[bagian header jangan di isi] "Judul", Sebatik, 00(0).
Submitted: 0000/00/00 Accepted: 0000/00/00 Published: 0000/00/00

Sebatik Vol. 28 No. 2 December 2024
ISSN: 1410-3737(p) 2621-069X(e)

DOI:10.46984/sebatik.v28i2.0000

Sistem Pengenalan Tulisan Tangan (Handwriting Recognition) menggunakan Convolutional Neural Network (CNN) dengan Python dan TensorFlow

***Andre Irawan Ameng [1)], Pitrasacha Adytia [2)],**dan **Ahmad Abul Khair [3)]**

[1,2,3]Teknik Informatika, STMIK Widya Cipta Dharma
[1,2,3]Jl. M. Yamin, Gn. Kelua, Kec. Samarinda Ulu, Kota Samarinda, Kalimantan Timur 75123
Email: 2043043@wicida.ac.id[1)], pitra@wicida.ac.id[2)], abul@wicida.ac.id[3)]

ABSTRAK

Penelitian ini bertujuan untuk mengembangkan sistem pengenalan tulisan tangan otomatis berbasis Convolutional Neural Network (CNN) guna membantu mahasiswa dalam memahami tulisan tangan dosen dengan lebih mudah. Sistem ini dirancang untuk mengenali tulisan tangan dalam bentuk huruf individu dan menyusunnya menjadi kata tunggal yang dapat dikonversi menjadi teks digital. Dataset yang digunakan terdiri dari huruf alfabet dengan berbagai variasi gaya tulisan untuk memastikan keandalan sistem dalam mengenali karakter yang berbeda. mengikuti kerangka kerja SKKNI Nomor 299 Tahun 2020, yang meliputi tahapan Business Understanding, Data Understanding, Data Preparation, Modeling, dan Model Evaluation. Proses preprocessing dilakukan menggunakan OpenCV untuk mendeteksi area karakter, sedangkan model CNN dilatih dengan memanfaatkan perangkat lunak seperti Jupyter Notebook (Python) dan Visual Studio Code

Kata Kunci: Pengenalan Tulisan Tangan, *Convolutional Neural Network* (CNN), Teknologi Informasi, Digitalisasi Tulisan.

Implementation Of Convolutional Neural Network Method for Vehicle License Plate Recognition

ABSTRACT

This research aims to develop an automatic handwriting recognition system based on Convolutional Neural Networks (CNN) to assist students in better understanding their professors' handwriting. The system is designed to recognize handwriting in the form of individual letters and arrange them into a single word that can be converted into digital text. The dataset used consists of alphabetic characters with various handwriting styles to ensure the system's reliability in recognizing different characters. The framework follows SKKNI Number 299 of 2020, which includes the stages of Business Understanding, Data Understanding, Data Preparation, Modeling, and Model Evaluation. The preprocessing process is carried out using OpenCV to detect character areas, while the CNN model is trained using software such as Jupyter Notebook (Python) and Visual Studio Code.

***Keywords**: Handwriting Recognition, Convolutional Neural Network (CNN), Information Technology, Handwriting Digitalization.*

1. PENDAHULUAN

Penelitian ini berfokus pada pengetesan sistem dengan menggunakan algoritma Convolutional Neural Network (CNN) untuk mengenali dan mengonversi tulisan tangan pada tahap hanya sebuah kata tunggal menjadi sebuah kata dalam bentuk digital. Dimana memahami sebuah tulisan tangan ini sering menjadi masalah bagi mahasiswa dalam memahami informasi yang didapat, yang dimana karena perbedaan gaya tulisan tangan sering kali menyebabkan kesulitan dalam membaca dan menginterpretasikan arahan pada informasi yang di dapatkan. Penelitian ini bertujuan untuk menguji efektivitas algoritma Convolutional Neural Network (CNN) dalam mengenali tulisan tangan yang dalam bentuk huruf default untuk pengetesan algoritma Convolutional Neural Network (CNN), Dataset yang digunakan dalam penelitian ini masih dalam bentuk alfabet. Hal ini dilakukan untuk mengevaluasi apakah CNN dapat berfungsi secara optimal dalam mengenali huruf individu sebelum diterapkan pada skenario yang lebih kompleks di masa depan. Fokus utama dari penelitian ini adalah mengembangkan sistem yang mampu mengenali huruf secara individu dan kemudian menyusunnya menjadi sebuah kata. Saat ini, sistem belum dapat mengenali keseluruhan kalimat dari sebuah gambar, dan hanya dapat mengenali sebuah huruf default dengan ketentuan tertentu, penelitian ini bukan hanya sekadar proyek

setengah jadi, tetapi merupakan langkah awal dalam mengembangkan solusi berbasis kecerdasan buatan yang dapat membantu mahasiswa memahami tulisan tangan dengan lebih efisien.

Teknologi ini diharapkan dapat berkontribusi dalam digitalisasi tulisan tangan, sehingga mahasiswa tidak perlu lagi mengalami kesulitan dalam membaca catatan atau instruksi yang diberikan. Selain itu, penelitian ini dapat menjadi referensi bagi peneliti lain yang ingin mengembangkan sistem serupa dengan cakupan yang lebih luas. Dengan adanya teknologi ini, komunikasi akademik dapat menjadi lebih efektif, mendukung proses pembelajaran, dan meningkatkan produktivitas di lingkungan pendidikan.

