IMPLEMENTASI CONVOLUTIONAL NEURAL NETWORK PADA PENGENALAN TULISAN TANGAN

Hamzah Nur Al Falah [1], Ken Kinanti Purnamasari [2]

[1,2] Universitas Komputer Indonesia
Jl. Dipati Ukur No.112-116, Lebakgede, Coblong, Kota Bandung, Jawa Barat 40132
E-mail: hamzahnuralfalah@gmail.com[1], ken.kinanti@email.unikom.ac.id[2]

ABSTRAK

Pada penelitian sebelumnya telah dilakukan *Intelligence Character Recognition* (ICR) menggunakan *Convolutional Neural Network* (CNN) dengan tingkat akurasi mencapai 97,6% pada *Arabic Handwritten Character Dataset*. Pada penelitian lainnya CNN dikatakan unggul dalam kasus pengenalan menggunakan data citra. Dari jejak penelitian tersebut, dilakukan penelitian dengan fokus utama mengukur kemampuan CNN pada kasus ICR alfabet Inggris dengan data berasal dari *National Institute of Standards and Technology* (NIST) sebanyak 15,872 data latih dan 3,100 data uji dengan kelas karakter meliputi A-Z, a-z, dan 0-9. *Preprocessing* dilakukan sebelum pelatihan dan pengujian CNN yang terdiri dari tahap *grayscaling*, *smoothing*, *thresholding*, segmentasi, dan *scaling*. Tahap *feedforward* CNN terdiri dari *layer* ekstraksi dan *layer* klasifikasi. Setelah *feedforward*, dilakukan *backpropagation* untuk memperbaiki nilai bobot dan bias ketika pelatihan sehingga dapat digunakan untuk pengujian. Dengan parameter uji berupa *learning rate* yang terdiri dari 0.001, 0.002, 0.003, 0.004, dan 0.005 serta *epoch* sebanyak 28 diperoleh akurasi sebesar 72.48% pada tahap pengujian.

Kata Kunci: Kecerdasan Buatan, *Neural Network*, *Convolutional Neural Network*, CNN, *Intelligence Character Recognition*, ICR, *Handwritten Character Recognition*

1. PENDAHULUAN

Pada penelitian terdahulu masih didapati banyak kekurangan, terutama dalam mengenali karakter - karakter dengan ciri yang mirip [1]. Hal ini berkaitan dengan kemampuan metode dalam mengekstraksi ciri dari citra.

Telah dilakukan penelitian terkait ICR menggunakan *Artificial Neural Network* (ANN) dan *gradient feature extraction* dengan akurasi mencapai 86,2% [1]. Dalam penelitian lain diperoleh akurasi mencapai 85,62% menggunakan metode *multilayer feed forward neural network* [2], dan penelitian lainnya memperoleh akurasi 74 – 78% menggunakan ANN [3].

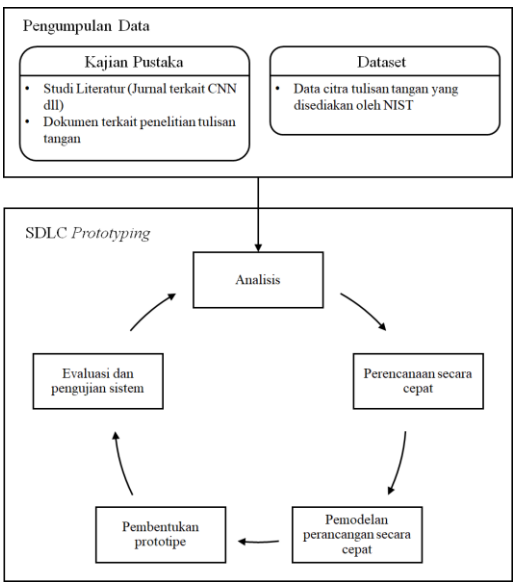
CNN merupakan metode yang memanfaatkan fitur konvolusi untuk mengekstraksi ciri citra, hal ini yang membedakan metode CNN dengan ANN. Pada penelitian sebelumnya telah dilakukan ICR menggunakan CNN dengan tingkat akurasi mencapai 97,6% menggunakan *Arabic Handwritten Character Dataset* [4]. Pada penelitian lainnya diperoleh akurasi mecapai 89% dengan kasus pengenalan wajah secara *realtime* [5], 96,1 – 97,9% lebih unggul daripada delapan metode lainnya termasuk RBF dan SVM pada pengenalan jenis kelamin berdasarkan wajah [6]. Dengan adanya pencapaian tingkat akurasi yang baik dari penelitian terdahulu, maka digunakanlah CNN untuk kasus ICR alfabet Inggris dengan data yang diperoleh dari NIST [7].

2. ISI PENELITIAN

Pada bab ini akan dijelaskan kajian terkait metode penelitian, arsitektur sistem, pengolahan citra digital, algoritma CNN, dan hasil pengujian CNN.

2.1 Metode Penelitian

Penelitian ini dilakukan dengan memperhatikan alur penelitian berikut.



