model.add(BatchNormalization())
14.    model.add(Conv2D(128, (3, 3), padding='same',
       activation='relu'))
15.    model.add(BatchNormalization())
16.    model.add(MaxPooling2D(pool_size=(2, 2), strides
       =(2,2)))
17.
18.    model.add(Conv2D(256, (3, 3),  padding='same',
       activation='relu'))
19.    model.add(BatchNormalization())
20.    model.add(Conv2D(256, (3, 3),  padding='same',
       activation='relu'))
21.    model.add(BatchNormalization())
22.    model.add(MaxPooling2D(pool_size=(2, 2), strides
       =(2,2)))
23.
24.    model.add(Conv2D(512, (3, 3),  padding='same',
       activation='relu'))
25.    model.add(BatchNormalization())
26.    model.add(Conv2D(512, (3, 3),  padding='same',
       activation='relu'))
27.    model.add(BatchNormalization())
28.
29.    model.add(Dropout(0,5))
30.
31.    model.add(ZeroPadding2D(padding=(1,2)))
32.    model.add(Conv2D(512, (4, 4), activation='relu',
       padding='valid'))
33.    model.add(BatchNormalization())
34.    model.add(ZeroPadding2D(padding=(0,4)))
```

```
35.    model.add(Conv2D(1024, (3, 9),activation='relu',
       padding='valid'))
36.    model.add(BatchNormalization())
37.
38.    model.add(Conv2D(63, (1, 1)))
39.    model.add(GlobalAveragePooling2D())
40.    model.add(Activation('softmax'))
41.
42.    return model
```

Kode Sumber 4.20 Fungsi Pembangunan Arsitektur Prediksi Karakter FCN

4.4.3 Implementasi Pelatihan dan Evaluasi

Proses pelatihan Prediksi Karakter FCN menggunakan *loss function* berupa *cross entropy* dan terdapat 3 macam *optimizer* yang nantinya akan dibandingkan kinerjanya, yakni RMSProp, Adam, dan SGD. Selain itu *learning rate* dan dropout juga akan dibandingkan untuk mengetahui kinerja masing. Pada proses pelatihan ini, dilakukan epoch sebanyak 50 kali dan ukuran *batch* yang digunakan ialah 64. Adapun implementasi pelatihan ini dapat dilihat pada Kode Sumber 4.21.

```
1.  model.compile(optimizer=opt,loss='categorical_cross
    entropy', metrics=['accuracy'])
2.  model_countchar = model.fit(X_train, y_train,
    validation_data=(X_test, y_test),epochs=50,
    batch_size=64,shuffle=True,verbose=2)
```

Kode Sumber 4.21 Implementasi Pelatihan Model Prediksi Karakter FCN

Proses evaluasi model dilakukan hanya menggunakan *Confusion Matrix*. *Confusion Matrix* dibangun menggunakan *library* Scikit-learn. Pada evaluasi ini, N (banyak karakter)

dianggap telah diketahui dan gambar akan diubah ukurannya menyesuaikan banyak karakter tersebut. *Confusion Matrix* yang digunakan ini dapat dilihat pada Kode Sumber 4.22.

```
1.  y_pred = lexicon_model.predict(data)
2.  report = classification_report(y_true, y_pred)
```

Kode Sumber 4.22 Implementasi Evaluasi Menggunakan *Confusion Matrix* pada Prediksi Karakter FCN

4.5 Implementasi Pengujian Sistem

Pada subbab ini akan dijelaskan mengenai implementasi tahapan yang ada pada saat evaluasi sistem.

4.5.1 Implementasi Praproses Citra Masukan

Praproses citra masukan pada data uji dilakukan untuk mengelola citra sehingga lebih mudah dikonsumsi oleh sistem. Adapun fungsi praproses citra untuk pengujian dapat dilihat pada Kode Sumber 4.23.

```
1.  def preproc(image):
2.     word_thresh_otsu = cv2.threshold(img, 0, 255,
       cv2.THRESH_BINARY+cv2.THRESH_OTSU)[1]
3.     word_thresh = cv2.threshold(word_thresh_otsu, 0,
       255, cv2.THRESH_BINARY_INV)[1]
4.     x,y,w,h = cv2.boundingRect(word_thresh)
5.     word_cropped = word_thresh_otsu[y:y+h, x:x+w]
6.
7.     final_image = cv2.resize(word_cropped,
       (128,32),interpolation=cv2.INTER_AREA)
8.     return final_image
```

Kode Sumber 4.23 Fungsi Praproses Citra Data Uji

Fungsi akan melakukan *thresholding* Otsu pada citra terlebih dahulu untuk melakukan perbaikan pada penggunaan warna tinta. *Thresholding* Otsu ini dilakukan pada baris dua. Setelahnya, akan dilakukan deteksi kontur. Deteksi kontur terjadi

pada baris 3 dan 4. Setelah tepian objek kata tulisan tangan diteukan, citra dipotong sesuai titik-titik tersebut. Terakhir, citra akan diubah ukurannya menjadi 32x128 pada baris 7.

4.5.2 Implementasi Pengujian Sistem Keseluruhan

Implementasi pengujian sistem keseluruhan dilakukan untuk menguji jalannya sistem keseluruhan. Sistem keseluruhan yang dimaksud ialah ketika rangkaian Lexicon CNN, Perhitungan Karakter CNN, dan Prediksi Karakter FCN menjadi satu dan berjalan berurutan.

Pada Prediksi Karakter FCN, karakter pada citra akan dikenali satu per satu menggunakan algoritma *sliding windows*. Implementasi *sliding windows* ini sendiri dapat dilihat pada Kode Sumber 4.24 untuk *sliding windows* sebesar 32x16 dan Kode Sumber 4.25 *untuk sliding windows* sebesar 32x32. Pada ukuran *sliding windows* 32x32, citra terlebih dahulu diberikan padding tambahan pada bagian kiri kanan sebesar 8 pixel.

```
1.  for i in range(2*n-1):
2.     start = i*8
3.     end = (i+2)*8
4.
5.     img_sliding = np.array(im_resize[0:32,start:end]
       , dtype=np.uint8)
6.     img_sliding = np.reshape(im_resize[0:32,start:en
       d],(1,32,16,1))
```

Kode Sumber 4.24 Implementasi Algoritma *Sliding Windows* Sebesar 32x16

```
1.  image = cv2.copyMakeBorder(im_resize, 0, 0, 8, 8, c
    v2.BORDER_CONSTANT, value=255)
2.  for i in range(2*n-1):
3.     start = i*8
4.     end = (i+4)*8
5.
6.     img_sliding = np.array(im_resize[0:32,start:end]
       , dtype=np.uint8)
```

```
7.     img_sliding = np.reshape(im_resize[0:32,start:en
       d],(1,32,32,1))
```

Kode Sumber 4.25 Implementasi Algoritma *Sliding Windows* Sebesar 32x32

Keluaran dari rangkaian sistem inilah yang nantinya akan dievaluasi. Proses evaluasi sistem dilakukan hanya menggunakan *Confusion Matrix* dan CER. *Confusion Matrix* ini dapat dilihat pada Kode Sumber 4.26. Fungsi CER dapat dilihat pada Kode Sumber 4.11.

```
1.  y_pred = lexicon_model.predict(data)
2.  report = classification_report(y_true, y_pred)
```

Kode Sumber 4.26 Implementasi Evaluasi Menggunakan *Confusion Matrix* pada Output Sistem Keseluruhan

```
1.  cer_total = 0
2.  for i in range(len(y_pred)):
3.    cer_value = cer(y_pred[i], y_true[i])
4.    cer_total +=cer_value
5.  cer_avg = cer_total/len(y_pred)
```

Kode Sumber 4.27 Implemetasi Perhitungan Rata-Rata CER

Implementasi perhitungan nilai CER pada evaluasi kali ini dapat dilihat pada Kode Sumber 4.27. Pencarian rata-rata didapat dengan mendapat nilai total CER untuk kemudian dibagi dengan jumlah data uji.

[Halaman ini sengaja dikosongkan]

BAB V
UJI COBA DAN EVALUASI

Bab ini akan membahas mengenai hasil uji coba sistem yang telah dirancang dan dibuat. Uji coba dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan.

5.1 Lingkungan Uji Coba

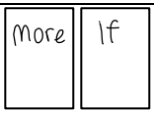
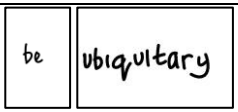
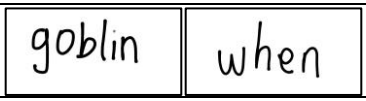
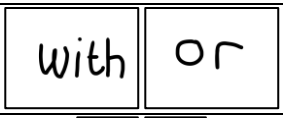
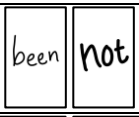
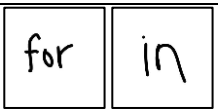
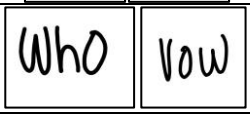
Implementasi tugas akhir ini menggunakan mengggunakan fasilitas Google Colaboratory yang memiliki sepesifikasi 4 buah CPU Intel(R) Xeon(R) CPU @ 2.20GHz, *Random Access Memory* (RAM) sebesar 16GB, dan mempunyai *Graphics Processing Unit* (GPU) dengan spesifikasi Tesla P100-PCIE sebesar 16GB. Pada sisi perangkat lunak, uji coba dilakuka menggunakan Python 3.6. yang dilengkapi beberapa *library* seperti OpenCV, Tensorflow, Keras, Numpy, Matplotlib dan Scikit-learn.

5.2 Deskripsi Dataset

Pada tugas akhir ini, dataset yang digunakan pada saat pelatihan Prediksi Karakter FCN ialah NIST SD19, sedangkan untuk Lexicon CNN dan Perhitungan Karakter CNN menggunakan citra kata gabungan dari citra karakter NIST SD19. Pada proses evaluasi untuk masing-masing *neural network* ini juga digunakan data validasi dari dataset pelatihan. Data latih dan data evaluasi ini memiliki rasio 9:1 dari dataset pelatihan.

Data uji untuk sistem keseluruhan ini diambil dari 10 orang yang menuliskan sekitar 25-68 kata yang berbeda-beda. Kata-kata ini dituliskan pada layar sentuh perangkat elektronik. Citra kata ini memiliki kriteria *background* bewarna putih dengan tinta yang digunakan bewarna hitam dengan ketebalan tinta yang berbeda-beda. Beberapa tulisan tangan ini dapat dilihat pada Tabel 5.1.

Tabel 5.1 Contoh Data Uji dari Tiap Penulis

Subjek	Contoh Tulisan
Penulis 1	
Penulis 2	
Penulis 3	
Penulis 4	
Penulis 5	
Penulis 6	
Penulis 7	
Penulis 8	
Penulis 9	
Penulis 10	

5.3 Hasil Praproses Citra

Sebelum memasuki proses pengenalanan tulisan tangan, citra masukan akan melalui tahapan praproses terlebih dahulu. Pertama-tama data yang diterima akan dibersihkan dulu dari data uji yang memiliki tulisan yang tidak terlalu baik. Citra kata yang dikategorikan tidak baik ini memiliki cira dimana terdapat karakter yang bertumpukan dengan karakter. Selanjutnya citra yang terpilih akan dipotong berdasarkan *bounding box* hasil dari deteksi kontur. Setelah melalui pemotongan, citra akan diubah ukurannya menjadi ukuran 32x128. Beberapa contoh hasil praproses dapat dilihat pada Gambar 5.1.

Gambar 5.1 Contoh Citra Setelah Melalui Praproses

5.4 Skenario Uji Coba

Proses uji coba berguna untuk menemukan parameter-parameter yang menghasilkan performa model yang paling optimal. Parameter yang tepat akan memberikan hasil yang lebih baik pada saat proses uji coba. Karena pada tugas akhir ini dibuat tiga model yang memiliki tugas berbedaberbeda, skenario uji coba akan dibedakan berdasarkan model dan juga akan dilakukan dengan menjalankan sistem keseluruhan.

5.4.1 Skenario Uji Coba pada Lexicon CNN

Pada Lexicon CNN, terdapat dua *corpus* yang digunakan dan kedua *corpus* tersebut menghasilkan daftar kata yang sering muncul. Kedua daftar ini memiliki kata-kata yang tidak jauh berbeda. Masing-masing daftar kata ini akan melalui skenario uji coba yang meliputi:

1. Skenario uji coba pada jenis *regularizer*
2. Skenario uji coba pada *optimizer*.
3. Skenario uji coba pada *learning rate*.

Pada skenario uji coba ini digunakan dataset dari data validasi sebesar 10% dari dataset atau sebesar 5.900 buah citra.

5.4.1.1 Skenario Jenis Regularizer

Pada skenario *regularizer*, terdapat dua *regularizer* yang akan diujikan, yakni L1 dan L2. Pada skenario perbandingan ini, digunakan *optimizer* Adam dengan *learning rate* sebesar 0,01. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.2.

Tabel 5.2 Perbandingan Hasil Evaluasi pada Uji Coba Jenis *Regularizer* pada Lexicon CNN

	L1	**L2**
Lama Prediksi (detik)	45,08	**43,92**
Akurasi (%)	99,76	**99,93**
Presisi (%)	99,77	**99,93**
Recall (%)	99,76	**99,93**
F1-score (%)	99,76	**99,93**
Rata-Rata CER(%)	0,14	**0,05**

5.4.1.2 Skenario Optimizer

Pada skenario *optimizer*, terdapat tiga *optimizer* yang akan diujikan, yakni SGD, Adam, dan RMSProp. Pada skenario perbandingan ini, jenis *regularizer* yang digunakan merupakan L2 sebesar 0.025 dan *learning rate* yang digunakan sebesar 0,01. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.3.

Tabel 5.3 Perbandingan Hasil Evaluasi pada Uji Coba *Optimizer* pada Lexicon CNN

	SGD	**Adam**	**RMSProp**
Lama Prediksi (detik)	44,18	**43,92**	43,21
Akurasi (%)	99,81	**99,93**	99,78
Presisi (%)	99,82	**99,93**	99,79
Recall (%)	99,81	**99,93**	99,78
F1-score (%)	99,81	**99,93**	99,78
Rata-Rata CER(%)	0,01	**0,05**	0,02

5.4.1.3 Skenario Learning Rate

Pada skenario *learning rate*, terdapat tiga parameter *learning rate* yang akan diujikan, yakni 0,01, 0,001, dan 0,0001. Pada skenario ini *optimizer* yang digunakan ialah Adam dengan *regularizer* L2 sebesar 0.025. Hasil Skenario ini dapat dilihat pada Tabel 5.4.

Tabel 5.4 Perbandingan Hasil Evaluasi pada Uji Coba *Learning Rate* pada Lexicon CNN

	0,01	**0,001**	**0,0001**
Lama Prediksi (detik)	43,92	**45,04**	43,30
Akurasi (%)	99,93	**99,98**	99,97
Presisi (%)	99,93	**99,98**	99,97
Recall (%)	99,93	**99,98**	99,97
F1-score (%)	99,93	**99,98**	99,97
Rata-Rata CER(%)	0,05	**0,01**	0,03

5.4.2 Skenario Uji Coba pada Perhitungan Karakter CNN

Pada Perhitungan Karakter CNN, terdapat empat skenario uji coba, antara lain:

1. Skenario uji coba pada arsitektur yang digunakan.
2. Skenario uji coba pada jenis *regularizer*
3. Skenario uji coba pada *optimizer*.
4. Skenario uji coba pada *learning rate*.
5. Skenario uji coba pada *dropout*.

Pada skenario uji coba ini digunakan dataset dari data validasi sebesar 10% dari dataset atau sebesar 1.600 buah citra.

5.4.2.1 Skenario Arsitektur

Pada skenario arsitektur yang digunakan, arsitektur Perhitungan Karakter CNN dan Lexicon CNN akan diujikan. Pada skenario perbandingan ini, digunakan *optimizer* Adam dengan L2

regularizer, *learning rate* sebesar 0,01, dan dropout 0,5. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.5.

Tabel 5.5 Perbandingan Hasil Evaluasi pada Uji Coba Arsitektur pada Perhitungan Karakter CNN

	Arsitektur Perhitungan Karakter CNN	**Arsiktektur Lexicon CNN**
Lama Prediksi (detik)	457,04	**15,59**
Akurasi (%)	6,25	**96,25**
Presisi (%)	0,39	**96,33**
Recall (%)	6,25	**96,25**
F1-score (%)	0,74	**96,25**

5.4.2.2 Skenario Jenis Regularizer

Pada skenario *regularizer*, terdapat dua *regularizer* yang akan diujikan, yakni L1 dan L2. Pada skenario perbandingan ini, digunakan *optimizer* Adam dengan *learning rate* sebesar 0,01 serta dropout 0,5 dengan arsitektur Lexicon CNN. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.6.