2. RUANG LINGKUP

Dalam penelitian ini permasalahan mencakup:

1. Cakupan permasalaahan
   Bagaimana efektivitas algoritma Convolutional Neural Network (CNN) dalam mengenali dan mengonversi tulisan tangan yang masih dalam bentuk huruf default menjadi teks digital, serta sejauh mana sistem ini dapat mengidentifikasi huruf individu dan menyusunnya menjadi kata sebelum dikembangkan lebih lanjut untuk mengenali kalimat secara utuh?.
2. Batasan-batasan penelitian
   Untuk menghindari meluasnya masalah yang akan dibahas dalam penelitian ini, maka penulis metetapkan batasan masalah yaitu sebagai berikut:
   1 Sistem lebih banyak fokus ke Bahasa Pemrograman Python.
   2 Fokus Penelitian pada huruf tanggal yang di gabung menjadi sebuah kata.
   3 Penggunaan dataset yang relevan dan representatif untuk melatih dan menguji.
   4 Memanfaatkan perangkat keras yang sudah tersedia, seperti laptop atau komputer pribadi, tanpa memerlukan akses ke infrastruktur komputasi yang mahal atau kompleks.
   5 Dataset yang di gunakan tidak yang rumit hanya menggunakan huruf tunggal dalam bentuk huruf default dari huruf A-Z yang pola penulisan yang bervariasi.
   6 Menggunakan library dan framework yang sudah tersedia dan didokumentasikan dengan baik, seperti TensorFlow memudahkan implementasi model CNN.
   7. Eksplorasi potensi pengenalan tulisan tangan dalam konteks nyata.
   8. Untuk implementasikannya menggunakan HTML ini tidak menggunakan realtime data atau databese.
3. Rencana hasil yang didapatkan
   Menguji efektivitas algoritma Convolutional Neural Network (CNN) dalam mengenali dan mengonversi tulisan tangan yang masih dalam bentuk huruf default menjadi kata tunggal dalam format digital, Dan Mengembangkan sistem pengenalan tulisan tangan yang dapat membantu mahasiswa memahami tulisan tangan dengan lebih mudah, sehingga mengurangi kesulitan dalam membaca dan menginterpretasikan informasi yang diberikan.

3. BAHAN DAN METODE

3.1 Machine Learning

Pada penelitian Sam'ani & Qamaruzzaman, 2017, Dengan judul Pengenalan Huruf Dan Angka Tulisan Tangan Mengunakan Metode Convolution Neural Network (CNN) metode yang digunakan ada 3 (tiga) Tahapan atau metode pada penelitian itu, yaitu yang pertama Metode Observasi , kedua Metode Kepustakaan, dan ke tiga Metode Pengembangan Sistem, dengan menggunakan CNN disini sebgai penghitungan.

Pada penelitaian ini ya itu yang penerapannya Convolution Neural Network (CNN) untuk Sistem Pengenalan Tulisan Tangan (Handwriting Recognition) yang dimana dalam penelitan dari 3 (tiga) penggunaan Convolution Neural Network (CNN) banyak menggunakan proses yang sama, yang membedakan jurnal dan penelitian ini adalah pada penelitian ini menggunakan python dan tensorflow berbasis android menggunakan metode CNN.

3.2 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah pengembangan dari Multilayer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis Deep Neural Network karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra Suartika dkk., 2016.

Maka perlu digunakan Convolutional Neural Network (CNN) untuk mengenali pola-pola yang terdapat pada tulisan tangan. Jaringan mengasumsikan bahwa input yang digunakan adalah data citra. Jaringan memiliki lapisan utama yang disebut lapisan convolution. Rahmawan dkk., 2023.

Pengenalan pola (pattern recognition). Pattern adalah entitas yang terdefinisi dan dapat diidentifikasi melalui ciri-cirinya dimana ciri – ciri tersebut digunakan untuk membedakan satu pattern dengan pattern lainnya. Pattern recognition adalah suatu teori, algoritma, sistem yang bertujuan menentukan kelompok atau kategori pattern berdasarkan ciri-ciri yang dimiliki oleh pattern tersebut. Dengan kata lain, patternrecognition membedakan suatu objek dengan objek lain Khunafa dkk., 2019.

Pengenalan tulisan tangan merupakan salah satu bentuk dari pengenalan pola, Dalam pembuatan sistem pengenalan pola berupa pengenalan huruf dan angka tulisan tangan penelitian ini menggunakan metode Convolution Neural Network (CNN). CNN adalah pengembangan dari Multilayer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi Sam'ani & Qamaruzzaman, 2017

Metode Convolutional Neural Network (CNN) dikenalkan sejak 1969 oleh Hubel (Neocognitron), lalu dilanjutkan oleh Yann LeCun. Metode CNN merupakan hasil dari pengembangan dari metode Neural Network.

CNN dapat digunakan untuk mendeteksi dan mengenali objek pada sebuah citra. CNN terdiri dari neuron yang memiliki bobot, bias dan fungsi aktivasi. Secara garis besar CNN tidak jauh berbeda dengan neural network lainnya Prihatiningsih dkk., 2019..