Gambar 1. Alur Penelitian

Penelitian diawali dengan tahap pengumpulan data guna mengetahui pencapaian dari penelitian sebelumnya, menghindari kesalahan – kesalahan yang pernah dilakukan, dan memperoleh saran serta

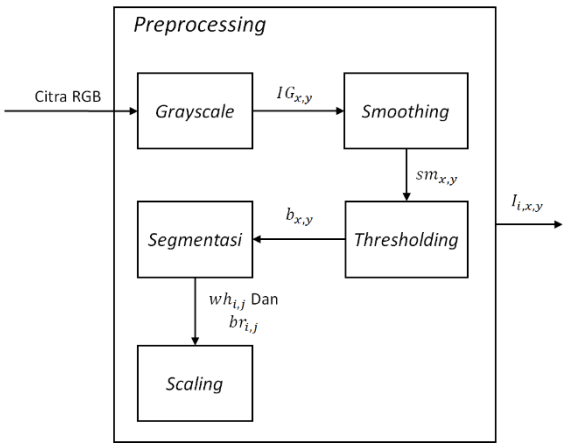
petunjuk agar penelitian yang akan dilakukan dapat berjalan dengan lancar serta membuahkan hasil yang lebih baik. Pengumpulan data yang dilakukan adalah mengumpulkan jurnal terkait metode yang digunakan, mencari *dataset* tulisan tangan.

Dalam *System Development Life Cycle* (SDLC) digunakan model *prototyping* yang terdiri dari analisis, perencanaan secara cepat, pemodelan perencanaan secara cepat, pembentukan prototipe dan evaluasi pengujian sistem. Berikut adalah penjelasan dari setiap tahap yang dilakukan pada penelitian ini.

a. Analisis, mengkaji metode – metode yang akan digunakan dalam sistem menggunakan buku – buku dan jurnal terkait.
b. Perencanaan secara cepat, menentukan tahapan – tahapan proses yang akan digunakan pada perangkat lunak, yang di dalamnya termasuk pemilihan metode.
c. Pemodelan perancangan secara cepat, memulai konstruksi pembuatan tampilan dan bentuk program.
d. Pembentukan prototipe, mengimplementasikan model dari perancangan cepat secara menyeluruh menjadi sebuah perangkat lunak.
e. Evaluasi dan pengujian sistem, mencari kesalahan kode program atau *error logic* dari prototipe yang telah selesai dibangun. Menguji hasil dari beberapa penggunaan parameter pada metode CNN, dicari kelemahan dan kekurangan dari prototipe yang ada seperti pemilihan parameter atau metode yang digunakan.

2.2 Pengolahan Citra Digital

Pengolahan citra merupakan cabang ilmu informatika yang bertujuan untuk memperbaiki kualitas citra, agar sesuai dengan kebutuhan. Pada penelitian ini, pengolahan citra dilakukan pada tahap *preprocessing*. Tahap utama dari *preprocessing* adalah *grayscaling*, *smoothing* dengan citra integral, *thresholding* menggunakan *Sauvola Threshold* dikombinasikan dengan citra integral, segmentasi menggunakan *Connected Component Labeling* (CCL), dan *scaling*. Berikut adalah diagram *block* untuk *preprocessing* yang digunakan.



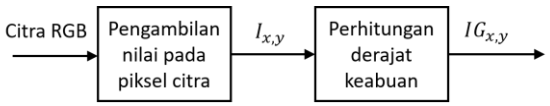
Gambar 2. Arsitektur Tahap *Preprocessing*

2.2.1 Citra RGB

Citra RGB atau citra berwarna merupakan citra yang memiliki tiga macam kanal warna yaitu *Red*, *Green*, dan *Blue*. Citra ini memiliki 24 bit warna untuk setiap pikselnya dengan rentang 0 sampai 255.

2.2.2 Konversi Citra RGB ke *Grayscale*

Tahap ini bertujuan untuk mengurangi kompleksitas komputasi, dengan mengubah rentang warna RGB pada citra menjadi derajat keabuan dengan satu buah kanal warna [8]. Adapun arsitektur tahap ini dapat dilihat sebagai berikut.



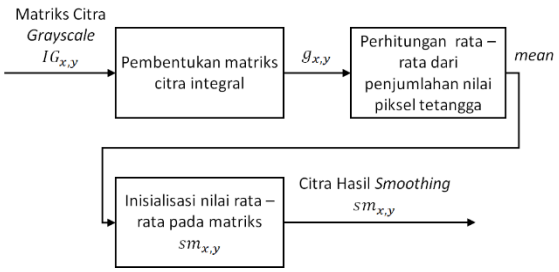
Gambar 3. Diagram *Block* Tahap *Grayscale*

Pada tahap *grayscaling* ini terdiri dari dua tahap, yang pertama adalah pengambilan nilai piksel citra RGB, dan tahap kedua adalah mengonversi nilai tersebut ke citra derajat keabuan atau *grayscale*.