Tabel 5.6 Perbandingan Hasil Evaluasi pada Uji Coba *Regularizer* pada Perhitungan Karakter CNN

	L1	**L2**
Lama Prediksi (detik)	16,04	**15,59**
Akurasi (%)	96,06	**96,25**
Presisi (%)	96,20	**96,33**
Recall (%)	96,06	**96,25**
F1-score (%)	96,06	**96,25**

5.4.2.3 Skenario Optimizer

Pada skenario *optimizer*, terdapat tiga *optimizer* yang akan diujikan, yakni Adam, SGD, dan RMSProp. Pada skenario perbandingan ini, digunakan *learning rate* sebesar 0,01 serta

dropout 0,5 dengan arsitektur Lexicon CNN dengan L2 *regularizer*. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.7.

Tabel 5.7 Perbandingan Hasil Evaluasi pada Uji Coba *Optimizer* pada Perhitungan Karakter CNN

	SGD	**Adam**	**RMSProp**
Lama Prediksi (detik)	15,65	**15,59**	15,54
Akurasi (%)	70,31	**96,25**	96,06
Presisi (%)	69,74	**96,33**	96,12
Recall (%)	70,31	**96,25**	96,06
F1-score (%)	69,61	**96,25**	96,06

5.4.2.4 Skenario Learning Rate

Pada skenario *learning rate*, terdapat tiga *learning rate* yang akan diujikan, yakni 0,01, 0,001, dan 0,0001. Pada skenario perbandingan ini, digunakan *optimizer* Adam dengan L2 *regularizer* serta dropout 0,5 dengan arsitektur Lexicon CNN. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.8.

Tabel 5.8 Perbandingan Hasil Evaluasi pada Uji Coba *Learning Rate* pada Perhitungan Karakter CNN

	0,01	**0,001**	**0,0001**
Lama Prediksi (detik)	15,59	**11,88**	11,58
Akurasi (%)	96,25	**97,75**	59,00
Presisi (%)	96,33	**97,76**	59,72
Recall (%)	96,25	**97,75**	59,00
F1-score (%)	96,25	**97,74**	58,22

5.4.2.5 Skenario Dropout

Pada skenario *dropout*, terdapat empat *dropout* yang akan diujikan, yakni 0,1, 0,3, 0,5, 0,7 dan 0,9. Pada skenario perbandingan ini, digunakan *optimizer* Adam dengan L2 *regularizer* serta *learning rate* 0,001 dengan arsitektur Lexicon CNN. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.9.

Tabel 5.9 Perbandingan Hasil Evaluasi pada Uji Coba *Dropout* pada Perhitungan Karakter CNN

	0,1	**0,3**	**0,5**	**0,7**	**0,9**
Lama Prediksi (detik)	**12,03**	12,06	11,88	11,68	11,47
Akurasi (%)	**98,56**	98,19	97,75	95,69	94,19
Presisi (%)	**98,57**	98,21	97,76	95,97	94,26
Recall (%)	**98,56**	98,19	97,75	95,69	94,19
F1-score (%)	**98,56**	98,18	97,74	95,97	94,18

5.4.3 Skenario Uji Coba pada Prediksi Karakter FCN

Pada Prediksi Karakter FCN, terdapat tiga skenario uji coba, antara lain:

1. Skenario uji coba pada *optimizer*.
2. Skenario uji coba pada *learning rate*.
3. Skenario uji coba pada *dropout*.

Pada skenario uji coba ini digunakan dataset dari data validasi sebesar 10% dari dataset atau sebesar 9.096 buah citra.

5.4.3.1 Skenario Optimizer

Pada skenario *optimizer*, terdapat tiga *optimizer* yang akan diujikan, yakni Adam, SGD, dan RMSProp. Pada skenario perbandingan ini, digunakan *learning rate* sebesar 0,01 serta dropout 0,5. Hasil skenario uji coba ini dapat dilihat pada

Tabel 5.10.

Tabel 5.10 Perbandingan Hasil Evaluasi pada Uji Coba *Optimizer* pada Prediksi Karakter FCN

	SGD	**Adam**	**RMSProp**
Lama Prediksi (detik)	207,65	**224,10**	225,38
Akurasi (%)	81,87	**82,55**	81,45
Presisi (%)	82,36	**83,81**	81,92
Recall (%)	81,87	**82,55**	81,45
F1-score (%)	81,79	**82,29**	81,37

5.4.3.2 Skenario Learning Rate

Pada skenario *learning rate*, terdapat tiga *learning rate* yang akan diujikan, yakni 0,01, 0,001, dan 0,0001. Pada skenario perbandingan ini, digunakan *optimizer* Adam serta dropout 0,5. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.11.

Tabel 5.11 Perbandingan Hasil Evaluasi pada Uji Coba *Learning Rate* pada Prediksi Karakter FCN

	0,01	**0,001**	**0,0001**
Lama Prediksi (detik)	224,10	**221,37**	226,95
Akurasi (%)	82,55	**82,86**	82,29
Presisi (%)	83,81	**83,48**	82,96
Recall (%)	82,55	**82,86**	82,29
F1-score (%)	82,29	**82,78**	81,96

5.4.3.3 Skenario Dropout

Pada skenario *dropout*, terdapat empat *dropout* yang akan diujikan, yakni 0,3, 0,5, 0,7 dan 0,9. Pada skenario perbandingan ini, digunakan *optimizer* Adam dengan *learning rate* 0,01. Hasil skenario uji coba ini dapat dilihat pada Tabel 5.12.

Tabel 5.12 Perbandingan Hasil Evaluasi pada Uji Coba *Dropout* pada Prediksi Karakter FCN

	0,3	**0,5**	**0,7**	**0,9**
Lama Prediksi (detik)	214,17	221,37	224,49	**215,58**
Akurasi (%)	82,85	82,86	82,95	**83,52**
Presisi (%)	84,08	83,48	83,51	**84,42**
Recall (%)	82,85	82,86	82,95	**83,52**
F1-score (%)	82,54	82,78	82,90	**83,29**

5.4.4 Skenario Uji Coba Sistem secara Menyeluruh

Pada uji coba sistem secara keseluruhan, akan digunakan citra kata untuk melalui rangkaian *neural network* secara keseluruhan. Pada skenario sistem keseleruhan ini, akan digunakan

model dari masing-masing *neural network* yang menghasilkan akurasi terbaik.

Untuk Lexicon CNN, model yang akan dipakai merupakan model yang menggunakan *optimizer* Adam dengan *learning rate* 0,001 dan L2 *regularizer*. Untuk Perhitungan Karakter CNN, model yang akan dipakai merupakan model yang menggunakan aristektur *neural network* seperti pada Lexicon CNN dan *optimizer* adam dengan *learning rate* 0,001, *dropout* 0,3, dan L2 *regularizer*. Sedangkan unutk model Prediksi Karakter FCN, model yang digunakan merupakan model yang menggunakan *optimizer* Adam, *learning rate* sebesar 0,001, dan *dropout* sebesar 0,9

Uji coba ini dilakukan menggunakan data uji dengan jumlah 609 citra. terdapat empat skenario uji coba, antara lain:

1. Skenario uji coba pada penggunaan Lexicon CNN.
2. Skenario uji coba pada besaran *sliding window*.
3. Skenario uji coba pada *confidence level* Lexicon CNN.

5.4.4.1 Skenario Penggunaan Lexicon CNN

Pada skenario penggunaan Lexicon CNN, akan dilakukan perbandingan pada saat sistem tidak menggunakan Lexicon CNN dan pada saat menggunakannya. Hasil skenario uji coba dapat dilihat pada

Tabel 5.13.

Tabel 5.13 Perbandingan Hasil Evaluasi pada Uji Coba Penggunaan Lexicon CNN pada Keseluruhan Sistem

	Tanpa Lexicon	**Dengan Lexicon**
Lama Prediksi (detik)	2472,23	**1812,49**
Akurasi (%)	4,43	**57,31**
Presisi (%)	12,36	**83,17**
Recall (%)	4,43	**57,31**
F1-score (%)	5,49	**64,98**
Rata-Rata CER(%)	63,08	**29,86**

5.4.4.2 Skenario Besaran *Sliding Window*

Pada skenario penggunaan besaran *sliding window*, akan dilakukan perbandingan pada saat sistem saat *sliding window* sebesar 32x16 dan sebesar 32x32. Hasil skenario uji coba dapat dilihat pada Tabel 5.14.

Tabel 5.14 Perbandingan Hasil Evaluasi pada Uji Coba Besaran *Sliding Window*

	Sliding window 32x16	**Sliding window 32x32**
Lama Prediksi (detik)	**1812,49**	2647,30
Akurasi (%)	**57,31**	56,91
Presisi (%)	**83,17**	82,59
Recall (%)	**57,31**	56,98
F1-score (%)	**64,98**	64,67
Rata-Rata CER(%)	**29,86**	35,19

5.4.4.3 Skenario *Confidence Level* Lexicon CNN

Pada skenario *confidence level*, akan dilakukan perbandingan pada saat sistem saat menggunakan *confidence level* sebesar 65%, 70%, dan 75%. Hasil skenario uji coba dapat dilihat pada Tabel 5.15.

Tabel 5.15 Perbandingan Hasil Evaluasi pada Uji Coba *Confidence Level* pada Sistem Keseluruhan.

	65%	**70%**	**75%**
Lama Prediksi (detik)	**1912,05**	1812,49	2170,47
Akurasi (%)	**68,80**	57,31	46,80
Presisi (%)	**83,45**	83,17	79,65
Recall (%)	**68,80**	57,31	46,80
F1-score (%)	**73,97**	64,98	55,60
Rata-Rata CER(%)	**22,25**	29,86	36,81

5.5 Hasil dan Evaluasi

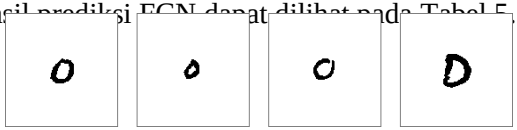
Pada sistem Lexicon CNN, akurasi terbaik dan rata-rata CER terbaik dihasilkan pada penggunaan L2 *regularizer*, *optimizer* Adam, dan *learning rate* sebesar 0,001 dalam mengenali citra berdasarkan kamus kata. Akurasi terbaik yang dihasilkan dari penggunaan parameter-parameter ialah sebesar 99,98% dan rata-rata CER sebesar 0,01%. Hal ini menunjukkan L2 *regularizer* mampu memberikan pinalti yang lebih baik pada *loss function* dan Adam tepat digunakan untuk arsitektur CNN karena menggunakan *adaptive learning rate* yang cenderung lebih cepat konvergen serta menghasilkan performa yang lebih baik dibandingkan dengan *optimizer* yang menggunakan *learning rate* statis.

Pada Perhitungan Karakter CNN, sistem lebih mampu memberikan akurasi yang baik pada saat menggunakan arsitektur CNN yang sama seperti pada Lexicon CNN. Hal ini menunjukkan bahwa fungsi aktivasi *maxout* terlalu banyak mengolah *neuron* sehingga terjadi *overfitting* serta menyebabkan proses konvolusi memakan waktu lebih banyak. Penggunaan ReLU dirasa sudah cukup karena ReLU akan memberi nilai pada semua neuron yang memiliki fungsi aktivasi bernilai negatif, sehingga neuron tersebut dianggap sudah tidak aktif.

Pada Perhitungan Karakter CNN ini, akurasi terbaik didapat pada penggunaan L2 *regularizer* dengan *optimizer* Adam, *learning rate* 0,001 dan *dropout* 0,1. Akurasi final yang dihasilkan ialah 98,56%. Hal ini menunjukkan L2 *regularizer* mampu memberikan pinalti yang lebih baik pada *loss function* dan Adam tepat digunakan untuk arsitektur CNN karena menggunakan *adaptive learning rate* yang cenderung lebih cepat konvergen serta menghasilkan performa yang lebih baik dibandingkan dengan *optimizer* yang menggunakan *learning rate* statis. *Learning rate* yang digunakan juga mampu membantu model melakukan koreksi bobot yang optimal pada saat *training*. Besaran *dropout* pun memberi pengaruh pada besar akurasi. Hal ini dapat dilihat bahwa semakin kecil *dropout*, semakin baik akurasi yang didapat pada model.

Pada Prediksi Karakter FCN, akurasi terbaik dihasilkan pada penggunaan *optimizer* Adam, *learning rate* 0,001 dengan L2 *regularizer* dan *dropout* sebesar 0,9. Akurasi terbaik dihasilkan sebesar 83,52%. Berkebalikan dengan Perhitungan Karakter CNN, semakin kecil *dropout*, semakin baik akurasi yang didapat pada model.

Akurasi prediksi karakter yang tidak cukup tinggi ini terjadi karena FCN belum mampu membedakan huruf besar, huruf kecil, dan angka dengan baik. Beberapa contoh *confusion matrix* dari hasil prediksi FCN dapat dilihat pada Tabel 5.16 dan



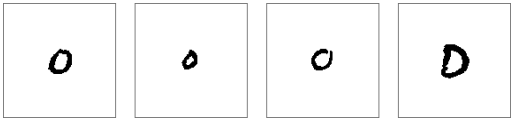
Gambar 5.2 Contoh Citra Karakter Kelas '0', 'O', 'o', dan 'D"

.

Pada Tabel 5.16 ditampilkan hasil prediksi dari pengenalan kelas karakter ‘0’. Hasilnya karakter kelas ‘0’ ini dikenali ke 3 kelas karakter lain, yaitu karakter ‘D’, ‘O’, dan ‘o’. Hal ini terjadi karena keempat karakter ini memiliki kemiripan pada penulisan tangan sehingga sistem kesulitan mebedakan empat kelas karakter ini. Contoh kelas karakter empat kelas karakter ini dapat dilihat pada Gambar 5.2Gambar 5.3.

Tabel 5.16 Confusion Matrix pada Data dengan Kelas '0'

	Hasil Prediksi			
	0	**D**	**O**	**o**
0	56	2	14	75

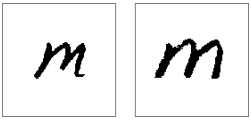


Gambar 5.2 Contoh Citra Karakter Kelas '0', 'O', 'o', dan 'D"

Pada Tabel 5.16 ditampilkan hasil prediksi dari pengenalan kelas karakter ‘m’. Hasilnya karakter kelas ‘m’ ini dikenali ke 1 kelas karakter lain, yaitu karakter ‘M’. Hal ini terjadi karena memang karakter ‘M’ dan ‘m’ tidak memiliki perbedaan yang signifikan sehingga sistem tidak dapat membedakan keedua kelas karakter ini. Contoh kelas karakter dari karakter ‘M’ dan ‘m’ dapat dilihat pada Gambar 5.3.

Tabel 5.17 Confusion Matrix pada Data dengan Kelas 'm'

	Hasil Prediksi	
	M	**m**
m	74	73



Gambar 5.3 Contoh Citra Karakter Kelas 'M' dan 'm"

Pada uji coba sistem secara keseluruhan, akurasi terbaik yang dapat dihasilkan ialah sebesar 68,80% dengan rata-rata CER 22,25% pada penggunaan Lexicon CNN dengan *confidence level* sebesar 65%. Akurasi terbaik ini dilakukan menggunakan pembacaan *sliding window* sebesar 32x16.

Pada pengujian tanpa menggunakan Lexicon CNN, akurasi yang dihasilkan sangatlah rendah dan rata-rata CER yang dihasilkan cukup tinggi. Sistem hanya dapat mengenali karakter dengan panjang maksimal 3, meskipun hasil prediksi perhitungan karakter cukup tinggi, yaitu sebesar 88,34%. Hal ini disebabkan Prediksi Karakter FCN masih lemah dalam memprediksi *blank prediction* pada data uji sehingga pembacaan karakter menggunakan *sliding windows* masih sering terjadi duplikasi, penambahan, dan penghapusan karakter. Dari sini juga dapat

dilihat bahwa pengenalan teks tulisan tangan dengan Lexicon CNN dapat sangat membantu sistem dalam mengenali tulisan tangan.

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan yang didasari oleh hasil uji coba yang telah dilakukan pada bab sebelumnya. Kesimpulan nantinya sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut di masa depan.

6.1 Kesimpulan

Dalam pengerjaan Tugas Akhir ini setelah melalui tahap perancangan aplikasi, implementasi metode, serta uji coba, diperoleh kesimpulan sebagai berikut:

1. Berdasarkan uji coba parameter pada sistem Lexicon CNN, model yang dibangun menghasilkan akurasi yang paling baik sebesar 99,98% dengan rata-rata CER 0,01%.