3.3 *Softmax Classifier*

Khunafa dkk., 2019, *Softmax Classifier* Fungsi *softmax classifier* adalah meghitung semua probabilitas masing-masing kelas target terhadap kemungkin seluruh kelas. Biasanya softmax digunakan untuk klasifikasi yang memiliki banyak kelas. Sehingga dengan fungsi ini dapat menghitung probabilitas dari masing-masing kelas dan menentukan kelas target data yang diinputkan berdasarkan nilai probabilitas yang tertinggi. Rentang nilai probabilitas dari softmax classifier sendiri adalah 0 hingga 1 dengan jumlah keseluruhan nilai probabilitas = 1. Adapun rumus dari softmax classifier

3.4 Layer Konvulusi

1. Input

   Adalah Input adalah data citra yang digunakan untuk pengenalan data atau data latih.
2. Convolutions

   Convolution adalah salah satu bagian dari proses extrak fitur, convolution dijalankan pada data input menggunakan sebuah filter atau kernel yang kemudian digunakan untuk memetakan fitur. Operasi konvolusi yaitu terjadi dengan menggeser filter di atas input. Di setiap lokasi, perkalian matriks dilakukan dan menjumlahkan hasilnya ke dalam peta fitur atau feature maps.
3. Subsampling

   Subsampling merupakan bagian kedua dari extrak fitur setelah convolution, penggunaan metode subsampling dalam sebagian besar cnn adalah *max* pooling atau mengambil nilai pixel terbesar di setiap pooling kernel. maka, dengan mengurangi jumlah parameter sekalipun, informasi terpenting dari pixel tersebut tetap diambil.
4. Fully connected layer

   Feature map yang dihasilkan dari feature extraction masih berbentuk multidimensional array dan harus dilakukan konversi data dari fitur map menjadi array 1 dimensi (*flatten*). *Fully connected layer* terdiri dari banyak neuron, di mana masing-masing neuron terhubung ke semua neuron di lapisan selanjutnya. Lapisan-lapisan di ujung jaringan ini digunakan untuk membuat prediksi oleh jaringan dan untuk memberikan hasil non-linier akhir dari fitur.

3.2 SKKNI Nomor 299 Tahun 2020

Menurut keputusan Menteri Republik Indonesia Nomor 299 Tahun 2020 tentang penetapan standar kompetensi kerja nasional indonesia kategori informasi dan komunikasi golongan pokok aktivitas pemrograman, konsultasi komputer dan kegiatan yang berhubungan dengan itu bidang keahlian Artificial Intelligence sub bidang Data Science.

TUJUAN UTAMA	**FUNGSI KUNCI**	**FUNGSI UTAMA**	**FUNGSI DASAR**
Menemukan pengetahuan, insight atau pola yang bermanfaat dari data untuk berbagai keperluan	Menganalisis kebutuhan organisasi	Business understanding	Menentukan objektif bisnis
			Menentukan tujuan teknis data science
			Membuat rencana proyek data science
		Data understanding	Mengumpulkan data
			Menelaah data
			Memvalidasi data
	Mengembangkan model	*Data preparation*	Memilah data
			Membersihkan data
			Menentukan label
			Mengintegrasikan data
		Modeling	Membangun skenario pengujian
			Membangun model
		Model evaluation	Mengevaluasi hasil pemodelan
			Melakukan review proses pemodelan
	Menggunakan model yang dihasilkan	*Deployment*	Membuat rencana deployment model
			Melakukan deployment model
			Membuat rencana pemeliharaan
			Melakukan pemeliharaan model
		Evaluation	Melakukan review proyek *data science*
			Membuat laporan akhir proyek data

4. PEMBAHASAN

Metode Pengembangan sistem yang digunakan pada penelitian ini ialah menggunakan metode SKKNI (Standar Kompetensi Kerja Nasional Indonesia) No. 299 tahun 2020, bidang keahlian Artificial Intelligence, sub bidang Data Science. Yang terdiri dari tujuh fungsi utama seperti :

1 business understanding
2 data understanding
3 data preparation
4 modeling
5 model evaluation
6 Deployment
7 Evaluation

4.1 Business understanding

Tahapan pemahaman bisnis bertujuan untuk mendefinisikan masalah bisnis yang akan diselesaikan serta tujuan dari proyek penelitian ini. Dalam penelitian ini masalah utama yang dihadapi adalah bagaimana mengimplementasikan metode Convolutional Neural Network (CNN) untuk mengenali plat nomor kendaraan tingkat akurasi yang tinggi. Sesuai dengan fungsi dasar dalam SKKNI Nomor 299 Tahun 2020, bagian ini mencakup tiga elemen utama: Menentukan objektif bisnis, Menentukan tujuan teknis data science, dan Membuat rencana proyek data science seperti :

1. Identifikasi Permasalahan
2. Perancangan Sistem
3. Membuat Rencana Proyek Data Science

4.2 Data Understanding

Berdasarkan fungsi dasar yang tercantum dalam SKKNI Nomor 299 Tahun 2020, tahap ini mencakup mengumpulkan data, menelaah data, dan memvalidasi data. Proses ini penting untuk memastikan bahwa dataset yang digunakan memiliki kualitas yang memadai untuk mendukung pengembangan model yang andal**.**

1. Mengumpulkan data

Dataset yang digunakan dalam penelitian ini diperoleh dari repositori GitHub, yang menyediakan total gambar sebanyak 316.572. Data berupa gambar berbagai Alphabet default. Pemilihan repositori ini dilakukan berdasarkan kualitas dan kelengkapan dataset, sehingga dapat mencerminkan kondisi nyata yang mungkin ditemui. Dataset yang diambil memiliki struktur yang memadai, dengan label yang terorganisir berdasarkan karakter, sehingga mempermudah proses pelatihan dan validasi model.