Dalam pengenalan sebuah citra, kompleksitas akan berpengaruh terhadap performa dari sistem, sehingga penggunaan citra *grayscale* dibandingkan dengan citra RGB memiliki waktu eksekusi dan kompleksitas yang lebih rendah. Terlebih lagi, dengan penggunaan satu kanal warna ini, dapat ditentukan juga jumlah bit yang akan digunakan untuk memudahkan penyeragaman warna.

2.2.3 Citra Integral

Citra integral merupakan konsep menjumlahkan setiap piksel citra dengan nilai tetangganya untuk menyimpan informasi jumlah keseluruhan intensitas citra [9]. Berikut adalah diagram *block* dari tahap citra integral.



Gambar 4. Diagram *Block* Tahap Citra Integral

Adapun langkah pembentukan citra integral adalah sebagai berikut.

a. Pembuatan data matriks citra integral $g_{x,y}$ dari citra *grayscale*. Tahap pembentukan matriks citra integral menggunakan persamaan 1.

$$g_{x,y} = \sum_{i=0}^{x}\sum_{j=0}^{y} I_{i,j} \quad (1)$$
$$x = 0,1,2,..,M-1$$
$$y = 0,1,2,..,N-1$$

Nilai dari koordinat x dan y berdasarkan ukuran matriks citra *grayscale* yang direpresentasikan dengan $M \times N$.

b. Menentukan dimensi *mask* untuk rentang perhitungan rata – rata citra integral pada tahap selanjutnya.

c. Perhitungan rata – rata citra integral pada rentang dimensi *mask*.

$$mean(x,y) = \frac{\left(g_{x+dx-1,y+dy-1} + g_{x-dx,y-dy}\right) - \left(g_{x-dx,y+dy-1} + g_{x+dx-1,y-dy}\right)}{wx \times wy}$$

$$dx = round\left(\frac{wx}{2}\right), \quad dy = round\left(\frac{wy}{2}\right),$$

$$x = 0,1,2..M-1, \quad y = 0,1,2..N-1 \tag{2}$$

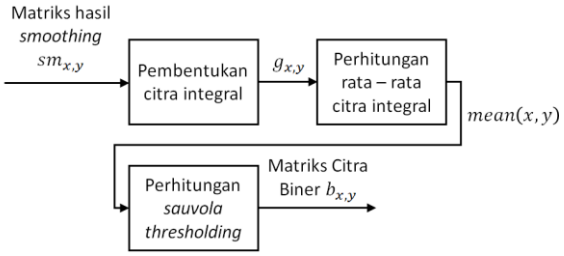
Perhitungan rata – rata ini dilakukan berdasarkan rentang *mask* yang telah ditentukan. Nilai dari rentang *mask* akan mempengaruhi nilai wx dan wy sehingga mempengaruhi nilai matriks citra integral yang digunakan.

Pada dasarnya penggunaan citra integral ini mirip dengan *mean filtering*, yaitu salah satu metode *smoothing* yang memperoleh nilai melalui rata – rata penjumlahan nilai piksel dalam rentang *mask*. Berbeda dengan *mean filtering* yang memperoleh nilai dengan mencacah satu persatu nilai pada rentang *mask*, citra integral memanfaatkan perhitungan matematis pada koordinat matriks citra integral untuk memperoleh nilai rata – rata.

Pada penelitian ini, persamaan 2 digunakan untuk tahap *smoothing* yang direpresentasikan dengan $sm(x,y)$, dan pada tahap *thresholding* sebagai salah satu variabel perhitungan *sauvola threshold*.

2.2.4 Citra Biner

Citra biner merupakan citra dengan satu kanal warna berukuran 2 bit. Citra ini hanya terdiri dari nilai nol (0) dan satu (1). *Cell* pada matriks citra biner yang bernilai satu merepresentasikan warna putih, sedangkan nol merepresentasikan warna hitam. Matriks biner ini digunakan sebagai data masukan pada tahap segmentasi. Citra biner sendiri diperoleh dengan memanfaatkan nilai ambang batas sebagai batasan dan penentu suatu nilai. Berikut adalah diagram *block* tahap *thresholding*.



Gambar 5. Diagram *Block* Tahap *Thresholding*

Matriks yang digunakan sebagai *input* pada tahap ini adalah hasil dari tahap *smoothing* $(sm_{x,y})$. Kasus pada penelitian ini menggunakan data citra dengan *background* berwarna hitam, sehingga perhitungan citra biner dilakukan dengan aturan pada persamaan berikut.

$$b_{x,y} = \begin{cases} 1, & sm_{x,y} \le T \\ 0, & sm_{x,y} > T \end{cases} \tag{3}$$

Matriks $b_{x,y}$ merupakan representasi dari citra biner, sedangkan variabel *T* merupakan nilai ambang batas yang digunakan.

