PENGIMPLEMENTASIAN OCR MENGGUNAKAN CNN UNTUK EKSTRAKSI TEKS PADA GAMBAR

Ivan Wijaya [1)] Chairisni Lubis [2)]

[1) 2)] Teknik Informatika, FTI, Universitas Tarumanaraga
Jl. Letjen S Parman no 1, Jakarta 11440 Indonesia
email : ivan.535180030@stu.untar.ac.id[1)], Chairisnil@fti.untar.ac.id[2)]

ABSTRACT

OCR merupakan sebuah sistem yang digunakan untuk mengekstraksi tulisan yang terdapat pada sebuah gambar sehingga dapat mempercepat proses input data. Sistem OCR merupakan sebuah sistem yang terdiri dari 2 proses yaitu pendeteksian teks dan pengenalan teks. Pada perancangan ini, digunakan 2 model CNN untuk melakukan pendeteksian dan pengenalan teks. Digunakan CNN dengan struktur *Feature Pyramid Network* yang menggunakan *Backbone* VGG19 untuk mendeteksi lokasi teks pada gambar. Untuk mengenali teks pada gambar akan digunakan CNN dengan LSTM untuk melakukan pengenalan teks pada area gambar yang telah terdeteksi. Kedua CNN dilatih ddengan menggunakan dataset ICDAR 2015, COCO-Text, dan ICDAR 2019. Pada akhir pelatihan, didapatkan model pendeteksian teks dengan besaran *F1-Score* sebesar 49.18%, dan model pengenalan teks dengan besar *Correctly Recognized Word* sebesar 55.80%

Kata Kunci

CNN, Text Detection, Text Recognition, Python

1. Pendahuluan

Perkembangan teknologi telah banyak memberi pengaruh terhadap kehidupan manusia serta ilmu pengetahuan. Dengan perkembangan pengetahuan yang pesat, terdapat berbagai teknologi yang terus berkembang seperti diantaranya sistem *pattern recognition* dengan menggunakan *neural network*. Salah satu bidang yang terus berkembang dalam teknologi ini adalah OCR (*Optical Character Recognition*). OCR merupakan sebuah sistem untuk mengekstraksi tulisan yang terdapat pada gambar sehingga dapat mempercepat proses input data. Teks dalam sebuah gambar dapat di ekstrak secara langsung tanpa perlu di input oleh user.

Sistem OCR pada umumnya terbagi menjadi 2 proses berbeda yaitu proses pendeteksian teks dan proses pengenalan teks. Proses deteksi dalam sistem OCR memiliki fungsi untuk mendeteksi bagian dalam sebuah gambar yang memiliki kemungkinan merupakan sebuah text. Proses ini dapat dibuat secara manual dengan mendeteksi fitur sebuah teks dalam gambar, ataupun menggunakan deep learning untuk secara efektif mendeteksi teks menggunakan data training [7]. Pada proses pengenalan teks, teks akan dipisahkan per karakter dan di klasifikasikan sesuai dengan fitur yang ada pada karakter tersebut, sehingga pada akhir proses didapatkan karakter yang sesuai dengan karakter yang ada pada gambar. Karakter yang telah dikenali kemudian akan digabungkan sehingga membentuk sebuah teks utuh.

Skripsi ini akan membahas mengenai cara pengimplementasian sistem OCR dengan menggunakan CNN untuk mengekstraksi tulisan dalam sebuah gambar. Sistem ini ditujukan untuk mempercepat proses input data yang dilakukan oleh user dengan mengurangi banyak kata yang perlu diketik pada proses input data. Pada sistem ini, pengguna perlu menginput gambar yang memiliki tulisan yang ingin di ekstrak.

2. Dasar Teori

2.1 OCR

OCR (*Optical Character Recognition*) merupakan sebuah proses konversi gambar tulisan menjadi tulisan yang dapat dimengerti oleh mesin. Pada awalnya, proses OCR hanya dapat dilakukan pada satuan karakter dengan satu jenis font. Namun, sistem OCR modern telah dapat mengekstrak tulisan dengan berbagai jenis font dengan tingkat keakurasian yang tinggi [9].