2. Menelaah Data Pada tahap ini, memastikan setiap karakter alphabet (huruf) memiliki representasi yang memadai untuk mendukung pelatihan model secara adil. Proses ini dilakukan dengan menganalisis distribusi data untuk mengetahui apakah jumlah sampel pada setiap kelas sudah seimbang.

3. Memvalidasi Data

Tahapan ini dilakukan untuk memastikan bahwa data yang digunakan bebas dari error dan duplikasi. Setiap gambar diperiksa untuk memastikan bahwa gambar dalam format yang konsisten, yaitu pada format .png. Skrip di bawah ini berfungsi untuk mengonversi file gambar dengan format tertentu (seperti .jpg, .jpeg, .bmp, .tiff, dan .gif) menjadi format PNG di dalam direktori tertentu, termasuk subdirektorinya

4.3 Data Preparation

Pada tahap data preparation, dilakukan berbagai proses untuk memastikan dataset siap digunakan dalam pelatihan model. Langkah pertama adalah melakukan *import library* yang di butuhkan kemudian melakukan upload data gambar yang sudah disiapkan. Selanjutnya standarisasi ukuran gambar, di mana semua gambar diubah menjadi dimensi yang seragam agar sesuai dengan input model *Convolutional Neural Network (CNN)*.

1. Import library

```
from tensorflow.keras import layers, models, callbacks
import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img, img_to_array


from keras.metrics import sparse_categorical_crossentropy


import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix


import os

import pandas as pd
```

Pada skrip diatas merupakan skrip untuk memanggil library tensorflow yang memiliki berbagai macam fungsi untuk melakukan pengolahan gambar seperti,aumentasi data , melakukan *preproccessing*, membuat model *Convolutional Neural Network(CNN)* hingga *multi layer perceptron* (jaringan saraf tiruan) ,hingga dapat melakukan penyimpanan model dengan format file H5 dan Keras .

2. Upload Data

```
PATH = 'D:/file skripsi/dataset/class/processed_images'
```

Pada skrip diatas merupakan tampilan skrip untuk mengupload data dari folder path yang ada di windows explore dan di masukan kedalam variabel agar memudahkan dalam melakukan pemanggilan data pada saat melakukan augmentasi dan preprocessing

3.Prepocessing

```
BATCH_SIZE = 56
IMG_SIZE = (150, 150)
```

Pada skrip diatas merupakan skrip yang digunakan untuk meyiapkan parameter yang akan berfungsi untuk mengubah ukuran gambar dengan lebar 150 pixel dan tinggi 150 pixel (150x150)sehingga mempermudah dalam persiapan data training dan validation serta jumlah

Sebatik Vol. 28 No. 2 December 2024
ISSN: 1410-3737(p) 2621-069X(e)

jurnal.wicida.ac.id/index.php/sebatik

batch yang merupakan jumlah gambar yang akan di input pada model pada saat training

4.Prepocessing

```
# Membuat generator pelatihan dengan normalisasi
train_datagen = ImageDataGenerator(
    rescale=1./255,  # Normalisasi pixel (0-1)
    rotation_range=180,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Pada Skripdiats sebuah objek yang bertugas untuk mengubah dan membagi data gambar untuk melatih model Convulutional Neural Network. Pertama, Gambar secara default memiliki nilai piksel dalam rentang [0, 255]. Dengan menggunakan rescale = 1./255, setiap nilai piksel akan dinormalisasi menjadi rentang [0, 1]. Langkah ini penting untuk membantu proses pembelajaran model menjadi lebih stabil dan konvergen lebih cepat. Kemudian menerapkan transformasi rotation range atau putaran pada gambar dengan intensitas tertentu. Nilai 180 menandakan tingkat pemutaran gambar yang akan diterapkan secara acak. Hasilnya, beberapa gambar pada saat pelatihan akan berputar/berbalik arah, yang membantu model menjadi lebih robust terhadap variasi rotasi atau kemiringan objek pada gambar asli. Serta melakukan zoom in atao zoom out secara acak dengan kisaran 20%. Dengan adanya zoom acak, model akan mengenali terhadap perubahan jarak objek dari kamera atau variasi ukuran objek.selain itu data dibagi menjadi 2 bagian yaitu 70% untuk melatih model dan 15% untuk menguji kinerja model dan 15% untuk validasi. Semua langkah ini bertujuan untuk meningkatkan kemampuan model dalam mengenali pola pada gambar yang bervariasi dan mengurangi kemungkinan model terkaky bergantung pada data pelatihan tertentu

4.4Permodelan

```
# Membuat generator train
train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=IMG_SIZE,
    batch_size=BATCH_SIZE,
    color_mode='grayscale',
    class_mode='sparse'
)

# Membuat generator validasi dengan normalisasi
validation_datagen = ImageDataGenerator(rescale=1./255)

validation_generator = validation_datagen.flow_from_directory(
    VALIDATION_DIR,
    target_size=IMG_SIZE,
    batch_size=BATCH_SIZE,
    color_mode='grayscale',
    class_mode='sparse'
)
```

Skrip diatas bertujuan untuk mempersiapkan data yang akan dipakai dalam pembuatan model Convolutional Neural Network (CNN). Skrip ini membagi data menjadi dua bagian: data pelatihan (train) dan data validasi (validation). Pembagian ini penting agar model CNN dapat belajar dari data pelatihan dan kemudian diuji pada data validasi untuk melihat seberapa baik model tersebut bekerja pada data yang belum pernah dilihat sebelumnya