Thresholding atau ambang batas, merupakan nilai yang digunakan sebagai acuan dalam penentu suatu hal. Baik untuk mengambil keputusan atau dalam kasus ini untuk menentukan apakah nilai dari matriks hasil *smoothing* $sm_{x,y}$ pada koordinat tertentu berwarna putih atau hitam. Nilai yang diberikan pada variabel ini akan mempengaruhi matriks biner yang dihasilkan. Oleh karena itu digunakan metode *sauvola threshold* yang merupakan salah satu dari metode *Locally Adaptive Thresholding*, yaitu metode yang mampu menentukan nilai ambang batas secara otomatis berdasarkan nilai piksel ketetanggan dalam rentang dimensi *mask* dan dikombinasi dengan citra integral [9]. Berikut adalah persamaan yang digunakan.

$$T(x,y) = mean(x,y)\left[1 + k\left(\frac{\partial(x,y)}{1 - \partial(x,y)} - 1\right)\right]$$

$$\partial(x,y) = sm(x,y) - mean(x,y)$$

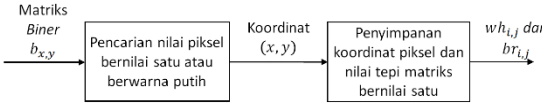
$$\partial(x,y) = \begin{cases} -1, & \partial(x,y) = 1 \\ \partial(x,y), & \partial(x,y) \neq 1 \end{cases} \tag{4}$$

Keterangan:

$T(x,y)$	=	Nilai ambang batas hasil *sauvola*
$mean(x,y)$	=	Matriks hasil citra integral dengan *input* $sm(x,y)$
$\partial(x,y)$	=	Standar deviasi
k	=	Nilai bias dengan rentang [0.1,0.9], yang digunakan dalam penelitian ini adalah 0.1

2.2.5 Segmentasi Citra

Metode yang digunakan adalah *Connected Component Labeling* (CCL). CCL mampu menyegmentasi citra berdasarkan hubungan antar piksel dengan memanfaatkan dimensi *mask* pada tahap pencarian, pada penelitian ini digunakan *mask* berdimensi 5×5. Berikut adalah diagram *block* dari tahap CCL.



Gambar 6. Diagram *Block* Tahap Segmentasi

Adapun variabel yang perlu didefinisikan dari tahap segmentasi adalah sebagai berikut.

Tabel 1. Denisi Variabel Tahap Segmentasi

No	Variabel	Keterangan
1	$wh_{i,j}$	Matriks yang menyimpan koordinat piksel berwarna putih (x, y), dengan variabel i sebagai indeks untuk objek karakter dan j sebagai indeks untuk nilai koordinat piksel.
2	$br_{i,j}$	Matriks untuk menyimpan koordinat dari batas terluar objek karakter dengan variabel i sebagai indeks objek karakter dan j sebagai indeks nilai koordinat. Variabel j terdiri dari empat buah nilai, indeks pertama merepresentasikan batas teratas, indeks kedua untuk batas paling kanan, indeks ketiga untuk batas paling bawah, dan indeks keempat untuk batas paling kiri.

Dari gambar 6 tersebut dapat dikembangkan dan dijelaskan dengan tahapan sebagai berikut.

a. Pencarian nilai piksel bernilai satu dimulai dari koordinat $(0,0)$ pada matriks citra biner $b_{x,y}$. Jika ditemukan nilai piksel satu (1) maka simpan data koordinat tersebut pada matriks $wh_{i,j}$, lalu *set* nilai piksel pada $b_{x,y}$ tersebut menjadi nol (0).

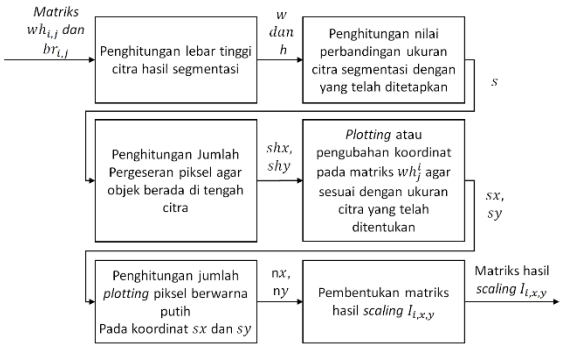
b. Lakukan pengisian nilai pada matriks $br_{i,j}$. Apabila matriks $br_{i,j}$ belum memiliki nilai, maka cukup inisialisasikan koordinat x pada indeks pertama dan indeks ketiga, koordinat y pada indeks kedua dan keempat. Selain daripada kondisi pertama, berlaku ketentuan pada persamaan sebagai berikut.

$$br_{i,0} = \begin{cases} x, & if(x < br_{i,0}) \\ br_{i,0}, & lainnya \end{cases}$$
$$br_{i,1} = \begin{cases} y, & if(y > br_{i,1}) \\ br_{i,1}, & lainnya \end{cases}$$
$$br_{i,2} = \begin{cases} x, & if(x > br_{i,2}) \\ br_{i,2}, & lainnya \end{cases}$$
$$br_{i,3} = \begin{cases} y, & if(y < br_{i,3}) \\ br_{i,3}, & lainnya \end{cases} \quad (5)$$

c. Pencarian selanjutnya dimulai berdasarkan dimensi *mask*, dengan pusat *mask* berada pada koordinat yang telah disimpan pada matriks $wh_{i,j}$ sebelumnya. Seperti pada tahap b, jika ditemukan piksel bernilai satu (1) pada matriks $b_{x,y}$ maka simpan koordinatnya pada matriks $wh_{i,j}$ dan *set* nilai matriks $b_{x,y}$ tersebut menjadi nol (0). Lakukan kembali tahap c dan d hingga tidak ditemukan lagi piksel bernilai satu pada dimensi *mask* untuk matriks $b_{x,y}$. Untuk pencarian objek karakter selanjutnya, dimulai dari koordinat pertama yang disimpan pada matriks $wh_{i,j}$.