Proses OCR banyak digunakan sebagai bentuk proses input data dari gambar dokumen seperti passport, invoice, kartu pengenal, dan dokumen lainnya. OCR merupakan proses yang umum digunakan sebagai metode pendigitalan dokumen fisik, sehingga dapat di cari dan diedit secara digital.

2.2 *Convolutional Neural Network*

CNN (*Convolutional Neural Network*) merupakan jenis dari *artificial neural network* yang umumnya digunakan untuk menganalisa gambar visual [11]. Pada CNN, tidak semua neuron pada sebuah lapisan berhubungan dengan neuron pada lapisan yang lain, hal ini ditujukan untuk mencegah *overfitting*. Melalui proses ini, CNN memerlukan proses *preprocessing* yang lebih sedikit dibandingkan algoritma pengklasifikasian gambar lainnya.

Pada CNN, sebuah input diproses menggunakan lapisan konvolusi untuk menghasilkan sebuah peta fitur. Pada sebuah lapisan konvolusi, informasi dalam sebuah data saling dikaitkan dan diteruskan ke lapisan berikutnya. Proses pengkaitan data inilah yang digunakan CNN untuk mempelajari data dan mengekstrak fitur [1]**Error! Reference source not found.**. Namun, sistem ini memiliki kelemahan dimana untuk memproses data input yang besar, diperlukan neuron yang banyak. Oleh karena itu, untuk memproses data yang besar, sebuah CNN perlu mengurangi jumlah parameter input dan memberikan bobot awal yang lebih teratur.

Sebuah CNN umumnya akan memiliki 2 bagian utama yaitu lapisan konvolusi dan lapisan *pooling* [12]. Pada lapisan konvolusi, dilakukan pengidentifikasian dan pembagian fitur-fitur yang terdapat dalam sebuah gambar berdasarkan keterhubungan setiap pixel sehingga dapat dilakukan analisa. Pada lapisan *pooling*, setiap fitur yang dideteksi pada lapisan konvolusi dihubungkan sehingga dapat dilakukan prediksi.

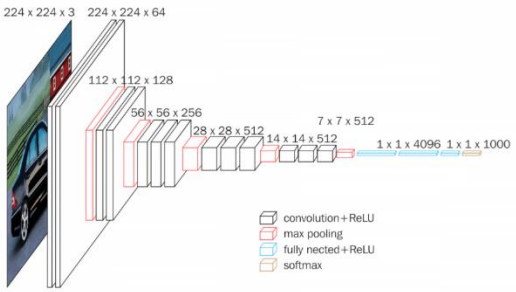
2.3 *Long Short-Term Memory*

Long Short-Term Memory (LSTM) merupakan arsitektur RNN yang digunakan pada bidang *Deep Learning* [2]. arsitektur LSTM memiliki perbedaan dibanding arsitektur *neural network* pada umumnya, dimana LSTM memiliki koneksi *feedback* sehingga LSTM memiliki kemampuan untuk memproses data secara berurutan seperti data suara dan video.

2.4 VGGNet

VGGNet merupakan sebuah model CNN yang dibuat oleh simonyan dan zisserman dari grup VGG (Visual Geometry Group) Universitas Oxford pada tahun 2014 [10]. Model ini merupakan juara kedua pada kompetisi ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2014 dalam bidang pengklasifikasian benda. Model ini dilatih menggunakan dataset ImageNet ILSVRC yang memiliki 1000 kelas benda dengan 1.3 Juta gambar pelatihan, 100.000 gambar pengujian, dan 50.000 gambar validasi [1].

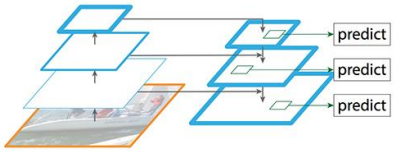
VGGNet memiliki 2 varian, yaitu VGG16 dan VGG19. VGG16 memiliki 16 *Weight Layer*, dan VGG19 memiliki 19 *Weight Layer*. VGGNet memiliki input berupa 224*224 pixel RGB. Data input kemudian diproses menggunakan susunan lapisan konvolusi dengan 5 lapisan *max pooling* diantara grup lapisan konvolusi dengan tujuan untuk melakukan *down-sampling* terhadap output setiap grup lapisan konvolusi. Struktur model ini dapat dilihat pada Gambar 1.