```
# Layer CNN tanpa Lambda
input_layer = layers.Input(shape=(150, 150, 1))
x = layers.Conv2D(32, (3, 3), padding='valid', activation='relu')(input_layer)
x = layers.MaxPooling2D((2, 2), strides=(2, 2), padding='valid')(x)
x = layers.Conv2D(64, (3, 3), padding='valid', activation='relu')(x)
x = layers.MaxPooling2D((2, 2), strides=(2, 2), padding='valid')(x)
x = layers.Conv2D(128, (3, 3), padding='valid', activation='relu')(x)
x = layers.MaxPooling2D((2, 2), strides=(2, 2), padding='valid')(x)
x = layers.Conv2D(256, (3, 3), padding='valid', activation='relu')(x)
x = layers.MaxPooling2D((2, 2), strides=(2, 2), padding='valid')(x)

# Meratakan hasil ekstraksi fitur
x = layers.Flatten()(x)
x = layers.Dense(512, activation='relu')(x)
```

Pada skrip diatas menjelaskan proses pembuatan model CNN. Alur ini dimulai pada lapisan pertama dengan pengambilan input gambar berukuran 150x150 pixel dengan 1 chanell warna, kemudian melakukan operasi konvolusi pada setiap pixel yang dilewati oleh kernel berukuran 3x3, kemudian dilakukan fungsi aktivasi Relu(Retified Linear Unit),selanjutnya hasilnya diteruskan ini diteruskan ke lapisan pooling. Gambar tersebut kemudian melalui beberapa lapisan kovolusi dan pooling beberapa kali secarberulang. Setelah itu , hasilnya di ubah menjadi matrix 1 dimensi menggunakan fungsi Flatten() dan diteruskan ke lapisan fully connecteed untuk menghasil kan prediksi di lapisan terakhir

```
# Kompilasi model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

Pada skrip diatas merupakan tahap berikutnya setelah pembuatan model CNN adalah melakukan konfigurasi untuk pelatihan dan pengujian model dengan menggunakan skrip model. compile (optimizer=’adam’,loss=’sparse_categorical_crossent ropy’, metrics= [‘accuracy’]). Proses ini mencakup tiga komponen utama:

1.Optimizer

Optimizer ‘Adam’: secara otomatis menyesuaikan laju pembelajaran setiap parameter, memadukan keunggulan metode AdaGrad dan RMSProp, sehingga proses pelatihan model menjadi lebih cepat, stabil, dan umumnya cocok untuk berbagai jenis masalah tanpa memerlukan banyak penyesuaian hyperparameter.

2.Fungsi Kerugian (loss function)

Fungsi kerugian sparse categorical crossentropy: Digunakan untuk
masalah klasifikasi dengan lebih dari dua kelas. Ini mengukur seberapa
baik prediksi model dibandingkan dengan label sebenarnya dan
memberikan umpan balik untuk mengurangi kesalahan.

3.Metrik Akurasi

Metrik akurasi: Menunjukkan persentase keakuratan model dalam melakukan prediksi dengan benar kemudian dibandingkan dengan total prediksi, membantu mengevaluasi kinerja model selama pelatihan dan pengujian.

```
# Pelatihan model
history = model.fit(train_generator,
                    steps_per_epoch=train_generator.n // train_generator.batch_size,
                    epochs=40,
                    validation_data=validation_generator,
                    validation_steps=validation_generator.n // validation_generator.batch_size,
                    callbacks=[model_save_callback]
                    )
```

Pada penelitian ini, model deep learning dilatih menggunakan data yang disediakan oleh generator pelatihan dan validasi. Proses pelatihan ini dilakukan dengan menggunakan metode model.fit() dari Keras/TensorFlow, yang secara umum berfungsi untuk melatih model berdasarkan data yang diberikan, baik data pelatihan maupun validasi.

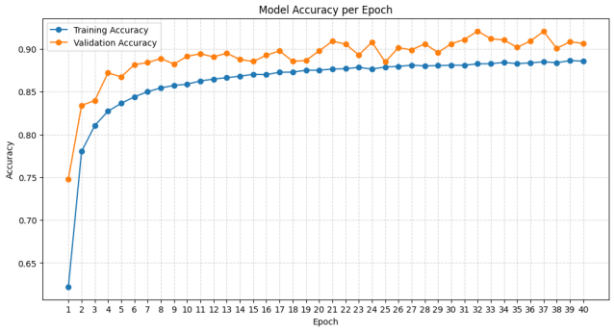
Untuk menangani dataset yang besar dan tidak dapat dimuat sepenuhnya ke dalam memori, digunakanlah *data generator* (train_generator dan validation_generator). Generator ini bertugas untuk memberikan data dalam bentuk batch secara bertahap, memungkinkan pemrosesan data yang lebih efisien tanpa membebani memori. Generator train_generator menyediakan data pelatihan, sedangkan validation_generator menyediakan data validasi untuk evaluasi model setelah setiap epoch. Proses pelatihan dilakukan dalam beberapa iterasi yang disebut sebagai *epoch*. Setiap epoch terdiri dari sejumlah langkah yang disebut *steps*, di mana setiap langkah mengacu pada pemrosesan satu batch data oleh model. Parameter steps_per_epoch menentukan jumlah langkah yang dilakukan dalam satu epoch. Pada kasus ini, jumlah langkah per epoch dihitung dengan membagi total sampel dalam data pelatihan (train_generator.n) dengan ukuran batch (train_generator.batch_size). Hal yang sama diterapkan pada data validasi dengan parameter validation_steps, yang dihitung dengan cara yang serupa menggunakan validation_generator.n dan validation_generator.batch_size.