2.2.6 *Scaling*

Scaling pada penelitian ini bertujuan untuk menyeragamkan ukuran citra hasil segmentasi, yaitu matriks $wh_{i,j}$ dan $br_{i,j}$. Nilai koordinat dari citra dengan ukuran yang beragam tersebut perlu dikonversi agar seragam, yaitu sebesar 28×28. Berikut adalah diagram *block* dari tahap *scaling*.



Gambar 7. Diagram *Block* Tahap *Scaling*

Adapun langkah pengerjaannya adalah sebagai berikut.

a. Menghitung lebar (w) dan tinggi (h) citra hasil segmentasi menggunakan persamaan berikut.

$$w = br_{i,1} - br_{i,3} + 1$$
$$h = br_{i,2} - br_{i,0} + 1 \quad (6)$$
$$i = 1,2..,Jumlah\ Objek$$

b. Menghitung nilai perbandingan ukuran citra hasil segmentasi (s) dengan ukuran yang telah ditetapkan $(u = 28)$.

$$s = \begin{cases} \frac{u}{w}, & if(w \geq h) \\ \frac{u}{h}, & if(w < h) \end{cases} \quad (7)$$

c. Menghitung jumlah pergeseran setiap piksel pada koordinat x dan y untuk citra hasil *scaling*.

$$shx = \begin{cases} round\left(\frac{u}{2}\right) - round\left(\frac{s \times h}{2}\right), & if(w \geq h) \\ 0, & if(w < h) \end{cases}$$
$$shy = \begin{cases} round\left(\frac{u}{2}\right) - round\left(\frac{s \times w}{2}\right), & if(w < h) \\ 0, & if(w \geq h) \end{cases} \quad (8)$$

Keterangan:

shx = Jumlah pergeseran piksel pada koordinat baris

shy = Jumlah pergeseran piksel pada koordinat kolom

d. *Scaling* koordinat, ide pada tahap ini adalah mengonversi nilai koordinat yang ada pada matriks $wh_{i,j}$ terhadap matriks berukuran 28×28.

$$sx = (x - br_{i,0}) \times s + shx$$
$$sy = (y - br_{i,3}) \times s + shy \quad (9)$$

Keterangan:

sx = Koordinat x hasil *scaling*
sy = Koordinat y hasil *scaling*
x = Koordinat x yang diperoleh dari nilai matriks $wh_{i,j}$
y = Koordinat y yang diperoleh dari nilai matriks $wh_{i,j}$

e. Menghitung jumlah *plotting* piksel berwarna putih dimulai dari sx dan sy. Tahap ini bertujuan untuk menentukan berapa banyak piksel berwarna putih yang diperoleh dari satu koordinat pada matriks $wh_{i,j}$ ketika berukuran 28×28.

$$nx = \begin{cases} 2, & if(sx \bmod 1 > 0 \; and \; sx > 1) \\ 1, & lainnya \end{cases}$$
$$ny = \begin{cases} 2, & if(sy \bmod 1 > 0 \; and \; sy > 1) \\ 1, & lainnya \end{cases} \quad (10)$$

Keterangan:

nx = Jumlah *plotting* piksel berwarna putih dimulai dari sx
ny = Jumlah *plotting* piksel berwarna putih dimulai dari sy

2.3 Algoritma CNN

CNN memiliki dua buah *layer* utama, yaitu *layer* konvolusi dan *layer* klasifikasi. *Layer* konvolusi terdiri dari tahap konvolusi dan *average pooling*. Diantara tahap konvolusi dengan *pooling* terdapat penggunaan fungsi aktivasi. Terdapat banyak variasi fungsi aktivasi yang biasa digunakan pada *Neural Network*, namun pada penelitian ini fungsi aktivasi yang digunakan adalah *Leaky Rectified Linear Unit* (LReLU). Berikut adalah persamaan yang digunakan pada tahap LReLU.

$$y_i = \begin{cases} x_i & if \; x_i \geq 0 \\ \frac{x_i}{a_i} & if \; x_i < 0 \end{cases} \quad (11)$$

Keterangan:

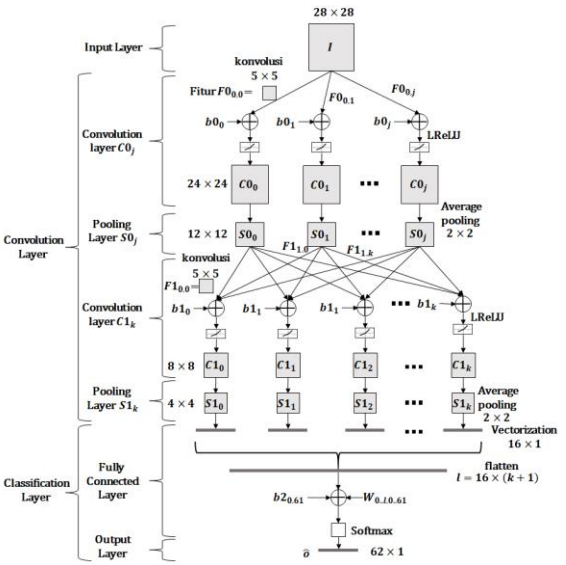
y_i = Hasil dari perhitungan fungsi aktivasi LReLU
x_i = Nilai hasil konvolusi
a_i = Nilai *artificial* dengan rentang satu $[1, \infty]$

Dalam penelitian ini digunakan nilai $a_i = 5.5$ merujuk pada penelitian oleh Bing Xu et al. [10].

2.3.1 Arsitektur CNN

Arsitektur CNN yang digunakan pada penelitian ini memiliki dua tahap pada konvolusi dan *average pooling* seperti yang tertera pada gambar 2. Citra yang menjadi masukan CNN merupakan citra hasil *preprocessing* berukuran 28×28 dengan nilai berada pada rentang satu atau nol.

Adapun jumlah *neuron* pada konvolusi pertama, kedua, dan *fully connected* masing – masing adalah 6 *neuron* (Indeks j), 12 *neuron* (Indeks k), dan 62 *neuron* (Indeks m). Matriks bobot yang digunakan untuk setiap lapisan konvolusi berukuran 5×5, sedangkan pada lapisan *fully connected* berukuran 192×62. *Array* bias untuk setiap lapisan berukuran sama dengan jumlah *neuron* pada lapisan tersebut.



Gambar 8. Arsitektur CNN

Konvolusi dilakukan tanpa menggunakan *padding*. Sehingga dengan menggunakan fitur atau bobot berukuran 5×5 akan mengurangi ukuran citra *input*, menjadi 24×24 pada konvolusi pertama. Pada setiap tahap konvolusi, dilakukan pergeseran fitur atau bobot sebanyak satu *stride*.

Hasil yang diperoleh dari tahap konvolusi dihitung menggunakan persamaan fungsi aktivasi *LReLU*. Selanjutnya dilakukan *average pooling* berukuran 2×2 dengan perpindahan 2 *stride* untuk baris dan kolom.

Setelah tahap *average pooling* pertama, sistem akan memperoleh matriks berukuran 12×12 dan akan menjadi *input* untuk tahap konvolusi kedua. Pada konvolusi kedua akan dihasilkan matriks berukuran 8×8 yang akan menjadi *input* bagi tahap *average pooling* kedua. Lalu tahap terakhir dari *convolution layer*, yaitu *average pooling* kedua yang menghasilkan matriks berukuran 4×4.

Dalam perhitungan *error* digunakan metode *Cross Entropy Error Function* yang juga berfungsi untuk mereduksi tingkat *error* pada bobot dan bias untuk tahap *backpropagation*.

2.3.2 Pengenalan Variabel dan Inisialisasi

Tahap awal dari CNN adalah inisialisasi matriks bobot dengan nilai *random*, dan *aray* bias dengan

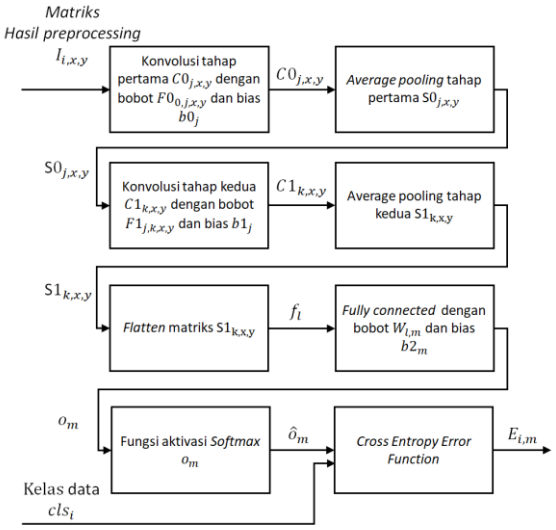
nilai nol (0). Berikut adalah variabel yang digunakan pada arsitektur CNN.

Tabel 2. Keterangan Variabel Pada CNN

Lapisan	Variabel	Keterangan
Konvolusi I	$F0_{0,j,x,y}$	Bobot
	$b0_j$	Bias
	$C0_{j,x,y}$	Hasil Konvolusi I
Pooling I	$S0_{j,x,y}$	Hasil *Pooling* I
Konvolusi II	$F1_{j,k,x,y}$	Bobot
	$b1_k$	Bias
	$C1_{k,x,y}$	Hasil Konvolusi II
Pooling II	$S1_{k,x,y}$	Hasil *Pooling* II
Fully Connected	$W_{l,m}$	Bobot
	$b2_m$	Bias
	o	Hasil *fully connected*
Output	$\hat{o}$	Hasil *softmax*

2.3.3 *Feedforward* CNN

Feedforward merupakan alur maju pada *Neural Network* (NN) yang memungkinkan sistem berjalan sesuai dengan modul yang digunakan. Pada tahap ini dilakukan beberapa tahapan yang dapat dilihat pada diagram *block* berikut.