Gambar 1 Struktur model VGG16

2.5 *Feature Pyramid Nertworks*

FPN (*Feature Pyramid Networks*) merupakan pengekstrak fitur yang memiliki input berupa gambar dengan ukuran yang tidak ditentukan, dan output berupa peta fitur dengan ukuran yang proporsional pada tingkat yang berbeda-beda [4]. Proses ini menggunakan sebuah ConvNet sebagai *backbone model* yang menghasilkan hirarki fitur berisikan peta fitur dengan skala kelipatan 2 pada setiap lapisan prediksinya. Struktur model cnn ini dapat dilihat pada Gambar 3.



Gambar 2 Struktur Model FPN

2.6 *Smooth L1 Loss*

Smooth L1 Loss atau biasa disebut Huber Loss merupakan sebuah metode perhitungan *loss function* yang umumnya digunakan untuk permasalahan berjenis regresi. Fungsi ini menggunakan hasil kuadrat perbedaan hasil estimasi dengan hasil asli jika perbedaan absolut kedua nilai dibawah 1, dan menggunakan perbedaan absolut hasil estimasi dan hasil asli jika perbedaan memiliki nilai absolut diatas 1. Metode perhitungan ini digunakan untuk mengurangi tingkat sensitifitas *loss function* terhadap data *outlier [3]*.

$$\text{loss}(x,y) = \begin{cases} 0.5(x-y)^2, & if|x-y| < 1 \\ |x-y| - 0.5 & otherwise \end{cases} \quad (1)$$

Keterangan:
x = hasil prediksi
y = hasil asli

2.7 *Focal Loss*

Focal Loss merupakan sebuah metode perhitungan *loss function* yang umumnya digunakan untuk permasalahan berjenis pengklasifikasian. Fungsi ini menambahkan sebuah faktor modulasi pada fungsi *cross*

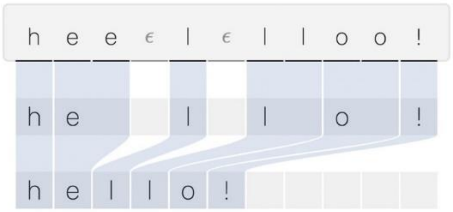
entropy loss dengan tujuan untuk mengurangi loss relatif dari data yang telah di klasfikasikan dengan baik [5].

$$\mathrm{Pt}(x,y) = \begin{cases} x, & if\ y = 1 \\ 1-x, & otherwise \end{cases} \quad (2)$$

$$\mathrm{FL(Pt)} = -(1-Pt)^{\gamma}\log(pt)$$

Keterangan:
x = hasil prediksi
y = hasil asli
γ = besaran faktor modulasi

2.8 *Connectionist Temporal Classification*



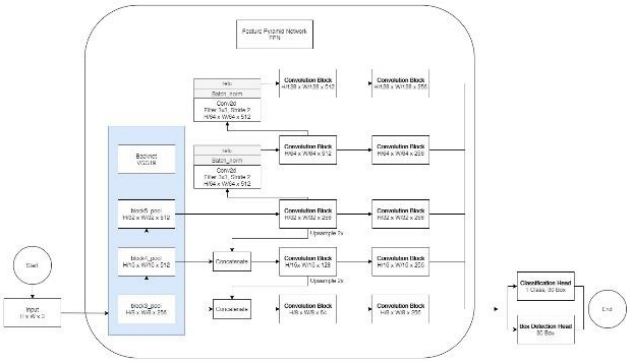
Gambar 3. Cara kerja CTC

Connectionist Temporal Classification (CTC) merupakan metode output dan penghitungan loss function dari neural network untuk menyelesaikan permasalahan dengan satuan waktu yang bervariasi seperti pengenalan tulisan tangan, pengenalan suara dan pengenalan teks [6].