2. Berdasarkan uji coba parameter pada sistem Perhitungan Karakter CNN, model yang dibangun menghasilkan akurasi terbaik sebesar 98,56%.
3. Berdasarkan uji coba parameter pada sistem Prediksi Karakter FCN, model yang dibangun menghasilkan akurasi terbaik sebesar 83,52% untuk pengenalan 62 kelas. Model Prediksi Karakter FCN ini belum mampu memprediksi karakter dengan baik, karena masih belum dapat membedakan beberapa huruf besar, huruf kecil, dan angka.
4. Berdasarkan uji coba pada satu kesatuan sistem, sistem hanya mampu menghasilkan akurasi sebesar 68,80% dengan rata-rata CER sebesar 22,25%. Hal ini menunjukkan algoritma *sliding windows* belum mampu meminimalisir penambahan, penggantian, dan penghapusan karakter dalam pembacaan citra tulisan tangan. Selain itu disini dapat dilihat bahwa pengenalan berdasarkan kamus kata memang sangat membantu dalam pengenalan teks tulisan tangan.

6.2 Saran

Saran yang diberikan untuk pengembangan sistem pengenalan tulisan tangan menggunakan *Fully Convolutional Neural Network* , yaitu:

1. Pengembangan sistem pada bagian praproses citra masukan agar mampu menerima citra yang lebih bervariasi, seperti praproses pada tulisan latin, praproses pada tulisan yang tidak menggunakan tinta bewarna hitam, serta tulisan tangan *offline* atau tidak ditulis pada layar sentuh.
2. Pengembangan sistem yang memungkinkan untuk mengenali citra kalimat atau bahkan paragraf. Salah satu metode yang dapat digunakan dalam proses pemotongan dari kalimat atau paragraf ke dalam kita ialah menggunakan R-CNN.
3. Pengembangan sistem pada *neural network* Prediksi Karakter yang mampu memisahkan huruf kapital, huruf kecil, dan angka dengan baik untuk menghasilkan akurasi yang lebih akurat.
4. Pengembangan sistem pada bagian pembacaan karakter demi karakter untuk menghindari duplikasi, penambahan, atau penghilangan karakter dari pembacaan kata.

DAFTAR PUSTAKA

[1] R. Ptucha, F. P. Such, S. Pilla, F. Brockler, V. Singh dan P. Hutkowski, "Intelligent Character Recognition using Fully Convolutional Neural Network," *Pattern Recognition,* vol. 88, pp. 604-613, 2018.

[2] National Institute of Standards and Technology, US Department of Commerce, "NIST Special Database 19," 2019. [Online]. Available: https://www.nist.gov/srd/nist-special-database-19. [Diakses 6 Juni 2020].

[3] Bird, Steven, E. Loper dan E. Klei, Natural Language Processing with Python, O'Reilly Media Inc, 2009.

[4] "Digital Image Processing Basics," GeeksforGeeks, 2018. [Online]. Available: https://www.geeksforgeeks.org/digital-image-processing-basics/. [Diakses 14 Juni 2020].

[5] S. Kapur, Computer Vision with Python 3, Birmingham, UK: Packt Publishing Ltd., 2017.

[6] J. Jeong, "Computer Vision for Beginners: Part 4," Towards Data Science, 23 April 2019. [Online]. Available: https://towardsdatascience.com/computer-vision-for-beginners-part-4-64a8d9856208. [Diakses 20 Juni 2020].

[7] T. Flores, "Gaussian Blurring with Python and OpenCV," Medium, 23 Mei 2019. [Online]. Available: https://medium.com/analytics-vidhya/gaussian-blurring-with-python-and-opencv-ba8429eb879b. [Diakses 20 Juni 2020].

[8] B. Yuwono, "Image Smoothing Menggunakan Mean Filtering, Median Filtering, Modus Filtering, dan Gaussian Filtering," *Telematika,* vol. 7, 2015.

[9] D. Jurafsky dan J. H. Martin, “Speech and Language Processing,” 2019. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/. [Diakses 4 June 2020].

[10 “Convolutional Neural Network,” MathWorks, 2018. [Online]. Available: https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html#matlab. [Diakses 8 Juni 2020].

[11 M. Zufar dan B. Setiyono, “Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time,” *Jurnal Sains dan Seni ITS,* vol. 5, pp. 72-73, 2016.

[12 S. F. Umayah, “Penerapan Pengolahan Citra Menggunakan Metode Deep learning untuk Mendeteksi Kecacatan Permukaan Buah Manggis,” Yogyakarta, 2017.

[13 D. S. Gupta, “Architecture of Convolutional Neural Networks(CNN),” Analythics Vidhya, 29 Juni 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/architecture-of-convolutional-neural-networks-simplified-demystified/. [Diakses 20 Juni 2020].

[14 “What are Max Pooling, Average Pooling, Global Max Pooling, and Global Average Pooling,” Machine Curve, 30 Januari 2020. [Online]. Available: https://www.machinecurve.com/index.php/2020/01/30/what-are-max-pooling-average-pooling-global-max-pooling-and-global-average-pooling/. [Diakses 23 Juni 2020].

[15 P. Jain, “Complete Guide of Activation Function,” Towards Data Science, 2019. [Online]. Available: https://towardsdatascience.com/complete-guide-of-activation-functions-34076e95d044. [Diakses 15 Juni 2020].

[16 G. Hinton, Neural Networks for Machine Learning.

[17 S. Ryder, An Overview of Gradient Descent Optimization Algorithm, 19 Januari 2016. [Online]. Available: https://ruder.io/optimizing-gradient-descent/index.htm. [Diakses 23 Juni 2020].

[18 "A Gentle Introduction to Dropout for Regularizing Deep Neural Network," Machine Learning Mastery, 3 Desember 2018. [Online]. Available: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/. [Diakses 23 Juni 2020].

[19 N. S. G. Hinton, A. Krizhevsky, I. Sutskever dan R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research,* vol. 15, pp. 1929-1958, 2014.

[20 J. D. Seo, "Deeper Understanding of Batch Normalization with Interactive Code in Tensorflow [Manual Back Propagation]," Medium, 2018. [Online]. Available: https://medium.com/@SeoJaeDuk/deeper-understanding-of-batch-normalization-with-interactive-code-in-tensorflow-manual-back-1d50d6903d35. [Diakses 17 Juni 2020].

[21 A. Deshpande, "A Beginner's Guide To Understanding Convolutional Neural Networks Part 2," Github, 29 Juli 2016. [Online]. Available: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/. [Diakses 20 Juni 2020].

[22 W. Di, A. Bhardwaj dan J. Wei, Deep Learning Essentials, Birmingham: Packt Publishing Ltd., 2018.

[23 E. Shelhamer, J. Long dan T. Darrel, “Fully Convolutional Networks for Semantic Segmentation,” *arXiv:1605.06211v1[cs.CV],* 2016.

[24 S. Narkhede, “Understanding Confusion matrix,” Towards Data Science, 9 Mei 2018. [Online]. Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62. [Diakses 20 Juni 2020].

[25 “About Python,” Python, 2014. [Online]. Available: https://www.python.org/about/. [Diakses 19 Mei 2020].

[26 “Keras: The Python Deep Learning Library,” Keras, 2020. [Online]. Available: https://keras.io/. [Diakses 19 Mei 2020].

[27 “Tensorflow,” Tensorflow, 2020. [Online]. Available: https://www.tensorflow.org/. [Diakses 19 Mei 2020].

[28 “OpenCV,” OpenCV, 2017. [Online]. Available: https://opencv.org/. [Diakses 19 Mei 2020].

[29 “NumPy,” NumPy, 2018. [Online]. Available: http://www.numpy.org/. [Diakses 19 Mei 2020].

[30 L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt dan G. Varoquaux, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research,* vol. 12, pp. 2825-2830, 2011.

[31 “Matplotlib,” Matplotib, 2018. [Online]. Available: https://matplotlib.org/index.html. [Diakses 19 Mei 2020].

[32 A. Dalmia, “Implementation of Maxout activation after Convolution,” India, 2017.

[33 G. Hinton, *Neural Networks for Machine Learning.*

LAMPIRAN

L.1 Daftar Kata yang Paling Sering Muncul pada Corpus Brown

No	Kata	Frekuensi Kemunculan
1.	the	70.003
2.	of	36.473
3.	and	28.935
4.	to	26.247
5.	a	23.510
6.	in	21.419
7.	that	10,596
8.	is	10.109
9.	was	9.815
10.	he	9.548
11.	for	9.500
12.	it	8.771
13.	with	7.290
14.	as	7.261
15.	his	6.999
16.	on	6.765
17.	be	6.388
18.	at	5.377
19.	by	5.346
20.	i	5.253
21.	this	5.145
22.	had	5.133
23.	not	4.620
24.	are	4.394
25.	but	4.382
26.	from	4.371
27.	or	4.209
28.	have	3.942
29.	an	3.749

30.	they	3.620
31.	which	3.561
32.	one	3.439
33.	were	3.346
34.	you	3.287
35.	all	3.094
36.	her	3.036
37.	she	2.860
38.	there	2.728
39.	would	2.719
40.	their	2.669
41.	we	2.652
42.	him	2.619
43	been	2.472
44.	has	2.437
45.	when	2.331
46.	who	2.252
47.	will	2.251
48.	no	2.219
49.	more	2.217
50.	if	2.198

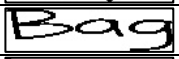
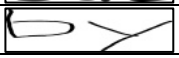
L.2 Daftar Kata yang Paling Sering Muncul pada Corpus Gutenberg

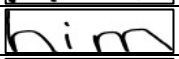
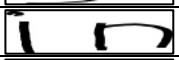
No	Kata	Frekuensi Kemunculan
1.	the	133.561
2.	and	95.376
3.	of	71.254
4.	to	48.048
5.	a	33.922
6.	in	33.551
7.	i	28.978
8.	that	28.416

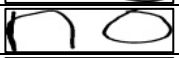
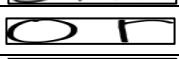
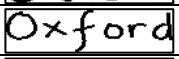
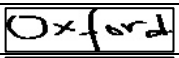
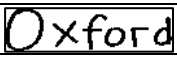
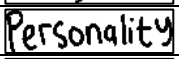
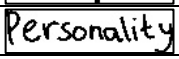
9.	he	25.556
10.	it	21.841
11.	his	21.410
12.	for	19.509
13.	was	18.711
14.	with	17.596
15.	not	17.369
16.	is	16.406
17.	be	16.113
18.	you	16.054
19.	as	14.520
20.	but	13.898
21.	all	13.719
22.	they	13.027
23.	him	13.026
24.	shall	11.681
25.	her	11.557
26.	my	10,505
27.	had	10,321
28.	them	10.249
29.	have	10.113
30.	me	9. 474
31.	said	9.429
32.	at	9.172
33.	from	9.073
34.	she	9.040
35.	unto	9.010
36.	this	8.908
37.	which	8.771
38.	on	8.579
39.	by	8.509
40.	lord	8.446
41.	their	7.791
42.	so	7.782

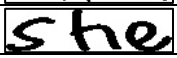
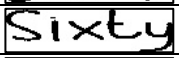
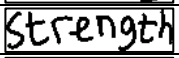
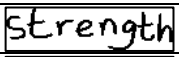
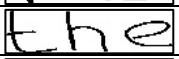
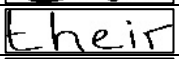
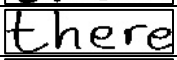
43	will	7.369
44.	were	6.870
45.	are	6.845
46.	thou	6.751
47.	when	6.323
48.	or	6.315
49.	one	6.132
50.	there	6.122

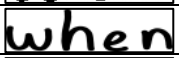
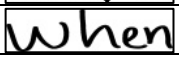
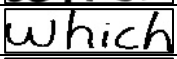
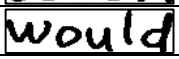
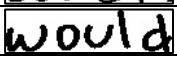
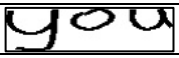
L.3 Contoh Data Uji

Kata	Contoh Citra 1	Contoh Citra 2	Contoh Citra 3
a			
all			
an			
and			
are			
as			
at			
BAG		-	-
Bag		-	-
bag		-	-
beginners			
be			
been			
but			
by			

for			
frogs		-	-
FROM		-	-
From		-	-
from			
goblin		-	-
gymnasts		-	-
had			
has			
have			
he			
her			
him			
his			
How		-	-
I			
if			
in			
is			
it			
Jump			-
jump			
jumping		-	-
jumps		-	-

JUTE		-	-
Jute		-	-
jute		-	-
lord			
me			
more			
my			
no			
not			
of			
on			
one			
or			
over			
Oxford			
personality			-
PICKED		-	-
Picked		-	-
picked		-	-
piqued		-	-
quick			
Quick		-	-
QUICKLY		-	-
Quickly		-	-

quickly		-	-
razorback		-	-
said			
shall			
she			
six		-	-
SIXTY		-	-
Sixty		-	-
sixty		-	-
strength			
so			
that			
THE		-	-
the			
their			
them			
there			
they			
this			
thou			
till		-	-
to			
ubiquitary			
unto			

vow			
was			
We		-	-
we			
WERE		-	-
Were		-	-
were			
when			
which			
who			
will			
with			
would			
WOVEN		-	-
Woven		-	-
woven		-	-
you			
ZIPPERS		-	-
Zippers		-	-
zippers		-	-

L.4 Hasil Uji Coba Parameter L1 *Regularizer* pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9709	1,0000	0,9852	100