Pelatihan model ini dilakukan selama 40 epoch, yang ditentukan oleh parameter epochs=40. Selama setiap epoch, model dilatih dengan data pelatihan, kemudian diuji menggunakan data validasi. Tujuan dari penggunaan lebih dari satu epoch adalah untuk memastikan model belajar dari data pelatihan dan meningkatkan kinerjanya dalam menggeneralisasi data yang tidak terlihat sebelumnya, yang diukur melalui data validasi.

Selama pelatihan, digunakan callback untuk menyimpan model terbaik yang dihasilkan pada setiap epoch. Callback ini berupa objek model_save_callback, yang merupakan implementasi dari fungsi ModelCheckpoint di Keras. Callback ini akan menyimpan model pada setiap langkah pelatihan berdasarkan kriteria tertentu, seperti akurasi atau loss yang lebih rendah pada data validasi. Hal ini penting untuk memastikan bahwa model yang disimpan adalah model terbaik yang dapat diandalkan, dan memungkinkan untuk melanjutkan pelatihan di masa mendatang jika diperlukan. Dengan menggunakan callback, proses pelatihan menjadi lebih fleksibel dan memungkinkan pengguna untuk memperoleh model yang optimal tanpa harus menyelesaikan seluruh pelatihan.

Fungsi model.fit() mengembalikan objek history, yang berisi informasi tentang metrik pelatihan selama setiap epoch, termasuk nilai loss dan akurasi untuk data pelatihan dan validasi. Objek ini sangat berguna untuk analisis lebih lanjut, seperti menggambar grafik yang menunjukkan perubahan kinerja model sepanjang pelatihan. Misalnya, grafik loss dan akurasi dapat digunakan untuk mengevaluasi apakah model mengalami overfitting atau underfitting, serta untuk memahami sejauh mana model belajar dari data.

4.5 Model Evaluation



Plot di atas menunjukkan akurasi model selama proses pelatihan dan validasi pada setiap epoch. Sumbu horizontal (x-axis) merepresentasikan jumlah epoch atau iterasi pelatihan, sedangkan sumbu vertikal (y-axis) menunjukkan nilai akurasi model. Observasi: Training Accuracy (Garis Biru): Akurasi pelatihan meningkat secara signifikan pada awal epoch, dari sekitar 0.65 hingga mendekati 0.88 pada epoch ke-10. Setelah itu, peningkatan akurasi cenderung melambat dan akhirnya stabil mendekati 0.89 hingga akhir pelatihan (epoch ke-40). Garis ini menunjukkan bahwa model belajar dengan baik dari data pelatihan dan mencapai titik stabil.

Validation Accuracy (Garis Orange): Akurasi validasi juga mengalami peningkatan cepat di awal epoch, naik hingga sekitar 0.90 pada epoch ke-10. Setelah itu, nilai akurasi validasi berfluktuasi di sekitar 0.90–0.91, menunjukkan bahwa performa model pada data validasi cukup stabil. Namun, garis ini memiliki sedikit fluktuasi dibandingkan dengan garis pelatihan, yang mungkin disebabkan oleh variasi dalam data validasi.

Stabilitas Model: Akurasi pelatihan dan validasi menunjukkan perbedaan yang kecil, yang mengindikasikan bahwa model tidak mengalami overfitting. Konvergensi: Model telah mencapai titik stabil di sekitar epoch ke-30 hingga ke-40, di mana akurasi pelatihan dan validasi tidak lagi mengalami peningkatan yang signifikan.

Performa: Nilai akurasi validasi yang tinggi (sekitar 90%) menunjukkan bahwa model memiliki performa

yang baik dalam mengenali data yang belum dilihat sebelumnya. Kesimpulan: Plot ini menunjukkan bahwa model Convolutional Neural Network (CNN) berhasil belajar dengan baik dari data pelatihan dan memberikan hasil akurasi yang stabil pada data validasi, dengan perbedaan kecil antara akurasi pelatihan dan validasi. Dapat dilihat Hal ini menunjukkan bahwa model sudah cukup optimal dalam mengenali pola dari data tulisan tangan.

4.6 Deployment

Tahapan ini melibatkan implementasi model yang telah dievaluasi ke dalam sebuah prototype web. Pengujian dalam kondisi nyata, dan pemantauan kinerja model setelah implementasi. Dalam penelitian ini, deployment dimulai dengan mengintegrasikan model pada sebuah web prototype. Langkah ini diikuti oleh uji coba dalam lingkungan nyata untuk memastikan bahwa model bekerja dengan baik dalam kondisi operasional sebenarnya. Pengujian ini penting untuk memverifikasi bahwa model dapat mengenali tulisan tangan dengan akurasi yang tinggi. Setelah implementasi, kinerja model dipantau secara terus-menerus untuk mengidentifikasi dan memperbaiki masalah yang mungkin timbul, serta untuk memastikan bahwa model tetap berfungsi secara optimal. digunakan untuk melakukan penyesuaian dan peningkatan lebih lanjut pada model dan aplikasi