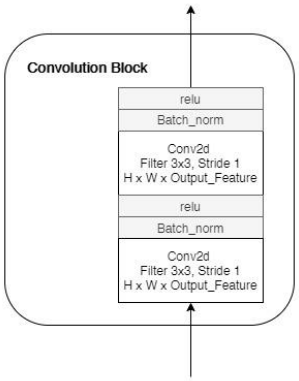
Untuk mendapatkan output dari proses CTC, untuk setiap satuan waktu dilakukan perhitungan probabilitas maksimal untuk setiap kelas dan hasil kelas yang berulang digabungkan menjadi satu. Perhitungan loss CTC didapatkan dengan menghitung probabilitas urutan yang berbeda pada setiap satuan waktu untuk kemudian ditotalkan.

$$p(Y|X) = \sum_{A \in A_{X,Y}} \prod_{t=1}^{T} pt(a_t|X) \quad (3)$$

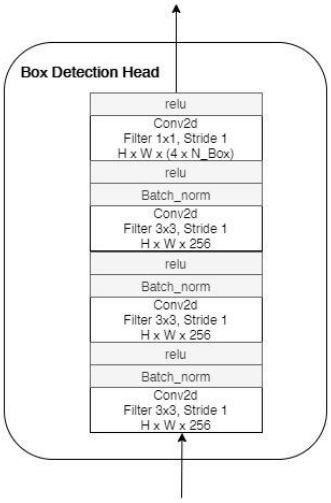
Keterangan:
$p(Y|X)$ = Besaran *loss* CTC
$pt(a_t|X)$ = Probabilitas urutan a pada satuan waktu t



Gambar 4 Diagram Model FPN



Gambar 5 Struktur *Convolution Block* pada FPN

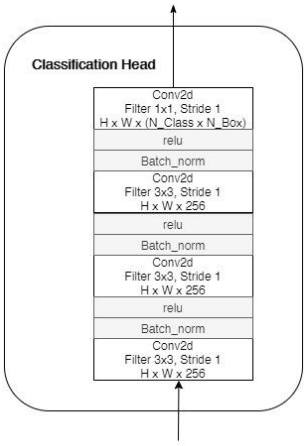


Gambar 6 Struktur *Box Detection Head* pada FPN

3. Hasil Percobaan

3.1 Hasil Pengujian Model Pendeteksian Teks

Model pendeteksian teks yang digunakan merupakan FPN dengan *backbone* VGG19. Model ini memiliki 5 output untuk setiap prediksi *bounding box* yang terdiri dari lokasi x, lokasi y, panjang, lebar, dan probabilitas bounding box tersebut merupakan teks. Struktur model ini dapat dilihat pada gambar 4



Gambar 7 Struktur *Classification Head* pada FPN

Pengujian model pendeteksian teks dilakukan untuk mengukur tingkat keakurasian model. Pengujian dilakukan dengan menggunakan data pelatihan dan pengujian pada dataset ICDAR 2015 *Text Localization* dan dataset COCO-Text. Hasil yang diukur pada model adalah *Focal Loss, Smooth L1 Loss, Precission, Recall* dan *F1-Score*. Tingkat akurasi model dihitung dengan menggunakan API pengevaluasian model ICDAR 2015 *Text Localization*.

Gambar 8 Hasil pengujian model pendeteksian teks

Model pendeteksian teks dilatih sebanyak 7 epoch dengan menggunakan gambar pelatihan dan pengujian pada dataset ICDAR 2015 Text Localization dan COCO-Text. Model diuji dengan menggunakan *batch size* 8 dengan *learning rate* 10^{-3} untuk 5 epoch pertama, dan 10^{-4} untuk 2 epoch berikutnya. hasil pengujian model dapat dilihat pada Tabel 12 Lampiran 9.