all	1,0000	1,0000	1,0000	100
an	1,0000	0,9800	0,9899	100
and	1,0000	1,0000	1,0000	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	0,9900	0,9950	100
be	1,0000	0,9800	0,9899	100
been	1,0000	1,0000	1,0000	100
but	1,0000	1,0000	1,0000	100
by	1,0000	0,9900	0,9950	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	0,9800	0,9899	100
has	0,9804	1,0000	0,9901	100
have	1,0000	1,0000	1,0000	100
he	0,9804	1,0000	0,9901	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	1,0000	1,0000	100
i	1,0000	1,0000	1,0000	100
if	1,0000	1,0000	1,0000	100
in	0,9900	0,9900	0,9900	100
is	1,0000	1,0000	1,0000	100
it	0,9900	0,9900	0,9900	100
lord	1,0000	1,0000	1,0000	100
me	0,9900	0,9900	0,9900	100
more	1,0000	1,0000	1,0000	100
my	1,0000	1,0000	1,0000	100
no	1,0000	1,0000	1,0000	100

not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	0,9901	1,0000	0,9950	100
one	1,0000	0,9900	0,9950	100
or	0,9901	1,0000	0,9950	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	0,9900	0,9950	100
the	0,9901	1,0000	0,9950	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	0,9901	1,0000	0,9950	100
this	1,0000	0,9900	0,9950	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9977	0,9976	0,9976	5900

L.5 Hasil Uji Coba Parameter L2 *Regularizer* pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9901	1,0000	0,9950	100
all	1,0000	1,0000	1,0000	100
an	1,0000	1,0000	1,0000	100
and	1,0000	1,0000	1,0000	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	1,0000	1,0000	100
be	1,0000	0,9900	0,9950	100
been	1,0000	1,0000	1,0000	100
but	1,0000	1,0000	1,0000	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	1,0000	1,0000	1,0000	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	0,9900	0,9950	100
i	1,0000	1,0000	1,0000	100
if	1,0000	0,9900	0,9950	100
in	1,0000	1,0000	1,0000	100
is	0,9901	1,0000	0,9950	100
it	1,0000	1,0000	1,0000	100
lord	1,0000	1,0000	1,0000	100
me	0,9901	1,0000	0,9950	100
more	1,0000	1,0000	1,0000	100
my	0,9901	1,0000	0,9950	100
no	1,0000	1,0000	1,0000	100

not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	1,0000	1,0000	1,0000	100
one	1,0000	0,9900	0,9950	100
or	1,0000	1,0000	1,0000	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	1,0000	1,0000	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9993	0,9993	0,9993	5900

L.6 Hasil Uji Coba Parameter *Optimizer* SGD pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9900	0,9900	0,9900	100
all	1,0000	1,0000	1,0000	100
an	0,9900	0,9900	0,9900	100
and	0,9901	1,0000	0,9950	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	0,9800	0,9899	100
be	1,0000	0,9700	0,9848	100
been	1,0000	0,9900	0,9950	100
but	1,0000	0,9900	0,9950	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	0,9804	1,0000	0,9901	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	1,0000	1,0000	100
i	1,0000	1,0000	1,0000	100
if	1,0000	0,9900	0,9950	100
in	1,0000	1,0000	1,0000	100
is	0,9901	1,0000	0,9950	100
it	1,0000	1,0000	1,0000	100
lord	1,0000	1,0000	1,0000	100
me	1,0000	1,0000	1,0000	100
more	1,0000	1,0000	1,0000	100

my	1,0000	1,0000	1,0000	100
no	1,0000	0,9900	0,9950	100
not	0,9901	1,0000	0,9950	100
of	1,0000	1,0000	1,0000	100
on	0,9901	1,0000	0,9950	100
one	1,0000	1,0000	1,0000	100
or	0,9804	1,0000	0,9901	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	0,9901	1,0000	0,9950	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100

Weighted Average	0,9982	0,9981	0,9981	5900

L.7 Hasil Uji Coba Parameter *Optimizer* Adam pada Lexicon CNN

Daftar Kata	**Precision**	**Recall**	**F1-Score**	**Support**
a	0,9901	1,0000	0,9950	100
all	1,0000	1,0000	1,0000	100
an	1,0000	1,0000	1,0000	100
and	1,0000	1,0000	1,0000	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	1,0000	1,0000	100
be	1,0000	0,9900	0,9950	100
been	1,0000	1,0000	1,0000	100
but	1,0000	1,0000	1,0000	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	1,0000	1,0000	1,0000	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	0,9900	0,9950	100
i	1,0000	1,0000	1,0000	100
if	1,0000	0,9900	0,9950	100
in	1,0000	1,0000	1,0000	100
is	0,9901	1,0000	0,9950	100
it	1,0000	1,0000	1,0000	100
lord	1,0000	1,0000	1,0000	100
me	0,9901	1,0000	0,9950	100

more	1,0000	1,0000	1,0000	100
my	0,9901	1,0000	0,9950	100
no	1,0000	1,0000	1,0000	100
not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	1,0000	1,0000	1,0000	100
one	1,0000	0,9900	0,9950	100
or	1,0000	1,0000	1,0000	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	1,0000	1,0000	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9993	0,9993	0,9993	5900

L.8 Hasil Uji Coba Parameter *Optimizer* RMSProp pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9800	0,9800	0,9800	100
all	1,0000	0,9900	0,9950	100
an	0,9802	0,9900	0,9851	100
and	0,9901	1,0000	0,9950	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	0,9434	1,0000	0,9709	100
be	0,9900	0,9900	0,9900	100
been	1,0000	0,9900	0,9950	100
but	1,0000	1,0000	1,0000	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	1,0000	1,0000	1,0000	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	1,0000	1,0000	100
i	1,0000	0,9900	0,9950	100
if	1,0000	1,0000	1,0000	100
in	1,0000	1,0000	1,0000	100
is	1,0000	1,0000	1,0000	100
it	1,0000	0,9800	0,9899	100
lord	1,0000	1,0000	1,0000	100
me	0,9901	1,0000	0,9950	100
more	1,0000	1,0000	1,0000	100
my	1,0000	1,0000	1,0000	100
no	1,0000	1,0000	1,0000	100

not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	1,0000	0,9900	0,9950	100
one	1,0000	0,9900	0,9950	100
or	1,0000	0,9900	0,9950	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	1,0000	1,0000	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	0,9900	0,9950	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9979	0,9978	0,9978	5900

L.9 Hasil Uji Coba Parameter *Learning Rate* 0,01 pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9901	1,0000	0,9950	100
all	1,0000	1,0000	1,0000	100
an	1,0000	1,0000	1,0000	100
and	1,0000	1,0000	1,0000	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	1,0000	1,0000	100
be	1,0000	0,9900	0,9950	100
been	1,0000	1,0000	1,0000	100
but	1,0000	1,0000	1,0000	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	1,0000	1,0000	1,0000	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	0,9900	0,9950	100
i	1,0000	1,0000	1,0000	100
if	1,0000	0,9900	0,9950	100
in	1,0000	1,0000	1,0000	100
is	0,9901	1,0000	0,9950	100
it	1,0000	1,0000	1,0000	100
lord	1,0000	1,0000	1,0000	100
me	0,9901	1,0000	0,9950	100
more	1,0000	1,0000	1,0000	100
my	0,9901	1,0000	0,9950	100
no	1,0000	1,0000	1,0000	100

not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	1,0000	1,0000	1,0000	100
one	1,0000	0,9900	0,9950	100
or	1,0000	1,0000	1,0000	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	1,0000	1,0000	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9993	0,9993	0,9993	5900

L.10 Hasil Uji Coba Parameter *Learning Rate* 0,001 pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9901	1,0000	0,9950	100
all	1,0000	1,0000	1,0000	100
an	1,0000	1,0000	1,0000	100
and	1,0000	1,0000	1,0000	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	1,0000	1,0000	100
be	1,0000	0,9900	0,9950	100
been	1,0000	1,0000	1,0000	100
but	1,0000	1,0000	1,0000	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	1,0000	1,0000	1,0000	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	1,0000	1,0000	100
i	1,0000	1,0000	1,0000	100
if	1,0000	1,0000	1,0000	100
in	1,0000	1,0000	1,0000	100
is	1,0000	1,0000	1,0000	100
it	1,0000	1,0000	1,0000	100
lord	1,0000	1,0000	1,0000	100
me	1,0000	1,0000	1,0000	100
more	1,0000	1,0000	1,0000	100
my	1,0000	1,0000	1,0000	100
no	1,0000	1,0000	1,0000	100

not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	1,0000	1,0000	1,0000	100
one	1,0000	1,0000	1,0000	100
or	1,0000	1,0000	1,0000	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	1,0000	1,0000	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9998	0,9998	0,9998	5900

L.11 Hasil Uji Coba Parameter *Learning Rate* 0,0001 pada Lexicon CNN

Daftar Kata	Precision	Recall	F1-Score	Support
a	0,9804	1,0000	0,9901	100
all	1,0000	1,0000	1,0000	100
an	1,0000	1,0000	1,0000	100
and	1,0000	1,0000	1,0000	100
are	1,0000	1,0000	1,0000	100
as	1,0000	1,0000	1,0000	100
at	1,0000	1,0000	1,0000	100
be	1,0000	0,9900	0,9950	100
been	1,0000	1,0000	1,0000	100
but	1,0000	1,0000	1,0000	100
by	1,0000	1,0000	1,0000	100
for	1,0000	1,0000	1,0000	100
from	1,0000	1,0000	1,0000	100
had	1,0000	1,0000	1,0000	100
has	1,0000	1,0000	1,0000	100
have	1,0000	1,0000	1,0000	100
he	1,0000	1,0000	1,0000	100
her	1,0000	1,0000	1,0000	100
him	1,0000	1,0000	1,0000	100
his	1,0000	1,0000	1,0000	100
i	1,0000	0,9900	0,9950	100
if	1,0000	1,0000	1,0000	100
in	1,0000	1,0000	1,0000	100
is	1,0000	1,0000	1,0000	100
it	1,0000	1,0000	1,0000	100
lord	1,0000	1,0000	1,0000	100
me	1,0000	1,0000	1,0000	100
more	1,0000	1,0000	1,0000	100
my	1,0000	1,0000	1,0000	100
no	1,0000	1,0000	1,0000	100

not	1,0000	1,0000	1,0000	100
of	1,0000	1,0000	1,0000	100
on	1,0000	1,0000	1,0000	100
one	1,0000	1,0000	1,0000	100
or	1,0000	1,0000	1,0000	100
said	1,0000	1,0000	1,0000	100
shall	1,0000	1,0000	1,0000	100
she	1,0000	1,0000	1,0000	100
so	1,0000	1,0000	1,0000	100
that	1,0000	1,0000	1,0000	100
the	1,0000	1,0000	1,0000	100
their	1,0000	1,0000	1,0000	100
them	1,0000	1,0000	1,0000	100
there	1,0000	1,0000	1,0000	100
they	1,0000	1,0000	1,0000	100
this	1,0000	1,0000	1,0000	100
thou	1,0000	1,0000	1,0000	100
to	1,0000	1,0000	1,0000	100
unto	1,0000	1,0000	1,0000	100
was	1,0000	1,0000	1,0000	100
we	1,0000	1,0000	1,0000	100
were	1,0000	1,0000	1,0000	100
when	1,0000	1,0000	1,0000	100
which	1,0000	1,0000	1,0000	100
who	1,0000	1,0000	1,0000	100
will	1,0000	1,0000	1,0000	100
with	1,0000	1,0000	1,0000	100
would	1,0000	1,0000	1,0000	100
you	1,0000	1,0000	1,0000	100
Weighted Average	0,9997	0,9997	0,9997	5900

L.12 Hasil Uji Coba Penggunaan Arsitektur Perhitungan Karakter CNN pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	0,0000	0,0000	0,0000	100
2	0,0000	0,0000	0,0000	100
3	0,0625	1,000	0,1176	100
4	0,0000	0,0000	0,0000	100
5	0,0000	0,0000	0,0000	100
6	0,0000	0,0000	0,0000	100
7	0,0000	0,0000	0,0000	100
8	0,0000	0,0000	0,0000	100
9	0,0000	0,0000	0,0000	100
10	0,0000	0,0000	0,0000	100
11	0,0000	0,0000	0,0000	100
12	0,0000	0,0000	0,0000	100
13	0,0000	0,0000	0,0000	100
14	0,0000	0,0000	0,0000	100
15	0,0000	0,0000	0,0000	100
16	0,0000	0,0000	0,0000	100
Weighted Average	0,0039	0,0625	0,0074	1600

L.13 Hasil Uji Coba Penggunaan Arsitektur Lexicon CNN pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	0,9900	0,9900	0,9900	100
4	0,9697	0,9600	0,9648	100
5	0,9515	0,9800	0,9655	100
6	0,9899	0,9800	0,9849	100

7	0,9802	0,9900	0,9851	100
8	0,9894	0,9300	0,9588	100
9	0,9394	0,9300	0,9347	100
10	0,9159	0,9800	0,9469	100
11	0,9789	0,9300	0,9538	100
12	0,9423	0,9800	0,9608	100
13	0,9592	0,9400	0,9495	100
14	0,8807	0,9600	0,9187	100
15	0,9565	0,8800	0,9167	100
16	0,9700	0,9700	0,9700	100
Weighted Average	0,9633	0,9625	0,9625	1600

L.14 Hasil Uji Coba Parameter L1 *Regularizer* pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	0,9901	1,0000	0,9950	100