Gambar 9. Diagram *Block Feedforward*

2.3.3.1 Konvolusi Pertama

Konvolusi pertama pada CNN bertujuan untuk mengekstraksi ciri citra *input* berukuran 28×28 menjadi 24×24 yang direpresentasikan dengan $C0_{j,x,y}$ menggunakan bobot $F0_{0,j,x,y}$ berukuran $fx \times fy$ (5×5) dan bias $b0_j$ dengan pergeseran (*stride*) bobot 1×1. Selanjutnya dilakukan perhitungan fungsi aktivasi *Leaky ReLU* (LReLU).

2.3.3.2 *Average Pooling* Pertama

Average pooling pertama direpresentasikan dengan $S0_{j,x,y}$ dengan matriks *input* $C0_{j,x,y}$ dari konvolusi pertama dengan *stride* $stx \times sty$ (2×2). Proses ini akan mengubah matriks berukuran 24×24 menjadi 12×12.

2.3.3.3 Konvolusi Kedua

Konvolusi kedua direpresentasikan dengan $C1_{k,x,y}$ dengan matriks *input* $S0_{j,x,y}$ dengan *stride* 1×1. Proses ini akan mengubah ukuran matriks 12×12 menjadi 8×8 menggunakan bobot $F1_{j,k,x,y}$ berukuran $fx \times fy$ (5×5) dan bias $b1_k$. Selanjutnya dilakukan perhitungan dengan fungsi aktivasi LReLU.

2.3.3.4 *Average Pooling* Kedua

Average pooling kedua direpresentasikan dengan $S1_{k,x,y}$. Dengan menggunakan *stride* $stx \times sty$ (2×2) matriks berukuran 8×8 akan berukuran 4×4 di akhir proses ini.

2.3.3.5 *Flatten*

Flatten merupakan tahap penggabungan matriks $S1_{k,x,y}$ berukuran 4×4 yang diubah terlebih dahulu menjadi *vector* sehingga berjumlah 16 data nilai. Pada lapisan *pooling* kedua terdapat 12 data *pooling* berdasarkan jumlah *neuron* yang digunakan, sehingga *neuron* pada tahap *flatten* akan berjumlah 192 *neuron* yang direpresentasikan dengan *array* f_l.

2.3.3.6 *Fully Connected* dan *Softmax*

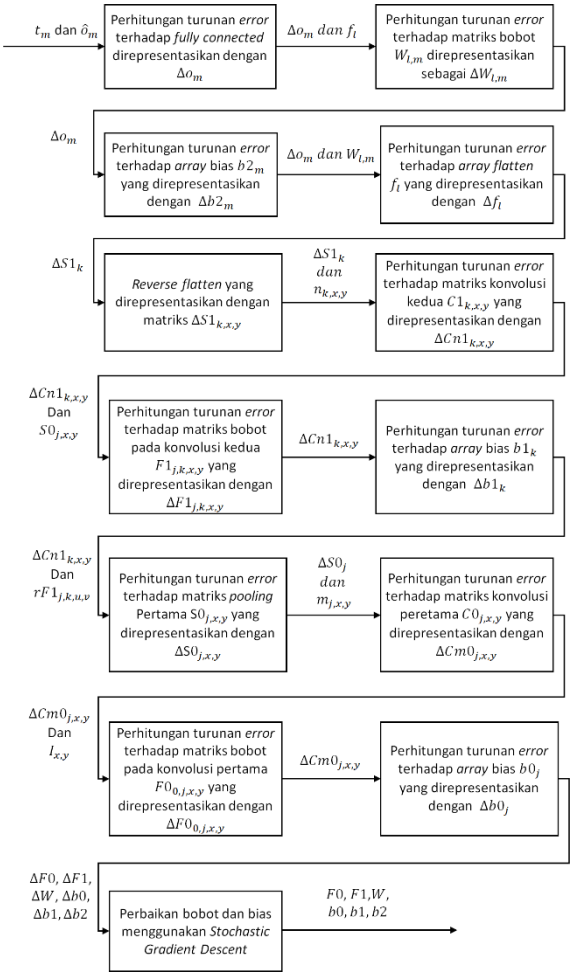
Fully Connected (o_m) pada penelitian ini bertujuan untuk menggabungkan setiap *neuron flatten* terhadap sejumlah *neuron output*, sedangkan *softmax* $(\hat{o}_m)$ bertujuan untuk menghitung probabilitas dari hasil *feedforward*.

2.3.3.7 *Cross Entropy Error Function*

Metode ini dipilih untuk menghitung dan mereduksi *error* yang diperoleh dari tahap *feed forward* pelatihan CNN. Metode ini tidak hanya mampu mereduksi *error* pada nilai bobot yang mencirikan kelas yang sedang dilatih, namun juga mengubah nilai untuk kelas lainnya [11].