Tabel 1 Tabel hasil pengujian model pendeteksian teks

Epoch-ke	Training Loss	Validation Loss
1	7.9379	4.0541
2	3.6962	3.6846
3	3.4701	3.6536
4	3.2180	3.4996
5	3.1890	3.4733
6	3.0832	3.4266
7	2.9532	3.3665

Pada akhir pengujian didapatkan model pada epoch ke 7 dengan Total *Loss* sebesar 3.3665, *Precission* sebesar 44.2307%, *Recall* sebesar 55.3683% dan *F1-Score* sebesar 49.1768%. Dapat dilihat pada Tabel 13 Lampiran 9, bahwa model dapat melakukan pendeteksian teks dengan cukup baik pada teks dengan kemiringan 0° - 10°, dengan *Recall* sebesar 60.94%. Akurasi model terus berkurang semakin besar tingkat kemiringan pada teks. model tidak dapat mendeteksi teks dengan besar kemiringan diatas 60°.Tabel 2 Tabel akurasi model berdasarkan rotasi teks

Rotasi		Recall
0	180	55.37%
0	10	60.94%
10	20	48.59%
20	30	34.13%
30	40	12.50%
40	50	13.33%
50	60	25.00%
60	70	0.00%
70	80	0.00%
80	90	0.00%
90	180	0.00%

Dapat dilihat pada Tabel 14 Lampiran 9, bahwa model dapat melakukan pendeteksian teks dengan cukup baik pada teks yang memiliki ukuran 64x64 px dengan besar *Recall* sebesar 62.75%, tetapi model tidak dapat mendeteksi teks dengan ukuran dibawah 32x32px.

Tabel 3 Tabel akurasi model berdasarkan ukuran teks

Ukuran Teks	Recall
8x8 px	0.00%
16x16 px	0.00%
32x32 px	50.00%
64x64 px	62.75%
128x128 px	48.44%

Dapat dilihat pada Tabel 15 Lampiran 9, dampak *backbone* yang berbeda terhadap tingkat akurasi model. Dilakukan pengujian dengan menggunakan 4 *backbone* yaitu VGG16, VGG19, ResNet50, dan Resnet101. Berdasarkan hasil pengujian, dapat dilihat bahwa model menggunakan VGG19 memiliki tingkat *Precission* dan *F1-Score* tertinggi dengan nilai *Precission* 44.23%, dan *F1-Score* sebesar 49.18%. Model menggunakan backbone VGG16 memiliki nilai *Recall* tertinggi sebesar 58.98%.

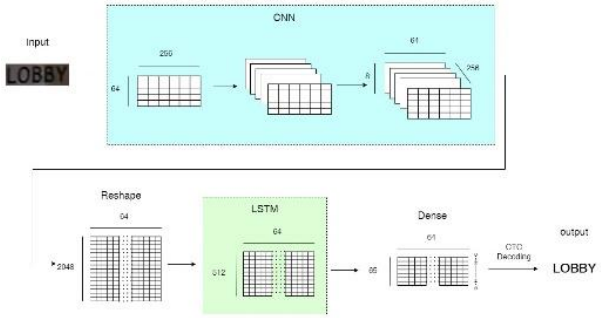
Tabel 4 Tabel akurasi model berdasarkan backbone model

Backbone	Precission	Recall	F1-Score
VGG19	**44.23%**	55.37%	**49.18%**
VGG16	27.43%	**58.98%**	37.45%
ResNet50	28.25%	52.82%	36.81%
ResNet101	32.81%	51.18%	39.98%