2	1,0000	0,9800	0,9899	100
3	1,0000	0,9900	0,9950	100
4	0,9802	0,9900	0,9851	100
5	0,9802	0,9900	0,9851	100
6	1,0000	0,9800	0,9899	100
7	0,9434	1,0000	0,9709	100
8	1,0000	0,9300	0,9637	100
9	0,9519	0,9900	0,9706	100
10	0,9780	0,8900	0,9319	100
11	0,9065	0,9700	0,9372	100
12	0,9792	0,9400	0,9592	100
13	0,9307	0,9400	0,9353	100
14	0,8889	0,9600	0,9231	100
15	0,8846	0,9200	0,9020	100
16	0,9783	0,9000	0,9375	100

Weighted Average	0,9620	0,9606	0,9607	1600

L.15 Hasil Uji Coba Parameter L2 *Regularizer* pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	0,9900	0,9900	0,9900	100
4	0,9697	0,9600	0,9648	100
5	0,9515	0,9800	0,9655	100
6	0,9899	0,9800	0,9849	100
7	0,9802	0,9900	0,9851	100
8	0,9894	0,9300	0,9588	100
9	0,9394	0,9300	0,9347	100
10	0,9159	0,9800	0,9469	100
11	0,9789	0,9300	0,9538	100
12	0,9423	0,9800	0,9608	100
13	0,9592	0,9400	0,9495	100
14	0,8807	0,9600	0,9187	100
15	0,9565	0,8800	0,9167	100
16	0,9700	0,9700	0,9700	100
Weighted Average	0,9633	0,9625	0,9625	1600

L.16 Hasil Uji Coba Parameter *Optimizer* SGD pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	0,9804	1,0000	0,9901	100
2	1,0000	0,9800	0,9899	100

3	1,0000	0,9700	0,9848	100
4	0,8981	0,9700	0,9327	100
5	0,8269	0,8600	0,8431	100
6	0,8172	0,7600	0,7876	100
7	0,7027	0,7800	0,7393	100
8	0,6296	0,6800	0,6538	100
9	0,6058	0,6300	0,6176	100
10	0,5934	0,5400	0,5654	100
11	0,5229	0,5700	0,5455	100
12	0,5323	0,3300	0,4074	100
13	0,5327	0,5700	0,5507	100
14	0,4819	0,4000	0,4372	100
15	0,4304	0,3400	0,3799	100
16	0,6042	0,8700	0,7131	100
Weighted Average	0,6974	0,7031	0,6961	1600

L.17 Hasil Uji Coba Parameter *Optimizer* Adam pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	0,9900	0,9900	0,9900	100
4	0,9697	0,9600	0,9648	100
5	0,9515	0,9800	0,9655	100
6	0,9899	0,9800	0,9849	100
7	0,9802	0,9900	0,9851	100
8	0,9894	0,9300	0,9588	100
9	0,9394	0,9300	0,9347	100
10	0,9159	0,9800	0,9469	100
11	0,9789	0,9300	0,9538	100
12	0,9423	0,9800	0,9608	100

13	0,9592	0,9400	0,9495	100
14	0,8807	0,9600	0,9187	100
15	0,9565	0,8800	0,9167	100
16	0,9700	0,9700	0,9700	100
Weighted Average	0,9633	0,9625	0,9625	1600

L.18 Hasil Uji Coba Parameter *Optimizer* RMSProp pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	0,9900	0,9950	100
2	1,0000	1,0000	1,0000	100
3	0,9706	0,9900	0,9802	100
4	0,9796	0,9600	0,9697	100
5	0,9612	0,9900	0,9754	100
6	0,9897	0,9600	0,9746	100
7	0,9700	0,9700	0,9700	100
8	0,9519	0,9900	0,9706	100
9	0,9600	0,9600	0,9600	100
10	0,9787	0,9200	0,9485	100
11	0,9429	0,9900	0,9659	100
12	0,9796	0,9600	0,9697	100
13	0,9468	0,8900	0,9175	100
14	0,8785	0,9400	0,9082	100
15	0,9368	0,8900	0,9128	100
16	0,9327	0,9700	0,9510	100
Weighted Average	0,9612	0,9606	0,9606	1600

L.19 Hasil Uji Coba Parameter *Learning Rate* 0,01 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	0,9900	0,9900	0,9900	100
4	0,9697	0,9600	0,9648	100
5	0,9515	0,9800	0,9655	100
6	0,9899	0,9800	0,9849	100
7	0,9802	0,9900	0,9851	100
8	0,9894	0,9300	0,9588	100
9	0,9394	0,9300	0,9347	100
10	0,9159	0,9800	0,9469	100
11	0,9789	0,9300	0,9538	100
12	0,9423	0,9800	0,9608	100
13	0,9592	0,9400	0,9495	100
14	0,8807	0,9600	0,9187	100
15	0,9565	0,8800	0,9167	100
16	0,9700	0,9700	0,9700	100
Weighted Average	0,9633	0,9625	0,9625	1600

L.20 Hasil Uji Coba Parameter *Learning Rate* 0,001 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	1,0000	0,9900	0,9950	100
4	0,9802	0,9900	0,9851	100
5	0,9898	0,9700	0,9798	100
6	0,9804	1,0000	0,9901	100

7	0,9900	0,9900	0,9900	100
8	0,9800	0,9800	0,9800	100
9	0,9706	0,9900	0,9802	100
10	0,9896	0,9500	0,9694	100
11	0,9612	0,9900	0,9754	100
12	0,9706	0,9900	0,9802	100
13	0,9798	0,9700	0,9749	100
14	0,9423	0,9800	0,9608	100
15	0,9381	0,9100	0,9239	100
16	0,9691	0,9400	0,9543	100
Weighted Average	0,9776	0,9775	0,9774	1600

L.21 Hasil Uji Coba Parameter *Learning Rate* 0,0001 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	0,9900	0,9950	100
2	0,9612	0,9900	0,9754	100
3	0,9300	0,9300	0,9300	100
4	0,8333	0,8500	0,8416	100
5	0,6869	0,6800	0,6834	100
6	0,5960	0,5900	0,5930	100
7	0,6118	0,5200	0,5622	100
8	0,4796	0,4700	0,4747	100
9	0,3739	0,4300	0,4000	100
10	0,4262	0,5200	0,4685	100
11	0,5000	0,1400	0,2188	100
12	0,3790	0,4700	0,4196	100
13	0,4167	0,4500	0,4327	100
14	0,5192	0,2700	0,3553	100
15	0,3590	0,4200	0,3871	100
16	0,4832	0,7200	0,5783	100

Weighted Average	0,5972	0,5900	0,5822	1600

L.22 Hasil Uji Coba Parameter *Dropout* 0,1 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	0,9800	0,9899	100
2	0,9901	1,0000	0,9950	100
3	1,0000	1,0000	1,0000	100
4	0,9804	1,0000	0,9901	100
5	1,0000	0,9900	0,9950	100
6	1,0000	1,0000	1,0000	100
7	0,9901	1,0000	0,9950	100
8	1,0000	0,9800	0,9899	100
9	0,9804	1,0000	0,9901	100
10	0,9800	0,9800	0,9800	100
11	0,9802	0,9900	0,9851	100
12	1,0000	0,9800	0,9899	100
13	0,9706	0,9900	0,9802	100
14	0,9510	0,9700	0,9604	100
15	0,9789	0,9300	0,9538	100
16	0,9703	0,9800	0,9751	100
Weighted Average	0,9857	0,9856	0,9856	1600

L.23 Hasil Uji Coba Parameter *Dropout* 0,3 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100

3	1,0000	0,9900	0,9950	100
4	0,9804	1,0000	0,9901	100
5	1,0000	0,9800	0,9899	100
6	0,9901	1,0000	0,9950	100
7	0,9804	1,0000	0,9901	100
8	1,0000	0,9800	0,9899	100
9	0,9709	1,0000	0,9852	100
10	0,9898	0,9700	0,9798	100
11	0,9899	0,9800	0,9849	100
12	0,9802	0,9900	0,9851	100
13	0,9706	0,9900	0,9802	100
14	0,9238	0,9700	0,9463	100
15	0,9574	0,9000	0,9278	100
16	0,9796	0,9600	0,9697	100
Weighted Average	0,9821	0,9819	0,9818	1600

L.24 Hasil Uji Coba Parameter *Dropout* 0,5 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	1,0000	0,9900	0,9950	100
4	0,9802	0,9900	0,9851	100
5	0,9898	0,9700	0,9798	100
6	0,9804	1,0000	0,9901	100
7	0,9900	0,9900	0,9900	100
8	0,9800	0,9800	0,9800	100
9	0,9706	0,9900	0,9802	100
10	0,9896	0,9500	0,9694	100
11	0,9612	0,9900	0,9754	100

12	0,9706	0,9900	0,9802	100
13	0,9798	0,9700	0,9749	100
14	0,9423	0,9800	0,9608	100
15	0,9381	0,9100	0,9239	100
16	0,9691	0,9400	0,9543	100
Weighted Average	0,9776	0,9775	0,9774	1600

L.25 Hasil Uji Coba Parameter *Dropout* 0,7 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	0,9900	0,9900	0,9900	100
4	0,9083	0,9900	0,9474	100
5	0,9785	0,9100	0,9430	100
6	0,9608	0,9800	0,9703	100
7	0,9796	0,9600	0,9697	100
8	1,0000	0,9700	0,9848	100
9	0,9796	0,9600	0,9697	100
10	0,9412	0,9600	0,9505	100
11	0,9703	0,9800	0,9751	100
12	0,9515	0,9800	0,9655	100
13	0,9880	0,8200	0,8962	100
14	0,8182	0,9900	0,8959	100
15	0,9655	0,8400	0,8984	100
16	0,9245	0,9800	0,9515	100
Weighted Average	0,9597	0,9569	0,9567	1600

L.26 Hasil Uji Coba Parameter *Dropout* 0,9 pada Perhitungan Karakter CNN

Jumlah Karakter	Precision	Recall	F1-Score	Support
1	1,0000	1,0000	1,0000	100
2	1,0000	1,0000	1,0000	100
3	0,9899	0,9800	0,9849	100
4	0,9333	0,9800	0,9561	100
5	0,9896	0,9500	0,9694	100
6	0,9802	0,9900	0,9851	100
7	0,9588	0,9300	0,9442	100
8	0,9135	0,9500	0,9314	100
9	0,9388	0,9200	0,9293	100
10	0,9175	0,8900	0,9036	100
11	0,8889	0,9600	0,9231	100
12	0,9674	0,8900	0,9271	100
13	0,9020	0,9200	0,9109	100
14	0,8942	0,9300	0,9118	100
15	0,9121	0,8300	0,8691	100
16	0,8962	0,9500	0,9223	100
Weighted Average	0,9426	0,9419	0,9418	1600

L.27 Hasil Uji Coba Parameter *Optimizer* SGD pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,5752	0,4422	0,5000	147
1	0,5722	0,7279	0,6407	147
2	0,9708	0,9048	0,9366	147
3	0,9865	0,9932	0,9898	147
4	0,9795	0,9728	0,9761	147
5	0,9139	0,9452	0,9293	146
6	0,9412	0,9796	0,9600	147

7	0,9931	0,9863	0,9897	146
8	1,0000	0,9932	0,9966	147
9	0,7317	0,8219	0,7742	146
A	0,9931	0,9863	0,9897	146
B	0,9733	0,9932	0,9832	147
C	0,6514	0,4830	0,5547	147
D	0,9784	0,9252	0,9510	147
E	0,9932	1,0000	0,9966	147
F	0,9032	0,9589	0,9302	146
G	1,0000	0,9864	0,9932	147
H	0,9664	0,9863	0,9763	146
I	0,6065	0,6395	0,6225	147
J	0,9242	0,8299	0,8746	147
K	0,7254	0,7007	0,7128	147
L	0,9796	0,9863	0,9829	146
M	0,7232	0,5510	0,6255	147
N	0,9866	1,0000	0,9932	147
O	0,5649	0,5034	0,5324	147
P	0,8137	0,5646	0,6667	147
Q	0,9930	0,9592	0,9758	147
R	1,0000	0,9863	0,9931	146
S	0,6769	0,6027	0,6377	146
T	0,9932	0,9932	0,9932	146
U	0,8067	0,6531	0,7218	147
V	0,5781	0,5034	0,5382	147
W	0,8000	0,6531	0,7191	147
X	0,7281	0,5646	0,6360	147
Y	0,7778	0,7192	0,7473	146
Z	0,6860	0,8027	0,7398	147
a	0,9720	0,9456	0,9586	147
b	0,9790	0,9524	0,9655	147
c	0,5904	0,7551	0,6627	147
d	0,9865	1,0000	0,9932	146
e	1,0000	0,9864	0,9932	147

f	0,9353	0,8844	0,9091	147
g	0,9730	0,9796	0,9763	147
h	0,9932	0,9932	0,9932	147
i	0,7121	0,6438	0,6763	146
j	0,8408	0,9041	0,8713	146
k	0,7067	0,7260	0,7162	146
l	0,6134	0,4966	0,5489	147
m	0,6461	0,7823	0,7077	147
n	1,0000	0,9932	0,9966	147
o	0,4519	0,6395	0,5296	147
p	0,6582	0,8776	0,7522	147
q	0,7615	0,6735	0,7148	147
r	0,9931	0,9728	0,9828	147
s	0,6645	0,7007	0,6821	147
t	0,9795	0,9795	0,9795	146
u	0,7022	0,8562	0,7716	146
v	0,5698	0,6667	0,6144	147
w	0,7118	0,8231	0,7634	147
x	0,6441	0,7755	0,7037	147
y	0,7278	0,7877	0,7566	146
z	0,7674	0,6735	0,7174	147
Weighted Average	0,8236	0,8187	0,8179	9096

L.28 Hasil Uji Coba Parameter *Optimizer* Adam pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,5352	0,7755	0,6333	147
1	0,5556	0,7823	0,6497	147
2	0,9839	0,8299	0,9004	147
3	1,0000	0,9864	0,9932	147
4	0,9660	0,9660	0,9660	147
5	0,9650	0,9452	0,9550	146

6	0,9714	0,9252	0,9477	147
7	0,9930	0,9658	0,9792	146
8	1,0000	0,9932	0,9966	147
9	0,6989	0,8904	0,7831	146
A	0,9865	1,0000	0,9932	146
B	1,0000	0,9932	0,9966	147
C	0,6463	0,7211	0,6817	147
D	0,9286	0,9728	0,9502	147
E	0,9735	1,0000	0,9866	147
F	0,9262	0,9452	0,9356	146
G	1,0000	0,9660	0,9827	147
H	0,9669	1,0000	0,9832	146
I	0,5536	0,6327	0,5905	147
J	0,9291	0,8027	0,8613	147
K	0,6928	0,7823	0,7348	147
L	0,9664	0,9863	0,9763	146
M	0,7143	0,6463	0,6786	147
N	0,9800	1,0000	0,9899	147
O	0,6640	0,5646	0,6103	147
P	0,9223	0,6463	0,7600	147
Q	1,0000	0,9456	0,9720	147
R	0,8848	1,0000	0,9389	146
S	0,6689	0,6781	0,6735	146
T	0,9932	0,9932	0,9932	146
U	0,9111	0,5578	0,6920	147
V	0,7368	0,3810	0,5022	147
W	0,9559	0,4422	0,6047	147
X	0,7559	0,6531	0,7007	147
Y	0,8250	0,6781	0,7444	146
Z	0,6471	0,8231	0,7246	147
a	0,9726	0,9660	0,9693	147
b	0,9416	0,9864	0,9635	147
c	0,7008	0,6054	0,6496	147
d	0,9932	0,9932	0,9932	146

e	0,9932	0,9932	0,9932	147
f	0,9122	0,9184	0,9153	147
g	0,9718	0,9388	0,9550	147
h	0,9932	0,9864	0,9898	147
i	0,9610	0,5068	0,6637	146
j	0,8313	0,9110	0,8693	146
k	0,7563	0,6164	0,6792	146
l	0,5714	0,5170	0,5429	147
m	0,6813	0,7415	0,7101	147
n	1,0000	0,9864	0,9932	147
o	0,6000	0,4082	0,4858	147
p	0,7366	0,9320	0,8228	147
q	0,8174	0,6395	0,7176	147
r	0,9735	1,0000	0,9866	147
s	0,6689	0,6871	0,6779	147
t	0,9860	0,9658	0,9758	146
u	0,6462	0,9384	0,7654	146
v	0,5926	0,8707	0,7052	147
w	0,6368	0,9660	0,7676	147
x	0,6845	0,7823	0,7302	147
y	0,7193	0,8425	0,7760	146
z	0,7280	0,6190	0,6691	147
Weighted Average	0,8381	0,8255	0,8229	9096

L.29 Hasil Uji Coba Parameter *Optimizer* RMSProp pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,5575	0,4286	0,4846	147