2.3.4 *Backpropagation* CNN

Backpropagation digunakan sebagai tahap untuk mereduksi *error* terhadap setiap nilai bobot dan bias. Berikut adalah diagram *block* dari tahap *backpropagation*.



Gambar 10. Diagram *Block Backpropagation*

Tahap ini mengharuskan sistem untuk menghitung *error* dari *cross entropy* hingga proses paling awal dari *feedforward*. Tahap ini akan menghasilkan nilai bobot dan bias yang diperbaharui dan hanya akan dilakukan pada *training* CNN. Bobot dan bias yang telah diperbaiki akan digunakan untuk keperluan *testing*.

2.3.5 Hasil Pengujian CNN

Pada penelitian ini digunakan data sebanyak 15,872 data latih dan 3,100 data uji yang berasal dari NIST *dataset* dengan parameter pelatihan pada tabel 2.

Tabel 3. Parameter Pelatihan

No	Parameter	Nilai
1	Epoch	28
2	Target error	0.00000001
3	*Learning rate*	$0.001, 0.002, 0.003, 0.004, 0.005$

Dari hasil pengujian diperoleh akurasi untuk setiap *learning rate* sebagai berikut.

Tabel 4. Perbandingan Akurasi Beberapa *Learning Rate*

Epoch	Akurasi Berdasarkan Learning Rate				
	0.001	**0.002**	**0.003**	**0.004**	**0.005**
1	48.71%	46.74%	45.39%	47.06%	43.1%
4	57.39%	57.35%	58.61%	58.87%	54.74%
8	59.03%	59.58%	61.87%	59.19%	59.23%
12	61.19%	62.19%	60.65%	60.94%	58.61%
16	61.81%	61.97%	59.45%	60.35%	56.58%
20	61.03%	61.68%	61.26%	61.52%	58.81%
24	62.39%	62.16%	61.06%	61.45%	56.81%
28	72.48%	66.13%	65.23%	58.94%	57.81%

Pada tabel 4, dapat diperhatikan bahwa *cell* berwarna keabuan merupakan perolehan akurasi tertinggi untuk setiap kelompok *learning rate*. Sedangkan jika dibandingkan secara menyeluruh, akurasi tertinggi diperoleh dengan menggunakan $learning\ rate = 0.001$ pada *epoch* ke-28.

3. PENUTUP

Berdasarkan hasil pengujian, penggunaan metode CNN pada ICR karakter Inggris dengan menggunakan *dataset* NIST memperoleh akurasi tertinggi untuk *learning rate* = 0.001 pada *epoch* ke-28.

Tinggi rendahnya akurasi yang diperoleh dipengaruhi oleh berbagai faktor yang masih belum terpecahkan seluruhnya. Hal yang dapat dikembangkan untuk penelitian selanjutnya adalah menggunakan arsitektur yang berbeda seperti penggunaan metode yang berbeda, baik di CNN ataupun pada tahap *preprocessing* hingga diperoleh tingkat akurasi yang lebih baik.

DAFTAR PUSTAKA

[1] R. A. Rahim, L. U. A. Khalik dan M. N. S. Zainudin, "Handwritten English Character Recognition Using Gradient Feature Extraction," *International Journal For Advance Research In Engineering and Technology*, vol. 3, 2015.

[2] A. Choudhary, R. Rishi dan S. Ahlawat, "Off-Line Handwritten Character Recognition using Features Extracted from Binarization Technique," *AASRI Conference on Intelligent Systems and Control*, vol. 4, 2013.

[3] R. K. Mandal dan N. R. Manna, "Handwritten English Character Recognition using Hoof Segmentation of Image Matrix (HSIM)," *AMSE Journals*, vol. 57, 2014.

[4] K. S. Younis, "Arabic Handwritten Character Recognition Based On Deep Convolutional Neural Networks," *Jordanian Journal of*

Computers and Information Technology (JJCIT), vol. 3, 2017.

[5] M. Zufar dan B. Setiyono, "Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time," *SAINS DAN SENI ITS*, vol. 5, p. 6, 2016.

[6] J. Lemley, S. Abdul-Wahid, D. Banik dan R. Andonie, "Comparison of Recent Machine Learning Techniques for Gender Recognition from Facial Images," *MAICS*, pp. 97 - 102, 2016.

[7] NIST, "NIST," NIST, 27 08 2010. [Online]. Available: https://www.nist.gov/srd/nist-special-database-19. [Diakses 29 04 2018].

[8] P. Hidayatullah, Pengolahan Citra Digital Teori dan Aplikasi Nyata, Bandung: Informatika, 2017.

[9] T. R. Singh, S. Roy, O. I. Singh, T. Sinam dan K. M. Singh, "A New Local Adaptive Thresholding Technique in Binarization," *International Journal of Computer Science Issues (IJCSI)*, vol. 8, no. 6, p. 2, 2011.

[10] B. Xu, N. Wang, T. Chen dan M. Li, "Empirical Evaluation of Rectified Activations in Convolution Network," *arXiv*, 2015.

[11] D. Jurafsky dan J. H. Martin, Speech and Language Processing; An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 3rd penyunt., California: Stanford University, 2018.