3.2 Hasil Pengujian Model Pengenalan Teks

Tabel 5 Struktur CNN-LSTM Pengenalan Teks

Nama Layer	Jenis Layer	Dimensi Output
Input	Input Layer	64 x 256 x 1
Convolution_1	Convolution_2d	64 x 256 x 32
Batch_Norm_1	Batch_Normalization	64 x 256 x 32
Relu_1	Activation Layer	64 x 256 x 32
Convolution_2	Convolution_2d	64 x 256 x 32
Batch_Norm_2	Batch_Normalization	64 x 256 x 32
Relu_2	Activation Layer	64 x 256 x 32
Max_Pool_1	Max_Pooling_2d	32 x 128 x 32
Convolution_3	Convolution_2d	32 x 128 x 64
Batch_Norm_3	Batch_Normalization	32 x 128 x 64
Relu_3	Activation Layer	32 x 128 x 64
Convolution_4	Convolution_2d	32 x 128 x 64
Batch_Norm_4	Batch_Normalization	32 x 128 x 64
Relu_4	Activation Layer	32 x 128 x 64
Max_Pool_2	Max_Pooling_2d	16 x 64 x 64
Convolution_5	Convolution_2d	16 x 64 x 128
Batch_Norm_5	Batch_Normalization	16 x 64 x 128
Relu_5	Activation Layer	16 x 64 x 128
Convolution_6	Convolution_2d	16 x 64 x 128
Batch_Norm_6	Batch_Normalization	16 x 64 x 128
Relu_6	Activation Layer	16 x 64 x 128
Max_Pool_3	Max_Pooling_2d	16 x 64 x 64
Convolution_7	Convolution_2d	8 x 64 x 256
Batch_Norm_7	Batch_Normalization	8 x 64 x 256
Relu_7	Activation Layer	8 x 64 x 256
Convolution_8	Convolution_2d	8 x 64 x 256
Batch_Norm_8	Batch_Normalization	8 x 64 x 256
Relu_8	Activation Layer	8 x 64 x 256
Reshape_1	Reshape	64 x 2048
Bidectional_1	Biderectional_LSTM	64 x 512
Bidectional_2	Biderectional_LSTM	64 x 512
Dense_out	Dense_Layer	64 x 65



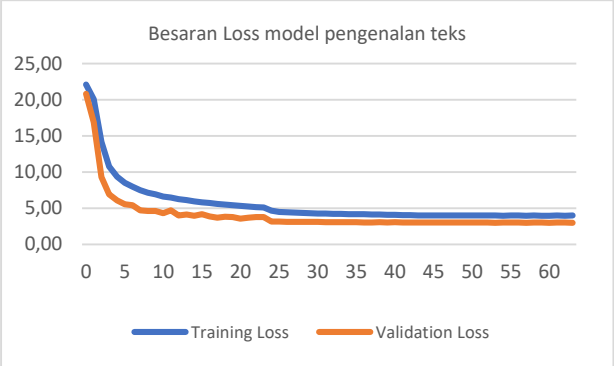
Gambar 9 Diagram model pengenalan teks

Model pengenalan teks yang digunakan merupakan Model CNN-LSTM. Model CNN yang digunakan memiliki 3 lapisan *pooling*, dan 8 lapisan konvolusi. Digunakan 2 lapisan LSTM *bidirectional* dengan lapisan dense sebagai output. Model ini memiliki 65 output yang terdiri dari 10 angka, 26 huruf kapital, 26 huruf kecil, 1 symbol, 1 karakter white space, dan 1 karakter kosong. Struktur model ini dapat dilihat pada Tabel 5.

Pengujian model pendeteksian teks dilakukan untuk mengukur tingkat keakurasian model. Pengujian dilakukan dengan menggunakan data pengujian dan pelatihan pada dataset ICDAR 2015 *Word Recognition* dan ICDAR 2019 *Cropped Word Script Identification*. Hasil yang diukur pada model adalah CTC *Loss*, *Total Edit Distance*, dan *Correctly Recognized Words*. Tingkat akurasi model dihitung dengan menggunakan API pengevaluasian model ICDAR 2015 *Word Recognition*

Gambar 10 Hasil pengenalan teks

Model pengenalan teks dilatih sebanyak 64 epoch dengan menggunakan gambar pelatihan dan pengujian pada dataset ICDAR 2015 *Word Recognition* dan ICDAR 2019 *Cropped Word Script Identification*. Model diuji dengan menggunakan *batch size* 64 dengan *learning rate* 10^{-3} yang akan dikurangi sebesar 0.1 apabila besaran loss validasi berhenti berkurang setelah 3 epoch. hasil pengujian model dapat dilihat pada Gambar 27 Lampiran 10 sampai dengan Gambar 30 Lampiran 10.



Gambar 11 Hasil pengujian model pengenalan teks

Pada akhir pengujian didapatkan model pada epoch ke 64 dengan besar CTC *Loss* sebesar 3.0046, *Total Edit Distance* sebesar 2145, dan *Correctly Recognized Words* sebesar 55.8016%.