1	0,6753	0,7075	0,6910	147
2	0,9638	0,9048	0,9333	147
3	0,9932	1,0000	0,9966	147
4	0,8938	0,9728	0,9316	147

5	0,9504	0,9178	0,9338	146
6	0,9396	0,9524	0,9459	147
7	0,9930	0,9726	0,9827	146
8	1,0000	0,9660	0,9827	147
9	0,7362	0,8219	0,7767	146
A	0,9862	0,9795	0,9828	146
B	1,0000	0,9864	0,9932	147
C	0,6838	0,6327	0,6572	147
D	0,9658	0,9592	0,9625	147
E	0,9932	1,0000	0,9966	147
F	0,8712	0,9726	0,9191	146
G	0,9931	0,9728	0,9828	147
H	0,9860	0,9658	0,9758	146
I	0,5647	0,6531	0,6057	147
J	0,8759	0,8639	0,8699	147
K	0,6784	0,7891	0,7296	147
L	0,9355	0,9932	0,9635	146
M	0,7063	0,6054	0,6520	147
N	0,9930	0,9660	0,9793	147
O	0,6139	0,4218	0,5000	147
P	0,8091	0,6054	0,6926	147
Q	0,9722	0,9524	0,9622	147
R	0,9931	0,9795	0,9862	146
S	0,6063	0,6644	0,6340	146
T	0,9862	0,9795	0,9828	146
U	0,7810	0,7279	0,7535	147
V	0,6103	0,5646	0,5866	147
W	0,6537	0,9116	0,7614	147
X	0,6815	0,6259	0,6525	147
Y	0,7425	0,8493	0,7923	146
Z	0,6772	0,5850	0,6277	147
a	0,9161	0,9660	0,9404	147
b	0,9664	0,9796	0,9730	147
c	0,6732	0,7007	0,6867	147

d	0,9799	1,0000	0,9898	146
e	1,0000	0,9728	0,9862	147
f	0,9474	0,8571	0,9000	147
g	0,9793	0,9660	0,9726	147
h	0,9931	0,9796	0,9863	147
i	0,7222	0,6233	0,6691	146
j	0,8671	0,8493	0,8581	146
k	0,7333	0,6027	0,6617	146
l	0,5620	0,5238	0,5423	147
m	0,6587	0,7483	0,7006	147
n	0,9932	0,9932	0,9932	147
o	0,4375	0,6667	0,5283	147
p	0,6702	0,8571	0,7522	147
q	0,7727	0,6939	0,7312	147
r	0,9796	0,9796	0,9796	147
s	0,6503	0,6327	0,6414	147
t	0,9470	0,9795	0,9630	146
u	0,7170	0,7808	0,7475	146
v	0,6159	0,6327	0,6242	147
w	0,8247	0,5442	0,6557	147
x	0,6604	0,7143	0,6863	147
y	0,8348	0,6575	0,7356	146
z	0,6307	0,7551	0,6873	147
Weighted Average	0,8192	0,8145	0,8137	9096

L.30 Hasil Uji Coba Parameter *Learning Rate* 0,01 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,5352	0,7755	0,6333	147
1	0,5556	0,7823	0,6497	147
2	0,9839	0,8299	0,9004	147
3	1,0000	0,9864	0,9932	147

4	0,9660	0,9660	0,9660	147
5	0,9650	0,9452	0,9550	146
6	0,9714	0,9252	0,9477	147
7	0,9930	0,9658	0,9792	146
8	1,0000	0,9932	0,9966	147
9	0,6989	0,8904	0,7831	146
A	0,9865	1,0000	0,9932	146
B	1,0000	0,9932	0,9966	147
C	0,6463	0,7211	0,6817	147
D	0,9286	0,9728	0,9502	147
E	0,9735	1,0000	0,9866	147
F	0,9262	0,9452	0,9356	146
G	1,0000	0,9660	0,9827	147
H	0,9669	1,0000	0,9832	146
I	0,5536	0,6327	0,5905	147
J	0,9291	0,8027	0,8613	147
K	0,6928	0,7823	0,7348	147
L	0,9664	0,9863	0,9763	146
M	0,7143	0,6463	0,6786	147
N	0,9800	1,0000	0,9899	147
O	0,6640	0,5646	0,6103	147
P	0,9223	0,6463	0,7600	147
Q	1,0000	0,9456	0,9720	147
R	0,8848	1,0000	0,9389	146
S	0,6689	0,6781	0,6735	146
T	0,9932	0,9932	0,9932	146
U	0,9111	0,5578	0,6920	147
V	0,7368	0,3810	0,5022	147
W	0,9559	0,4422	0,6047	147
X	0,7559	0,6531	0,7007	147
Y	0,8250	0,6781	0,7444	146
Z	0,6471	0,8231	0,7246	147
a	0,9726	0,9660	0,9693	147
b	0,9416	0,9864	0,9635	147

c	0,7008	0,6054	0,6496	147
d	0,9932	0,9932	0,9932	146
e	0,9932	0,9932	0,9932	147
f	0,9122	0,9184	0,9153	147
g	0,9718	0,9388	0,9550	147
h	0,9932	0,9864	0,9898	147
i	0,9610	0,5068	0,6637	146
j	0,8313	0,9110	0,8693	146
k	0,7563	0,6164	0,6792	146
l	0,5714	0,5170	0,5429	147
m	0,6813	0,7415	0,7101	147
n	1,0000	0,9864	0,9932	147
o	0,6000	0,4082	0,4858	147
p	0,7366	0,9320	0,8228	147
q	0,8174	0,6395	0,7176	147
r	0,9735	1,0000	0,9866	147
s	0,6689	0,6871	0,6779	147
t	0,9860	0,9658	0,9758	146
u	0,6462	0,9384	0,7654	146
v	0,5926	0,8707	0,7052	147
w	0,6368	0,9660	0,7676	147
x	0,6845	0,7823	0,7302	147
y	0,7193	0,8425	0,7760	146
z	0,7280	0,6190	0,6691	147
Weighted Average	0,8381	0,8255	0,8229	9096

L.31 Hasil Uji Coba Parameter *Learning Rate* 0,001 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,6154	0,5442	0,5776	147
1	0,5451	0,8639	0,6684	147
2	0,9559	0,8844	0,9187	147

3	1,0000	1,0000	1,0000	147
4	0,9653	0,9456	0,9553	147
5	0,9716	0,9384	0,9547	146
6	0,9517	0,9388	0,9452	147
7	0,9863	0,9863	0,9863	146
8	0,9862	0,9728	0,9795	147
9	0,8201	0,7808	0,8000	146
A	0,9932	0,9932	0,9932	146
B	0,9932	1,0000	0,9966	147
C	0,6485	0,7279	0,6859	147
D	0,9600	0,9796	0,9697	147
E	0,9865	0,9932	0,9898	147
F	0,9161	0,9726	0,9435	146
G	0,9667	0,9864	0,9764	147
H	0,9797	0,9932	0,9864	146
I	0,6291	0,6463	0,6376	147
J	0,8171	0,9116	0,8617	147
K	0,7358	0,7959	0,7647	147
L	0,9793	0,9726	0,9759	146
M	0,6627	0,7483	0,7029	147
N	0,9866	1,0000	0,9932	147
O	0,6504	0,5442	0,5926	147
P	0,8200	0,5578	0,6640	147
Q	0,9733	0,9932	0,9832	147
R	1,0000	1,0000	1,0000	146
S	0,6713	0,6575	0,6644	146
T	1,0000	0,9932	0,9966	146
U	0,6964	0,7959	0,7429	147
V	0,6104	0,6395	0,6246	147
W	0,6597	0,8571	0,7456	147
X	0,6933	0,7687	0,7290	147
Y	0,7610	0,8288	0,7934	146
Z	0,6528	0,8571	0,7412	147
a	0,9730	0,9796	0,9763	147

b	0,9795	0,9728	0,9761	147
c	0,6977	0,6122	0,6522	147
d	0,9732	0,9932	0,9831	146
e	0,9931	0,9796	0,9863	147
f	0,9562	0,8912	0,9225	147
g	0,9664	0,9796	0,9730	147
h	1,0000	0,9864	0,9932	147
i	0,8925	0,5685	0,6946	146
j	0,8779	0,7877	0,8303	146
k	0,7863	0,7055	0,7437	146
l	0,5909	0,4422	0,5058	147
m	0,7132	0,6259	0,6667	147
n	1,0000	0,9932	0,9966	147
o	0,5053	0,6463	0,5672	147
p	0,6650	0,8912	0,7616	147
q	0,7770	0,7823	0,7797	147
r	1,0000	0,9660	0,9827	147
s	0,6689	0,6871	0,6779	147
t	0,9730	0,9863	0,9796	146
u	0,7797	0,6301	0,6970	146
v	0,6207	0,6122	0,6164	147
w	0,7961	0,5578	0,6560	147
x	0,7348	0,6599	0,6953	147
y	0,7887	0,7671	0,7778	146
z	0,8165	0,6054	0,6953	147
Weighted Average	0,8348	0,8286	0,8278	9096

L.32 Hasil Uji Coba Parameter *Learning Rate* 0,0001 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,4861	0,8299	0,6131	147
1	0,6303	0,7075	0,6667	147

2	0,9496	0,8980	0,9231	147
3	0,9932	1,0000	0,9966	147
4	0,9536	0,9796	0,9664	147
5	0,9122	0,9247	0,9184	146
6	0,9648	0,9320	0,9481	147
7	0,9928	0,9452	0,9684	146
8	0,9608	1,0000	0,9800	147
9	0,7881	0,8151	0,8013	146
A	0,9732	0,9932	0,9831	146
B	1,0000	0,9864	0,9932	147
C	0,6894	0,7551	0,7208	147
D	0,9857	0,9388	0,9617	147
E	0,9932	0,9932	0,9932	147
F	0,9122	0,9247	0,9184	146
G	0,9733	0,9932	0,9832	147
H	0,9792	0,9658	0,9724	146
I	0,5833	0,7143	0,6422	147
J	0,8366	0,8707	0,8533	147
K	0,6806	0,8844	0,7692	147
L	0,9662	0,9795	0,9728	146
M	0,6395	0,7483	0,6897	147
N	1,0000	1,0000	1,0000	147
O	0,5686	0,5918	0,5800	147
P	0,7857	0,7483	0,7666	147
Q	0,9862	0,9728	0,9795	147
R	1,0000	0,9863	0,9931	146
S	0,7941	0,3699	0,5047	146
T	0,9799	1,0000	0,9898	146
U	0,8047	0,7007	0,7491	147
V	0,6412	0,5714	0,6043	147
W	0,8151	0,6599	0,7293	147
X	0,7451	0,5170	0,6104	147
Y	0,7500	0,6575	0,7007	146
Z	0,7500	0,6327	0,6863	147

a	0,9720	0,9456	0,9586	147
b	0,9536	0,9796	0,9664	147
c	0,7388	0,6735	0,7046	147
d	0,9865	1,0000	0,9932	146
e	1,0000	1,0000	1,0000	147
f	0,9110	0,9048	0,9078	147
g	0,9404	0,9660	0,9530	147
h	1,0000	0,9728	0,9862	147
i	0,7788	0,6027	0,6795	146
j	0,8750	0,8151	0,8440	146
k	0,8365	0,5959	0,6960	146
l	0,5748	0,4966	0,5328	147
m	0,7155	0,5646	0,6312	147
n	0,9932	0,9932	0,9932	147
o	0,6042	0,1973	0,2974	147
p	0,7628	0,8095	0,7855	147
q	0,7708	0,7551	0,7629	147
r	0,9932	0,9932	0,9932	147
s	0,5991	0,9048	0,7209	147
t	0,9861	0,9726	0,9793	146
u	0,6886	0,7877	0,7348	146
v	0,5949	0,6395	0,6164	147
w	0,7033	0,8707	0,7781	147
x	0,6263	0,8095	0,7062	147
y	0,7125	0,7808	0,7451	146
z	0,6592	0,8027	0,7239	147
Weighted Average	82,96	82,29	81,96	9096

L.33 Hasil Uji Coba Parameter *Dropout* 0,3 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,5660	0,6122	0,5882	147

1	0,5333	0,8163	0,6452	147
2	0,8650	0,9592	0,9097	147
3	0,9932	0,9932	0,9932	147
4	0,9658	0,9592	0,9625	147
5	0,9774	0,8904	0,9319	146
6	0,9338	0,9592	0,9463	147
7	0,9863	0,9863	0,9863	146
8	0,9797	0,9864	0,9831	147
9	0,7875	0,8630	0,8235	146
A	0,9929	0,9521	0,9720	146
B	1,0000	0,9728	0,9862	147
C	0,6184	0,8707	0,7232	147
D	0,9716	0,9320	0,9514	147
E	0,9865	0,9932	0,9898	147
F	0,9155	0,8904	0,9028	146
G	0,9864	0,9864	0,9864	147
H	0,9864	0,9932	0,9898	146
I	0,6000	0,7143	0,6522	147
J	0,9732	0,7415	0,8417	147
K	0,7205	0,7891	0,7532	147
L	0,9861	0,9726	0,9793	146
M	0,6623	0,6803	0,6711	147
N	1,0000	1,0000	1,0000	147
O	0,6015	0,5442	0,5714	147
P	0,7356	0,8707	0,7975	147
Q	0,9930	0,9660	0,9793	147
R	1,0000	0,9863	0,9931	146
S	0,7073	0,5959	0,6468	146
T	0,9733	1,0000	0,9865	146
U	0,9080	0,5374	0,6752	147
V	0,7857	0,2993	0,4335	147
W	0,8015	0,7415	0,7703	147
X	0,8053	0,6190	0,7000	147
Y	0,8661	0,6644	0,7519	146

Z	0,6600	0,8980	0,7608	147
a	0,9510	0,9252	0,9379	147
b	0,9792	0,9592	0,9691	147
c	0,7849	0,4966	0,6083	147
d	0,9932	1,0000	0,9966	146
e	1,0000	0,9932	0,9966	147
f	0,8993	0,9116	0,9054	147
g	0,9858	0,9456	0,9653	147
h	0,9932	0,9864	0,9898	147
i	0,8019	0,5822	0,6746	146
j	0,7989	0,9521	0,8688	146
k	0,7537	0,6918	0,7214	146
l	0,7024	0,4014	0,5108	147
m	0,6761	0,6531	0,6644	147
n	0,9799	0,9932	0,9865	147
o	0,5096	0,5442	0,5263	147
p	0,8403	0,6803	0,7519	147
q	0,8112	0,7891	0,8000	147
r	0,9605	0,9932	0,9766	147
s	0,6522	0,8163	0,7251	147
t	0,9931	0,9795	0,9862	146
u	0,6667	0,9041	0,7674	146
v	0,5578	0,9524	0,7035	147
w	0,7580	0,8095	0,7829	147
x	0,6793	0,8503	0,7553	147
y	0,7167	0,8836	0,7914	146
z	0,8667	0,4422	0,5856	147
Weighted Average	0,8408	0,8285	0,8254	9096

L.34 Hasil Uji Coba Parameter *Dropout* 0,5 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,6154	0,5442	0,5776	147
1	0,5451	0,8639	0,6684	147
2	0,9559	0,8844	0,9187	147
3	1,0000	1,0000	1,0000	147
4	0,9653	0,9456	0,9553	147
5	0,9716	0,9384	0,9547	146
6	0,9517	0,9388	0,9452	147
7	0,9863	0,9863	0,9863	146
8	0,9862	0,9728	0,9795	147
9	0,8201	0,7808	0,8000	146
A	0,9932	0,9932	0,9932	146
B	0,9932	1,0000	0,9966	147
C	0,6485	0,7279	0,6859	147
D	0,9600	0,9796	0,9697	147
E	0,9865	0,9932	0,9898	147
F	0,9161	0,9726	0,9435	146
G	0,9667	0,9864	0,9764	147
H	0,9797	0,9932	0,9864	146
I	0,6291	0,6463	0,6376	147
J	0,8171	0,9116	0,8617	147
K	0,7358	0,7959	0,7647	147
L	0,9793	0,9726	0,9759	146
M	0,6627	0,7483	0,7029	147
N	0,9866	1,0000	0,9932	147
O	0,6504	0,5442	0,5926	147
P	0,8200	0,5578	0,6640	147
Q	0,9733	0,9932	0,9832	147
R	1,0000	1,0000	1,0000	146
S	0,6713	0,6575	0,6644	146
T	1,0000	0,9932	0,9966	146
U	0,6964	0,7959	0,7429	147

V	0,6104	0,6395	0,6246	147
W	0,6597	0,8571	0,7456	147
X	0,6933	0,7687	0,7290	147
Y	0,7610	0,8288	0,7934	146
Z	0,6528	0,8571	0,7412	147
a	0,9730	0,9796	0,9763	147
b	0,9795	0,9728	0,9761	147
c	0,6977	0,6122	0,6522	147
d	0,9732	0,9932	0,9831	146
e	0,9931	0,9796	0,9863	147
f	0,9562	0,8912	0,9225	147
g	0,9664	0,9796	0,9730	147
h	1,0000	0,9864	0,9932	147
i	0,8925	0,5685	0,6946	146
j	0,8779	0,7877	0,8303	146
k	0,7863	0,7055	0,7437	146
l	0,5909	0,4422	0,5058	147
m	0,7132	0,6259	0,6667	147
n	1,0000	0,9932	0,9966	147
o	0,5053	0,6463	0,5672	147
p	0,6650	0,8912	0,7616	147
q	0,7770	0,7823	0,7797	147
r	1,0000	0,9660	0,9827	147
s	0,6689	0,6871	0,6779	147
t	0,9730	0,9863	0,9796	146
u	0,7797	0,6301	0,6970	146
v	0,6207	0,6122	0,6164	147
w	0,7961	0,5578	0,6560	147
x	0,7348	0,6599	0,6953	147
y	0,7887	0,7671	0,7778	146
z	0,8165	0,6054	0,6953	147
Weighted Average	0,8348	0,8286	0,8278	9096

L.35 Hasil Uji Coba Parameter *Dropout* 0,7 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,6216	0,6259	0,6237	147