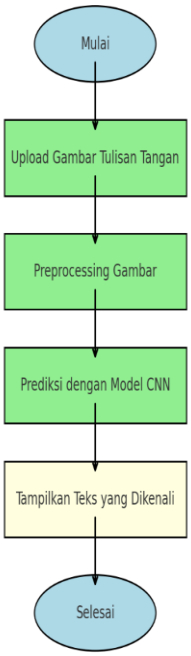
4.6 Preoses Deployment

Proses deployment dalam penelitian ini dirancang untuk memastikan alur sistem pengenalan tulisan tangan berjalan secara efisien dari awal hingga akhir. Sistem ini berbasis aplikasi web yang memungkinkan pengguna untuk mengunggah gambar, memprosesnya melalui backend, dan mendapatkan hasil prediksi berupa karakter alphabet. Backend sistem menggunakan metode Convolutional Neural Network (CNN) untuk melakukan prediksi, sedangkan preprocessing citra dilakukan menggunakan OpenCV untuk mengekstrak karakter dari gambar. Antarmuka web dibangun menggunakan framework Flask, yang memfasilitasi komunikasi antara frontend dan backend.

Setelah gambar diunggah oleh pengguna, backend memproses citra dengan langkah-langkah seperti konversi warna, thresholding, dan ekstraksi kontur untuk mendeteksi Region of Interest (ROI), yaitu area yang berisi tulisan tangan. ROI ini kemudian diteruskan ke model CNN yang telah dilatih sebelumnya untuk menghasilkan prediksi karakter. Hasil prediksi berupa kata dikirim kembali ke frontend dan ditampilkan kepada pengguna melalui antarmuka yang sederhana dan interaktif. Proses deployment ini dirancang untuk memberikan pengalaman pengguna yang intuitif serta memanfaatkan teknologi machine learning dan pengolahan citra untuk menghasilkan sistem yang akurat dan andal.

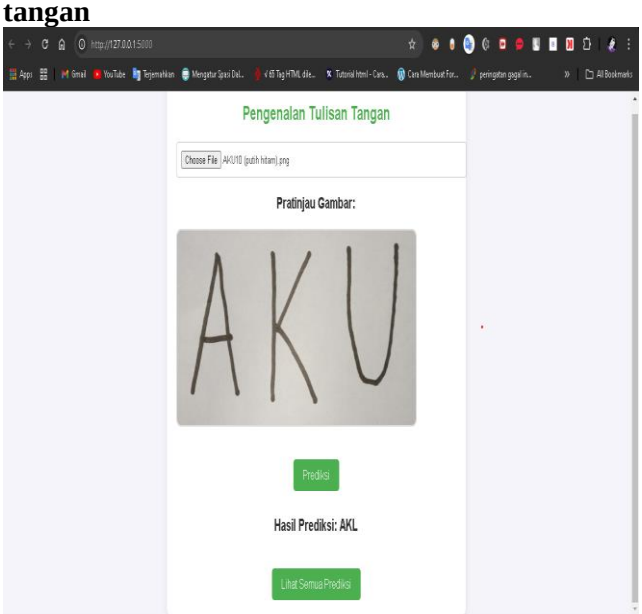
Sistem deployment dalam penelitian ini tidak bergantung pada data real-time, melainkan menggunakan dataset statis yang telah dilatih sebelumnya pada model CNN. Hal ini memungkinkan proses prediksi berjalan secara cepat dan efisien karena model hanya memerlukan pemrosesan citra yang telah ditentukan, tanpa memerlukan pembaruan atau penyesuaian model secara berkala.

4.6.1 Flowchart pengenalan plat kendaraan



Gambar 4. 1 Flowchart pengenalan plat nomor kendaraan

4.6.2 Tampilan halaman pengenalan tulisan tangan



Gambar 4. 2 Tampilan awal aplikasi pengenalan plat nomor kendaraan

Pada gambar 4.4 merupakan tampilan awal website, halaman utama menampilkan judul "Pengenalan tulisan tangan", yang memberikan informasi kepada pengguna mengenai fungsi utama aplikasi ini.

4.6Evaluation

Tahap selanjutnya ialah implementasi atau pengujian dimana sistem pengenalan plat akan diuji menggunakan metode pengujian sistem yaitu black-box testing dan white box testing yang berfungsi untuk melihat kesalahan sistem prototype ini dapat berjalan dengan lancar.

4.7.1 Black-Box Testing

Pada tahap ini, dilakukan pengujian sistem secara menyeluruh untuk memastikan bahwa sistem berfungsi dengan baik dan memenuhi seluruh persyaratan yang telah ditentukan. Pengujian bertujuan untuk mengevaluasi apakah prototipe yang dikembangkan mampu beroperasi sesuai dengan fungsi yang dirancang dan dapat diandalkan dalam berbagai kondisi erasional. Setiap komponen dan fitur sistem diuji secara komprehensif untuk memastikan bahwa tidak ada kesalahan atau kekurangan yang terlewatkan. Melalui pengujian yang mendetail, masalah yang muncul dapat diidentifikasi dan diperbaiki sebelum sistem diluncurkan secara penuh. Evaluasi ini juga mencakup pengujian kinerja sistem dalam berbagai skenario penggunaan untuk memastikan bahwa sistem mampu menangani kondisi nyata yang beragam dan memenuhi harapan pengguna

No	Pertanyaan	Kesesuaian	
		Sesuai	Tidak Sesuai
1	Apakah tombol memilih gambar dari penyimpanan perangkat berhasil?	Ya	
2	Apakah menampilkan gambar yang di pilih di web?	Ya	
3	Apakah menampilkan hasil prediksi dalam bentuk kata/huruf?	Ya	
4	Apakah tombol melihat semua prediksi berhasil berpindah halaman?	Ya	
5	Apakah tombol clear dan kembali pada halaman melihat semua prediksi berhasil?	Ya	

4.7.2 white-box testing

Pengujian white-box testing ini dilakukan dengan cara melihat ke dalam modul untuk meneliti kode-kode atau program yang ada, dengan tujuan untuk menganalisis apakah terdapat kesalahan dalam implementasi program. Dalam hal ini, pengujian bertujuan untuk memastikan bahwa setiap komponen dari sistem berjalan sesuai dengan spesifikasi yang telah ditentukan.