4. Kesimpulan

Berdasarkan hasil dari pengujian yang telah dilakukan, didapatkan kesimpulan sebagai berikut:

1. Sistem CNN Pendeteksian teks dapat melakukan pendeteksian teks dengan cukup baik dengan nilai *Precission* sebesar 44.2307%, nilai *Recall* sebesar 55.3683%, dan *F1-Score* sebesar 49.1768% dalam melakukan pendeteksian teks pada gambar.
2. Model pengenalan teks dapat melakukan pengenalan teks dengan nilai *Total Edit Distance* yang buruk sebesar 2145, dan nilai *Correctly Recognized Words* yang cukup baik sebesar 55.8016% dalam melakukan pengenalan teks pada gambar.
3. CNN dapat digunakan untuk membangun sistem OCR yang dapat mengekstrak teks dalam gambar dengan cukup baik dengan menggunakan 2 model berbeda untuk mendeteksi dan mengenali teks.

REFERENSI

[1] Heravi, Einaz J.; and Aghdam, Hamed H. 2017, ”Guide to convolutional neural networks: a practical application to traffic-sign detection and classification”, Springer, Berlin.

[2] Hochreiter, Sepp; and Schmidhuber, Jürgen. “Neural Computation: Long short-term memory”, https://www.researchgate.net/publication/13853244_Long_Short-term_Memory, 30 November 2021

[3] Huber, Peter. ”Robust Estimation of a Location Parameter”. https://www.semanticscholar.org/paper/Robust-Estimation-of-a-Location-Parameter-Huber/e6bdbc325de48cbd24a04829f5ce33612513677f, 30 Agustus 2021

[4] Lin, Tsung-Yi; Goyal, Priya; Girshick, Ross; He, Kaiming; and Dollár, Piotr. 2017, ”Feature Pyramid Networks for Object Detection”, 2017 IEEE Conference on Computer Vision and Pattern Recognition

[5] Lin, Tsung-Yi; Goyal, Priya; Girshick, Ross B.; He, Kaiming; and Dollár, Piotr. 2017, ”Focal Loss for Dense Object Detection”, 2017 IEEE International Conference on Computer Vision

[6] Liwicki, Marcus; Graves, Alex; Bunke, Horst; and Schmidhuber, Jürgen. “A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks”, https://people.idsia.ch/~juergen/icdar_2007.pdf, 15 December 2021

[7] McCulloch, Warren; and Pitts, Walter. “A Logical Calculus of Ideas Immanent in Nervous Activity”. https://www.cs.cmu.edu/~./epxing/Class/10715/reading/McCulloch.and. Pitts.pdf, 27 Agustus 2021

[8] Russakovsky, Olga; Deng, Jia; Su, Hao; Krause, Jonathan; Satheesh, Sanjeev; Ma, Sean; Huang, Zhiheng; Karpathy, Andrej; Khosla, Aditya; Bernstein, Michael; C. Berg, Alexander; and Fei, Li Fei. “ImageNet Large Scale Visual Recognition Challenge 2014”, https://www.image-net.org/challenges/LSVRC/2014/index.php, 28 Agustus 2021

[9] Schantz, Herbert F. 1982, “The history of OCR, optical character recognition”, Recognition Technologies Users Association, Manchester

[10] Simonyan, Karen; and Zisserman, Andrew. “Very deep convolutional networks for large-scale image recognition”. https://arxiv.org/pdf/1409.1556.pdf, 28 Agustus 2021

[11] Valueva, M.V.; Nagornov, N.N.; Lyakhov, P.A.; Valuev, G.V.; and Chervyakov, N.I. “Application of the residue number system to reduce hardware costs of the convolutional neural network implementation”, https://www.sciencedirect.com/science/article/abs/pii/S0378475420301580, 27 Agustus 2021

[12] Venkatesan, Ragav; and Li, Baoxin. 2017, “Convolutional Neural Networks in Visual Computing: A Concise Guide”, CRC Press, Boca Raton

Ivan Wijaya, mahasiswa S1, program studi Teknik Informatika, Fakultas Teknologi Informasi Universitas tarumanagara.

Dra. Chairisni Lubis, M.Kom., memperoleh gelar Dra. Dari Matematika dan Ilmu Pengetahuan Alam Universitas Indonesia dan gelar M.Kom dari Universitas Indonesia. Saat ini sebagai Dosen program studi Teknik Informatika, Universitas Tarumanagara