1	0,5955	0,7211	0,6523	147
2	0,9706	0,8980	0,9329	147
3	1,0000	1,0000	1,0000	147
4	0,9597	0,9728	0,9662	147
5	0,9568	0,9110	0,9333	146
6	0,9333	0,9524	0,9428	147
7	0,9930	0,9726	0,9827	146
8	1,0000	0,9864	0,9932	147
9	0,7263	0,8904	0,8000	146
A	0,9932	0,9932	0,9932	146
B	1,0000	1,0000	1,0000	147
C	0,6328	0,7619	0,6914	147
D	0,9726	0,9660	0,9693	147
E	0,9866	1,0000	0,9932	147
F	0,9167	0,9795	0,9470	146
G	0,9732	0,9864	0,9797	147
H	0,9931	0,9863	0,9897	146
I	0,5491	0,6463	0,5938	147
J	0,9030	0,8231	0,8612	147
K	0,6983	0,8503	0,7669	147
L	0,9536	0,9863	0,9697	146
M	0,6776	0,7007	0,6890	147
N	0,9932	0,9864	0,9898	147
O	0,6486	0,4898	0,5581	147
P	0,7455	0,8367	0,7885	147
Q	0,9862	0,9728	0,9795	147
R	0,9932	0,9932	0,9932	146
S	0,5829	0,7466	0,6547	146
T	0,9932	1,0000	0,9966	146
U	0,7956	0,7415	0,7676	147

V	0,6034	0,7143	0,6542	147
W	0,6821	0,9048	0,7778	147
X	0,6522	0,8163	0,7251	147
Y	0,7262	0,8356	0,7771	146
Z	0,7273	0,7619	0,7442	147
a	0,9728	0,9728	0,9728	147
b	0,9857	0,9388	0,9617	147
c	0,7217	0,5646	0,6336	147
d	0,9865	1,0000	0,9932	146
e	0,9932	0,9932	0,9932	147
f	0,9776	0,8912	0,9324	147
g	0,9859	0,9524	0,9689	147
h	0,9932	0,9864	0,9898	147
i	0,7521	0,6027	0,6692	146
j	0,8487	0,8836	0,8658	146
k	0,7931	0,6301	0,7023	146
l	0,5868	0,4830	0,5299	147
m	0,6972	0,6735	0,6851	147
n	1,0000	0,9932	0,9966	147
o	0,5054	0,6327	0,5619	147
p	0,8092	0,7211	0,7626	147
q	0,8291	0,6599	0,7348	147
r	0,9861	0,9660	0,9759	147
s	0,6694	0,5510	0,6045	147
t	0,9864	0,9932	0,9898	146
u	0,7532	0,7945	0,7733	146
v	0,6557	0,5442	0,5948	147
w	0,8627	0,5986	0,7068	147
x	0,7545	0,5646	0,6459	147
y	0,8110	0,7055	0,7546	146
z	0,7279	0,7279	0,7279	147
Weighted Average	0,8351	0,8295	0,8290	9096

L.36 Hasil Uji Coba Parameter *Dropout* 0,9 pada Prediksi Karakter FCN

Kelas	Precision	Recall	F1-Score	Support
0	0,6512	0,3810	0,4807	147
1	0,5648	0,8299	0,6722	147
2	0,9396	0,9524	0,9459	147
3	0,9932	1,0000	0,9966	147
4	0,9792	0,9592	0,9691	147
5	0,9384	0,9384	0,9384	146
6	0,9855	0,9252	0,9544	147
7	0,9931	0,9795	0,9862	146
8	0,9866	1,0000	0,9932	147
9	0,8015	0,7192	0,7581	146
A	1,0000	0,9726	0,9861	146
B	0,9931	0,9796	0,9863	147
C	0,6792	0,7347	0,7059	147
D	0,9862	0,9728	0,9795	147
E	1,0000	1,0000	1,0000	147
F	0,9045	0,9726	0,9373	146
G	0,9865	0,9932	0,9898	147
H	0,9733	1,0000	0,9865	146
I	0,5663	0,7551	0,6472	147
J	0,8442	0,8844	0,8638	147
K	0,6722	0,8231	0,7401	147
L	0,9795	0,9795	0,9795	146
M	0,6355	0,8776	0,7371	147
N	1,0000	0,9864	0,9932	147
O	0,6774	0,4286	0,5250	147
P	0,8468	0,7143	0,7749	147
Q	1,0000	0,9796	0,9897	147
R	0,9865	1,0000	0,9932	146
S	0,6556	0,8082	0,7239	146
T	0,9860	0,9658	0,9758	146
U	0,8718	0,6939	0,7727	147

V	0,6780	0,5442	0,6038	147
W	0,8306	0,7007	0,7601	147
X	0,7961	0,5578	0,6560	147
Y	0,8014	0,8014	0,8014	146
Z	0,7292	0,7143	0,7216	147
a	0,9726	0,9660	0,9693	147
b	0,9542	0,9932	0,9733	147
c	0,7206	0,6667	0,6926	147
d	1,0000	1,0000	1,0000	146
e	0,9932	1,0000	0,9966	147
f	0,9489	0,8844	0,9155	147
g	0,9793	0,9660	0,9726	147
h	0,9865	0,9932	0,9898	147
i	0,8673	0,5822	0,6967	146
j	0,8741	0,8082	0,8399	146
k	0,7798	0,5822	0,6667	146
l	0,6000	0,3265	0,4229	147
m	0,8111	0,4966	0,6160	147
n	1,0000	1,0000	1,0000	147
o	0,4415	0,7959	0,5680	147
p	0,7558	0,8844	0,8150	147
q	0,7246	0,8231	0,7707	147
r	0,9932	0,9864	0,9898	147
s	0,7679	0,5850	0,6641	147
t	0,9231	0,9863	0,9536	146
u	0,7543	0,9041	0,8224	146
v	0,6333	0,7755	0,6972	147
w	0,7440	0,8503	0,7937	147
x	0,6510	0,8503	0,7375	147
y	0,8095	0,8151	0,8123	146
z	0,7466	0,7415	0,7440	147
Weighted Average	0,8442	0,8352	0,8329	9096

L.37 Hasil Uji Coba Penggunaan Algoritma *Sliding Windows* Sebesar 32x16 pada Sistem Keseluruhan

Hasil Sliding Windows	Precision	Recall	F1-Score	Support
	0,0000	0,0000	0,0000	0
0Ur	0,0000	0,0000	0,0000	0
0n	0,0000	0,0000	0,0000	0
0r	0,0000	0,0000	0,0000	0
1	0,0000	0,0000	0,0000	0
11IijiS	0,0000	0,0000	0,0000	0
3UUPYPEJRSS	0,0000	0,0000	0,0000	0
4CK	0,0000	0,0000	0,0000	0
4U	0,0000	0,0000	0,0000	0
AlUl	0,0000	0,0000	0,0000	0
All	0,0000	0,0000	0,0000	1
BAG	1,0000	1,0000	1,0000	1
Bag	0,0000	0,0000	0,0000	1
CCCt	0,0000	0,0000	0,0000	0
CJfRSNid	0,0000	0,0000	0,0000	0
CS	0,0000	0,0000	0,0000	0
CSDVn	0,0000	0,0000	0,0000	0
CSSOr	0,0000	0,0000	0,0000	0
CYf	0,0000	0,0000	0,0000	0
CfCSrr	0,0000	0,0000	0,0000	0
Cjltrd	0,0000	0,0000	0,0000	0
Cy	0,0000	0,0000	0,0000	0
Etg	0,0000	0,0000	0,0000	0
FCCSCCC	0,0000	0,0000	0,0000	0
FIRS	0,0000	0,0000	0,0000	0
FROM	0,0000	0,0000	0,0000	1
FYCjtKEJD	0,0000	0,0000	0,0000	0
From	0,0000	0,0000	0,0000	1
GDUMCJkLY	0,0000	0,0000	0,0000	0
GJlLj	0,0000	0,0000	0,0000	0

HONw	0,0000	0,0000	0,0000	0
How	0,0000	0,0000	0,0000	1
JUTE	0,0000	0,0000	0,0000	1
JVMNP	0,0000	0,0000	0,0000	0
Jhi	0,0000	0,0000	0,0000	0
Jump	0,0000	0,0000	0,0000	3
Jute	0,0000	0,0000	0,0000	1
JyCsp	0,0000	0,0000	0,0000	0
LIi	0,0000	0,0000	0,0000	0
LhOj	0,0000	0,0000	0,0000	0
Lhj	0,0000	0,0000	0,0000	0
MMMMR	0,0000	0,0000	0,0000	0
MMR	0,0000	0,0000	0,0000	0
MMe	0,0000	0,0000	0,0000	0
MUe	0,0000	0,0000	0,0000	0
Md	0,0000	0,0000	0,0000	0
My	0,0000	0,0000	0,0000	0
N	0,0000	0,0000	0,0000	0
NCj	0,0000	0,0000	0,0000	0
NkhhNd	0,0000	0,0000	0,0000	0
Nn	0,0000	0,0000	0,0000	0
NrUCCe	0,0000	0,0000	0,0000	0
Nrn	0,0000	0,0000	0,0000	0
Nt	0,0000	0,0000	0,0000	0
Ntt	0,0000	0,0000	0,0000	0
OIIiytrlrCCj	0,0000	0,0000	0,0000	0
ON1	0,0000	0,0000	0,0000	0
ONU1	0,0000	0,0000	0,0000	0
OOrA	0,0000	0,0000	0,0000	0
OUr	0,0000	0,0000	0,0000	0
OVeUr	0,0000	0,0000	0,0000	0
OiIrId	0,0000	0,0000	0,0000	0
On	0,0000	0,0000	0,0000	0
One	0,0000	0,0000	0,0000	0

Or	0,0000	0,0000	0,0000	0
Oxford	0,0000	0,0000	0,0000	6
P3ejrSjrNUt	0,0000	0,0000	0,0000	0
P7jtIId	0,0000	0,0000	0,0000	0
PCejrIYtj	0,0000	0,0000	0,0000	0
PICKED	0,0000	0,0000	0,0000	1
Picked	0,0000	0,0000	0,0000	1
PqyC	0,0000	0,0000	0,0000	0
PttSg	0,0000	0,0000	0,0000	0
QUICKLY	0,0000	0,0000	0,0000	1
Quick	0,0000	0,0000	0,0000	1
Quickly	0,0000	0,0000	0,0000	1
SCCSCCs4r	0,0000	0,0000	0,0000	0
SIXTY	0,0000	0,0000	0,0000	1
SJbN	0,0000	0,0000	0,0000	0
SNXCTTY	0,0000	0,0000	0,0000	0
SO	0,0000	0,0000	0,0000	0
SVhiIa	0,0000	0,0000	0,0000	0
ShN	0,0000	0,0000	0,0000	0
ShNe	0,0000	0,0000	0,0000	0
Sijnx	0,0000	0,0000	0,0000	0
Sixty	0,0000	0,0000	0,0000	1
SjOhNd	0,0000	0,0000	0,0000	0
Sjh	0,0000	0,0000	0,0000	0
Sjiid	0,0000	0,0000	0,0000	0
SlNrCCCCS	0,0000	0,0000	0,0000	0
SlxtLy	0,0000	0,0000	0,0000	0
Slxtj	0,0000	0,0000	0,0000	0
StJrrCg	0,0000	0,0000	0,0000	0
StSSregUtyh	0,0000	0,0000	0,0000	0
StrrIrCCgUh	0,0000	0,0000	0,0000	0
THE	1,0000	1,0000	1,0000	1
The	0,0000	0,0000	0,0000	2
UCJNW	0,0000	0,0000	0,0000	0

UJNh	0,0000	0,0000	0,0000	0
UNN	0,0000	0,0000	0,0000	0
UNNhjh	0,0000	0,0000	0,0000	0
UNbl7rCCy	0,0000	0,0000	0,0000	0
UNkNwji	0,0000	0,0000	0,0000	0
UNrjNrNjHtSINry	0,0000	0,0000	0,0000	0
UR	0,0000	0,0000	0,0000	0
UTEE	0,0000	0,0000	0,0000	0
UUR	0,0000	0,0000	0,0000	0
UURUR	0,0000	0,0000	0,0000	0
UUaK	0,0000	0,0000	0,0000	0
UjkjUUd	0,0000	0,0000	0,0000	0
UkjrCe	0,0000	0,0000	0,0000	0
VNNOhVhEIN	0,0000	0,0000	0,0000	0
VkjNW	0,0000	0,0000	0,0000	0
WEJPE	0,0000	0,0000	0,0000	0
WERE	0,0000	0,0000	0,0000	1
WMW	0,0000	0,0000	0,0000	0
WO	0,0000	0,0000	0,0000	0
WOVEN	0,0000	0,0000	0,0000	1
WUe	0,0000	0,0000	0,0000	0
We	0,0000	0,0000	0,0000	1
Were	0,0000	0,0000	0,0000	1
WhM	0,0000	0,0000	0,0000	0
Whhjh	0,0000	0,0000	0,0000	0
Woven	0,0000	0,0000	0,0000	1
YU	0,0000	0,0000	0,0000	0
ZIPPERS	0,0000	0,0000	0,0000	1
Zippers	0,0000	0,0000	0,0000	1
a	1,0000	1,0000	1,0000	9
aHt	0,0000	0,0000	0,0000	0
aNhNd	0,0000	0,0000	0,0000	0
aNn	0,0000	0,0000	0,0000	0
aNre	0,0000	0,0000	0,0000	0

aNt	0,0000	0,0000	0,0000	0
ah11	0,0000	0,0000	0,0000	0
all	0,0000	0,0000	0,0000	6
an	1,0000	0,8889	0,9412	9
and	1,0000	0,7778	0,8750	9
are	1,0000	0,8889	0,9412	9
as	1,0000	0,8889	0,9412	9
at	1,0000	0,6667	0,8000	9
b	0,0000	0,0000	0,0000	0
bIaSiiNhNIiS	0,0000	0,0000	0,0000	0
bN	0,0000	0,0000	0,0000	0
bO	0,0000	0,0000	0,0000	0
ba	0,0000	0,0000	0,0000	0
bag	0,0000	0,0000	0,0000	1
be	1,0000	0,5556	0,7143	9
been	1,0000	0,6667	0,8000	9
beginners	0,0000	0,0000	0,0000	5
bhAt	0,0000	0,0000	0,0000	0
bhj	0,0000	0,0000	0,0000	0
bj	0,0000	0,0000	0,0000	0
bjajh	0,0000	0,0000	0,0000	0
but	1,0000	0,6667	0,8000	9
by	1,0000	0,5556	0,7143	9
f	0,0000	0,0000	0,0000	0
fjj	0,0000	0,0000	0,0000	0
for	1,0000	0,8889	0,9412	9
frogs	0,0000	0,0000	0,0000	1
from	0,9000	0,9000	0,9000	10
glSUjl	0,0000	0,0000	0,0000	0
goblin	0,0000	0,0000	0,0000	1
gymnasts	0,0000	0,0000	0,0000	1
h	0,0000	0,0000	0,0000	0
hF	0,0000	0,0000	0,0000	0
hFdR	0,0000	0,0000	0,0000	0

hG	0,0000	0,0000	0,0000	0
hNOHt	0,0000	0,0000	0,0000	0
hNOS	0,0000	0,0000	0,0000	0
hNa	0,0000	0,0000	0,0000	0
hNaNS	0,0000	0,0000	0,0000	0
hNalS	0,0000	0,0000	0,0000	0
hNikS	0,0000	0,0000	0,0000	0
hNji	0,0000	0,0000	0,0000	0
hOt	0,0000	0,0000	0,0000	0
had	1,0000	1,0000	1,0000	9
hajjNa	0,0000	0,0000	0,0000	0
has	1,0000	0,3333	0,5000	9
have	1,0000	0,4444	0,6154	9
he	1,0000	0,8889	0,9412	9
her	1,0000	0,8889	0,9412	9
hi	0,0000	0,0000	0,0000	0
hiI	0,0000	0,0000	0,0000	0
hiS	0,0000	0,0000	0,0000	0
hii	0,0000	0,0000	0,0000	0
hiiS	0,0000	0,0000	0,0000	0
hijiI	0,0000	0,0000	0,0000	0
him	1,0000	1,0000	1,0000	9
his	1,0000	0,1111	0,2000	9
hjS	0,0000	0,0000	0,0000	0
i	1,0000	0,7778	0,8750	9
if	1,0000	0,5556	0,7143	9
ih	0,0000	0,0000	0,0000	0
iiS	0,0000	0,0000	0,0000	0
ijFCSCCS	0,0000	0,0000	0,0000	0
in	1,0000	0,4444	0,6154	9
is	1,0000	0,2222	0,3636	9
it	1,0000	0,3333	0,5000	9
iyE	0,0000	0,0000	0,0000	0
jISjhNhkOS	0,0000	0,0000	0,0000	0

jN1	0,0000	0,0000	0,0000	0
jNUrP	0,0000	0,0000	0,0000	0
jbUUtO	0,0000	0,0000	0,0000	0
jbri	0,0000	0,0000	0,0000	0
jhVlL	0,0000	0,0000	0,0000	0
jhjh	0,0000	0,0000	0,0000	0
jiNrrrIrI	0,0000	0,0000	0,0000	0
jihtJh	0,0000	0,0000	0,0000	0
jiiN1n	0,0000	0,0000	0,0000	0
jjNrd	0,0000	0,0000	0,0000	0
jllU	0,0000	0,0000	0,0000	0
jnNN	0,0000	0,0000	0,0000	0
jrESS	0,0000	0,0000	0,0000	0
jrrCP	0,0000	0,0000	0,0000	0
jryrCCPCS	0,0000	0,0000	0,0000	0
jtI4rlili	0,0000	0,0000	0,0000	0
jump	0,0000	0,0000	0,0000	3
jumping	0,0000	0,0000	0,0000	1
jumps	0,0000	0,0000	0,0000	1
jute	0,0000	0,0000	0,0000	1
lCCgS	0,0000	0,0000	0,0000	0
lg	0,0000	0,0000	0,0000	0
lord	1,0000	1,0000	1,0000	9
me	1,0000	0,5714	0,7273	7
more	1,0000	0,7778	0,8750	9
my	1,0000	0,6250	0,7692	8
nKO	0,0000	0,0000	0,0000	0
nO	0,0000	0,0000	0,0000	0
nOt	0,0000	0,0000	0,0000	0
nUO	0,0000	0,0000	0,0000	0
nUOUt	0,0000	0,0000	0,0000	0
nVNUen	0,0000	0,0000	0,0000	0
no	0,5000	0,1429	0,2222	7
not	1,0000	0,3000	0,4615	10

ns5	0,0000	0,0000	0,0000	0
oWnUe	0,0000	0,0000	0,0000	0
of	1,0000	0,7778	0,8750	9
on	1,0000	0,3333	0,5000	9
one	1,0000	0,6667	0,8000	9
or	1,0000	0,3333	0,5000	9
over	0,0000	0,0000	0,0000	1
personality	0,0000	0,0000	0,0000	3
picked	0,0000	0,0000	0,0000	1
piqued	0,0000	0,0000	0,0000	1
q7YYty	0,0000	0,0000	0,0000	0
qjNjk	0,0000	0,0000	0,0000	0
qjr	0,0000	0,0000	0,0000	0
qrVll	0,0000	0,0000	0,0000	0
qrlStEK	0,0000	0,0000	0,0000	0
qry1	0,0000	0,0000	0,0000	0
quick	0,0000	0,0000	0,0000	6
quickly	0,0000	0,0000	0,0000	1
r	0,0000	0,0000	0,0000	0
r7	0,0000	0,0000	0,0000	0
rC	0,0000	0,0000	0,0000	0
rCCPrrCj	0,0000	0,0000	0,0000	0
rCj	0,0000	0,0000	0,0000	0
rF	0,0000	0,0000	0,0000	0
rIISrrlOutbj	0,0000	0,0000	0,0000	0
rKJ	0,0000	0,0000	0,0000	0
rOt	0,0000	0,0000	0,0000	0
rS	0,0000	0,0000	0,0000	0
raNrjNrtbNajk	0,0000	0,0000	0,0000	0
razorback	0,0000	0,0000	0,0000	1
rff	0,0000	0,0000	0,0000	0
riiS	0,0000	0,0000	0,0000	0
riijSS	0,0000	0,0000	0,0000	0
rldt	0,0000	0,0000	0,0000	0

said	1,0000	0,7778	0,8750	9
shO	0,0000	0,0000	0,0000	0
shall	1,0000	1,0000	1,0000	9
she	1,0000	0,2222	0,3636	9
six	0,0000	0,0000	0,0000	1
sixty	0,0000	0,0000	0,0000	1
so	1,0000	0,8889	0,9412	9
strength	0,0000	0,0000	0,0000	5
t	0,0000	0,0000	0,0000	0
tJhii	0,0000	0,0000	0,0000	0
tJlhS	0,0000	0,0000	0,0000	0
tJliix	0,0000	0,0000	0,0000	0
tSrrrrSS	0,0000	0,0000	0,0000	0
tbU	0,0000	0,0000	0,0000	0
thN	0,0000	0,0000	0,0000	0
thNS	0,0000	0,0000	0,0000	0
thNSjj	0,0000	0,0000	0,0000	0
thNij	0,0000	0,0000	0,0000	0
thSij	0,0000	0,0000	0,0000	0
thSja	0,0000	0,0000	0,0000	0
tha	0,0000	0,0000	0,0000	0
that	1,0000	1,0000	1,0000	9
thb	0,0000	0,0000	0,0000	0
the	1,0000	0,7500	0,8571	8
their	1,0000	0,8889	0,9412	9
them	1,0000	0,7778	0,8750	9
there	0,8750	0,7778	0,8235	9
they	1,0000	0,8750	0,9333	8
thi	0,0000	0,0000	0,0000	0
this	1,0000	0,3333	0,5000	9
thou	1,0000	0,5556	0,7143	9
tiiil	0,0000	0,0000	0,0000	0
till	0,0000	0,0000	0,0000	1
tj	0,0000	0,0000	0,0000	0

tjhI	0,0000	0,0000	0,0000	0
tjhNjNr	0,0000	0,0000	0,0000	0
tjreOrgU	0,0000	0,0000	0,0000	0
tjrltrj	0,0000	0,0000	0,0000	0
tlhi	0,0000	0,0000	0,0000	0
tll1	0,0000	0,0000	0,0000	0
to	1,0000	0,7778	0,8750	9
tt	0,0000	0,0000	0,0000	0
uJR	0,0000	0,0000	0,0000	0
ubiquitary	0,0000	0,0000	0,0000	5
unto	1,0000	0,9000	0,9474	10
vow	0,0000	0,0000	0,0000	7
wNhNh	0,0000	0,0000	0,0000	0
was	1,0000	0,8000	0,8889	10
we	1,0000	0,1250	0,2222	8
were	1,0000	0,8182	0,9000	11
when	1,0000	0,1111	0,2000	9
which	1,0000	1,0000	1,0000	8
who	1,0000	0,8889	0,9412	9
will	1,0000	0,7500	0,8571	8
with	1,0000	0,8889	0,9412	9
would	1,0000	0,7778	0,8750	9
woven	0,0000	0,0000	0,0000	1
y3	0,0000	0,0000	0,0000	0
yS	0,0000	0,0000	0,0000	0
yS3y	0,0000	0,0000	0,0000	0
you	0,4444	0,4444	0,4444	9
yrSy	0,0000	0,0000	0,0000	0
zVPPssCS	0,0000	0,0000	0,0000	0
zippers	0,0000	0,0000	0,0000	1
Weighted Average	0,8364	0,5780	0,6552	609

L.38 Hasil Uji Coba Penggunaan Algoritma *Sliding Windows* Sebesar 32x16 pada Sistem Keseluruhan

Hasil Sliding Windows	Precision	Recall	F1-Score	Support
	0,0000	0,0000	0,0000	0
1	0,0000	0,0000	0,0000	0
11Iiji	0,0000	0,0000	0,0000	0
4tNNiS	0,0000	0,0000	0,0000	0
All	0,0000	0,0000	0,0000	1
BAG	0,0000	0,0000	0,0000	1
BBMG	0,0000	0,0000	0,0000	0
Bag	0,0000	0,0000	0,0000	1
C	0,0000	0,0000	0,0000	0
CC	0,0000	0,0000	0,0000	0
CCC	0,0000	0,0000	0,0000	0
CCCC	0,0000	0,0000	0,0000	0
CCCrICj	0,0000	0,0000	0,0000	0
CCJ	0,0000	0,0000	0,0000	0
CCSOO	0,0000	0,0000	0,0000	0
CCVN	0,0000	0,0000	0,0000	0
CJ	0,0000	0,0000	0,0000	0
CQCj	0,0000	0,0000	0,0000	0
Cq	0,0000	0,0000	0,0000	0
Csp	0,0000	0,0000	0,0000	0
DNSNNNN	0,0000	0,0000	0,0000	0
EPEE	0,0000	0,0000	0,0000	0
FCCc	0,0000	0,0000	0,0000	0
FF	0,0000	0,0000	0,0000	0
FFN	0,0000	0,0000	0,0000	0
FPCEJDD	0,0000	0,0000	0,0000	0
FROM	0,0000	0,0000	0,0000	1
From	0,0000	0,0000	0,0000	1
How	0,0000	0,0000	0,0000	1
INNl1	0,0000	0,0000	0,0000	0

IiiNNNNNIN	0,0000	0,0000	0,0000	0
IrCCC	0,0000	0,0000	0,0000	0
IrCCOCCji	0,0000	0,0000	0,0000	0
JUTE	0,0000	0,0000	0,0000	1
Jump	0,0000	0,0000	0,0000	3
Jute	0,0000	0,0000	0,0000	1
LCb	0,0000	0,0000	0,0000	0
LOrrrrr	0,0000	0,0000	0,0000	0
LWNj	0,0000	0,0000	0,0000	0
Lii	0,0000	0,0000	0,0000	0
Ljj	0,0000	0,0000	0,0000	0
LjwN	0,0000	0,0000	0,0000	0
M	0,0000	0,0000	0,0000	0
MM	0,0000	0,0000	0,0000	0
MMM	0,0000	0,0000	0,0000	0
MMMMM	0,0000	0,0000	0,0000	0
MMNN	0,0000	0,0000	0,0000	0
MNNN	0,0000	0,0000	0,0000	0
MrCCC	0,0000	0,0000	0,0000	0
N	0,0000	0,0000	0,0000	0
NCC	0,0000	0,0000	0,0000	0
NIrrNj	0,0000	0,0000	0,0000	0
NM	0,0000	0,0000	0,0000	0
NN	0,0000	0,0000	0,0000	0
NN1	0,0000	0,0000	0,0000	0
NNN	0,0000	0,0000	0,0000	0
NNN1	0,0000	0,0000	0,0000	0
NNNN	0,0000	0,0000	0,0000	0
NNNNNNN	0,0000	0,0000	0,0000	0
NNNNNNNN	0,0000	0,0000	0,0000	0
NNNNVNVEENN	0,0000	0,0000	0,0000	0
NNNNi	0,0000	0,0000	0,0000	0
NNNjNNN	0,0000	0,0000	0,0000	0
NNNt	0,0000	0,0000	0,0000	0

NNNtt	0,0000	0,0000	0,0000	0
NNO	0,0000	0,0000	0,0000	0
NNS	0,0000	0,0000	0,0000	0
NNUtt	0,0000	0,0000	0,0000	0
NNee	0,0000	0,0000	0,0000	0
NNhNi	0,0000	0,0000	0,0000	0
NNiS	0,0000	0,0000	0,0000	0
NNijS	0,0000	0,0000	0,0000	0
NNjjWU	0,0000	0,0000	0,0000	0
NNtd	0,0000	0,0000	0,0000	0
NNtt	0,0000	0,0000	0,0000	0
NO	0,0000	0,0000	0,0000	0
NOjjNNNNrrS	0,0000	0,0000	0,0000	0
NS	0,0000	0,0000	0,0000	0
NT	0,0000	0,0000	0,0000	0
NUNNNNNNNMOrry	0,0000	0,0000	0,0000	0
NVUVC	0,0000	0,0000	0,0000	0
NWWN	0,0000	0,0000	0,0000	0
NWWNN	0,0000	0,0000	0,0000	0
NWjN	0,0000	0,0000	0,0000	0
NaNrNIII	0,0000	0,0000	0,0000	0
Nai	0,0000	0,0000	0,0000	0
NhN	0,0000	0,0000	0,0000	0
NhNNNi	0,0000	0,0000	0,0000	0
NhNNj	0,0000	0,0000	0,0000	0
NhNiN	0,0000	0,0000	0,0000	0
Nhii	0,0000	0,0000	0,0000	0
NiNi	0,0000	0,0000	0,0000	0
NiS	0,0000	0,0000	0,0000	0
Nj	0,0000	0,0000	0,0000	0
NjCCCSS	0,0000	0,0000	0,0000	0
NjNrNMM	0,0000	0,0000	0,0000	0
Nl	0,0000	0,0000	0,0000	0
NrTT	0,0000	0,0000	0,0000	0

NriSS	0,0000	0,0000	0,0000	0
Ntt	0,0000	0,0000	0,0000	0
OO	0,0000	0,0000	0,0000	0
Oxford	0,0000	0,0000	0,0000	6
PICKED	0,0000	0,0000	0,0000	1
PPNIIaN	0,0000	0,0000	0,0000	0
PerrNSNNNNNCCS	0,0000	0,0000	0,0000	0
Picked	0,0000	0,0000	0,0000	1
QUICKLY	0,0000	0,0000	0,0000	1
Quick	0,0000	0,0000	0,0000	1
Quickly	0,0000	0,0000	0,0000	1
R	0,0000	0,0000	0,0000	0
S	0,0000	0,0000	0,0000	0
SCCCCCCrr	0,0000	0,0000	0,0000	0
SCOOOMO	0,0000	0,0000	0,0000	0
SIXTY	0,0000	0,0000	0,0000	1
SN	0,0000	0,0000	0,0000	0
SNNN	0,0000	0,0000	0,0000	0
SNWTTTM	0,0000	0,0000	0,0000	0
SNj	0,0000	0,0000	0,0000	0
SOS	0,0000	0,0000	0,0000	0
SS	0,0000	0,0000	0,0000	0
SSCCCgt	0,0000	0,0000	0,0000	0
SSNNNNN	0,0000	0,0000	0,0000	0
SSh	0,0000	0,0000	0,0000	0
SSjNd	0,0000	0,0000	0,0000	0
ShW	0,0000	0,0000	0,0000	0
Shh	0,0000	0,0000	0,0000	0
Sixty	0,0000	0,0000	0,0000	1
St	0,0000	0,0000	0,0000	0
StCCr	0,0000	0,0000	0,0000	0
T	0,0000	0,0000	0,0000	0
THE	0,0000	0,0000	0,0000	1
THEE	0,0000	0,0000	0,0000	0

The	0,0000	0,0000	0,0000	2
U	0,0000	0,0000	0,0000	0
UC	0,0000	0,0000	0,0000	0
UCN	0,0000	0,0000	0,0000	0
UM	0,0000	0,0000	0,0000	0
UNrrC	0,0000	0,0000	0,0000	0
UU	0,0000	0,0000	0,0000	0
UUC	0,0000	0,0000	0,0000	0
UUNNNN	0,0000	0,0000	0,0000	0
UUNNNNrrOC	0,0000	0,0000	0,0000	0
UUO	0,0000	0,0000	0,0000	0
UUUa	0,0000	0,0000	0,0000	0
UUVEE	0,0000	0,0000	0,0000	0
UUWNN	0,0000	0,0000	0,0000	0
UVN	0,0000	0,0000	0,0000	0
UW	0,0000	0,0000	0,0000	0
VNNNNNN	0,0000	0,0000	0,0000	0
VPscS	0,0000	0,0000	0,0000	0
WDN	0,0000	0,0000	0,0000	0
WERE	0,0000	0,0000	0,0000	1
WN	0,0000	0,0000	0,0000	0
WOS	0,0000	0,0000	0,0000	0
WOVEN	0,0000	0,0000	0,0000	1
WU	0,0000	0,0000	0,0000	0
WW	0,0000	0,0000	0,0000	0
WWN	0,0000	0,0000	0,0000	0
We	0,0000	0,0000	0,0000	1
Were	0,0000	0,0000	0,0000	1
Woven	0,0000	0,0000	0,0000	1
YN	0,0000	0,0000	0,0000	0
ZIPPERS	0,0000	0,0000	0,0000	1
Zippers	0,0000	0,0000	0,0000	1
Weighted Average	0,8278	0,5665	0,6437	609

L.39 *Confusion Matrix* Tiap Karakter dari Model Terbaik Prediksi Karakter FCN

	Hasil Prediksi			
	0	**D**	**O**	**o**
0	56	2	14	75

	Hasil Prediksi			
	1	**I**	**i**	**l**
1	122	11	4	10

	Hasil Prediksi				
	2	**Z**	**a**	**g**	**z**
2	140	3	1	1	2

	Hasil Prediksi
	3
3	147

	Hasil Prediksi				
	4	**H**	**Y**	**t**	**y**
4	141	1	2	1	2

	Hasil Prediksi					
	5	**J**	**S**	**j**	**s**	**t**
5	137	1	4	1	2	1

	Hasil Prediksi					
	0	**6**	**G**	**L**	**b**	**c**
6	1	136	2	1	6	1

	Hasil Prediksi		
	7	**T**	**Y**
7	143	2	1

	Hasil Prediksi
	8
8	147

	Hasil Prediksi		
	9	**g**	**q**
9	105	1	40

	Hasil Prediksi			
	A	**R**	**f**	**t**
A	142	2	1	1

	Hasil Prediksi		
	8	**B**	**e**
B	2	144	1

	Hasil Prediksi			
	C	**L**	**c**	**R**
C	108	1	37	1

	Hasil Prediksi				
	7	**B**	**D**	**O**	**O**
D	1	1	143	1	1

	Hasil Prediksi
	E
E	147

	Hasil Prediksi	
	F	f
F	142	4

	Hasil Prediksi	
	6	G
G	1	146

	Hasil Prediksi
	H
H	146

	Hasil Prediksi			
	1	I	i	l
I	16	111	5	15

	Hasil Prediksi			
	J	S	V	J
J	130	1	1	15

	Hasil Prediksi		
	K	k	x
K	121	24	2

	Hasil Prediksi		
	C	**L**	**l**
L	2	143	1

	Hasil Prediksi		
	H	**M**	**m**
M	1	123	17

	Hasil Prediksi		
	H	**N**	**v**
N	1	145	1

	Hasil Prediksi		
	0	**O**	**o**
O	12	63	72

	Hasil Prediksi	
	P	**p**
P	105	42

	Hasil Prediksi		
	P	**Q**	**a**
Q	1	144	2

	Hasil Prediksi
	R
R	146

	Hasil Prediksi			
	5	**S**	**j**	**S**
S	4	118	1	23

	Hasil Prediksi	
	T	**t**
T	141	5

	Hasil Prediksi			
	U	**V**	**b**	**u**
U	102	2	1	42

	Hasil Prediksi		
	U	**V**	**v**
V	2	80	65

	Hasil Prediksi		
	V	**W**	**w**
W	1	103	43

	Hasil Prediksi	
	X	**x**
X	82	65

	Hasil Prediksi				
	4	**V**	**Y**	**t**	**Y**
Y	2	1	117	1	25

	Hasil Prediksi		
	2	**Z**	**z**
Z	7	105	35

	Hasil Prediksi				
	0	**9**	**O**	**a**	**Q**
a	2	1	1	142	2

	Hasil Prediksi	
	6	**b**
b	1	146

	Hasil Prediksi	
	C	**c**
c	49	98

	Hasil Prediksi
	d
d	146

	Hasil Prediksi
	e
e	147

	Hasil Prediksi			
	F	**f**	**s**	**T**
f	15	130	1	1

	Hasil Prediksi		
	9	**g**	**q**
g	1	142	4

	Hasil Prediksi	
	L	**h**
h	1	146

	Hasil Prediksi			
	1	**I**	**i**	**L**
i	39	16	85	6

	Hasil Prediksi					
	J	**S**	**g**	**i**	**j**	**T**
j	23	2	1	1	118	1

	Hasil Prediksi		
	K	**h**	**k**
k	59	2	85

	Hasil Prediksi			
	1	**I**	**i**	**l**
l	39	57	3	48

	Hasil Prediksi	
	M	**m**
m	74	73

	Hasil Prediksi
	n
n	147

	Hasil Prediksi		
	0	**O**	**o**
o	16	14	117

	Hasil Prediksi	
	P	**p**
p	17	130

	Hasil Prediksi			
	9	**P**	**a**	**q**
q	24	1	1	121

	Hasil Prediksi		
	I	**Y**	**r**
r	1	1	145

	Hasil Prediksi			
	3	**5**	**S**	**s**
s	1	5	55	86

	Hasil Prediksi	
	f	**t**
t	2	144

	Hasil Prediksi			
	U	**V**	**u**	**y**
u	12	1	132	1

	Hasil Prediksi			
	U	**V**	**u**	**v**
v	1	31	1	114

	Hasil Prediksi		
	H	**W**	**w**
w	1	21	125

	Hasil Prediksi		
	X	**t**	**x**
x	21	1	125

	Hasil Prediksi			
	4	**V**	**Y**	**y**
y	1	1	25	119

	Hasil Prediksi		
	2	**Z**	**z**
z	2	36	109

BIODATA PENULIS

Penulis bernama Dewi Ayu Nirmalasari, bungsu dari tiga bersaudara yang lahir pada tanggal 18 Desember 1997 di Surabaya. Penulis telah mengenyam pendidikan di Sekolah Dasar Negeri Jember Lor 1 tahun 2004 hingga 2010, Sekolah Menengah Pertama Negeri 2 Jember pada tahun 2010 hingga 2013, dan Sekolah Menengah Atas Negeri 1 Jember pada tahun 2013 hingga 2016. Pada masa penulisan, penulis sedang menempuh masa studi S1 di Institut Teknologi Sepuluh Nopember, Surabaya di Departemen Teknik Informatika.

Penulis aktif dalam organisasi dan kepanitiaan Himpunan Mahasiswa Teknik Computer (HMTC) dan Schematics diantaranya menjadi staf Departemen Dalam Negeri HMTC ITS 2017-2018, staf ahli Departemen Kaderisasi dan Pemetaan Mahasiswa HMTC ITS 2018-2019, Bendehara III Schematics ITS 2017 dan Bendahara I Schematics ITS 2018. Komunikasi dengan penulis dapat melalui e-mail: dnirmalasari18@gmail.com.