5. KESIMPULAN

Penelitian ini berhasil mendapatkan hasil yang di inginkan dengan menggunakan algoritma CNN diaman dataset yang digunakan yang hanya berupa huruf tunggal dapat di kenali dan di pahami, maka dari itu kesimpulan dari penggunaan algoritma CNN pada penelitian ini dapat di gunakan untuk mengenal sebuah tulisan tangan dimana dapat mengenali sebuah huruf dengan cukup baik, tetapi juga memiliki beberapa masalah diamana dataset yang tidak seimbang dan kurang bervariasi mulai dari gaya tulisan, latar belakangnya, pencahayaan, dan juga dimana dataset tersbut di buat seperti kertas dapat mempengaruhi akurasi/ketepatan dalam mengenal sebuah huruf atau khususnya kata yang telah di gabungkan dari huruf tunggal agar menjadi sebuah kata dalam penelitian ini, walaupun hasil akurasi dari penelitian ini memberikan akurasi sebesar 89% dalam mengenali huruf tunggal sebgai datasetnya masih tidak terlalu dapat megnelani beberapa tulisan tangan dari dataset uji yang di gunakan.

Dengan demikian, pada penelitian ini dapat memberikan kontribusi dalam mempermudah proses digitalisasi tulisan tangan yang sebelumnya hanya dapat mengenali sebuah kata kedepannya dapat di kembangkan menjadi dapat mengenali kalimat yang panjang dan meningkatkan aksesibilitas informasi bagi mahasiswa. Namun, masih terdapat ruang untuk pengembangan lebih lanjut, seperti penambahan dataset dengan variasi tulisan yang lebih luas dan integrasi sistem dengan aplikasi berbasis Android untuk mempermudah penggunaannya

6. SARAN

Penelitian ini masih memiliki peluang untuk dikembangkan lebih lanjut, terutama dalam meningkatkan akurasi dan keandalannya. Diharapkan hasil penelitian ini dapat menjadi pijakan untuk pengembangan teknologi serupa di masa mendatang meliputi:

1.Peningkatan Akurasi Model: Meskipun model ini sudah memberikan hasil yang baik, akurasi model masih dapat ditingkatkan lebih lanjut dengan memperbanyak dataset, terutama untuk huruf-huruf yang jarang ditemukan atau memiliki variasi gaya tulisan yang lebih

kompleks. Selain itu, pemilihan algoritma atau teknik pemodelan lain seperti Transfer Learning juga dapat dipertimbangkan untuk meningkatkan performa model.

2. Pengujian dengan Data yang Lebih Variatif: Pengujian model menggunakan data tulisan tangan dari berbagai penulis dengan latar belakang dan gaya tulisan yang berbeda dapat membantu mengevaluasi kekuatan dan kelemahan sistem. Dengan lebih banyak variasi dalam data, model akan lebih robust dan efektif dalam mengenali tulisan tangan dari berbagai individu.

3. Penerapan pada Konteks yang Lebih Luas: Penelitian ini berfokus pada tulisan dengan gaya tulisan tangan lokal. Oleh karena itu, perlu dilakukan penelitian lebih lanjut dengan melibatkan lebih banyak subjek dari berbagai latar belakang untuk menguji kemampuan model dalam mengenali tulisan tangan di konteks yang lebih luas, seperti tulisan tangan dari pelajar atau pekerja.

4. Pengembangan Fitur Tambahan: Fitur tambahan, seperti kemampuan untuk mengonversi tulisan tangan yang lebih kompleks (misalnya, tanda tangan atau kaligrafi) atau mendeteksi kesalahan penulisan dalam teks yang dihasilkan, dapat menjadi fokus penelitian selanjutnya untuk memperluas kegunaan aplikasi/web site ini di berbagai bidang.

7. DAFTAR PUSTAKA

Palangkaraya, M. H. Q. S. (2017). Pengenalan Huruf Dan Angka Tulisan Tangan Mengunakan Metode Convolution Neural Network (CNN). *Speed-Sentra Penelitian Engineering dan Edukasi*, *9*(2).

Putra, W. S. E. (2016). Klasifikasi citra menggunakan convolutional neural network (CNN) pada caltech 101. *Jurnal Teknik ITS*, *5*(1).

Rahmawan, F., Habibi, R., & Setyawan, M. Y. H. (2023). Rekognisi Huruf Tulisan Tangan Menggunakan *Convolutional Neural Network*. *Jurnal Sistem Cerdas*, *6*(3), 262-276.

Asmara, R. A., Syulistyo, A. R., & Qudsi, N. K. (2019). Identifikasi Citra Tulisan Tangan Digital Menggunakan Convolutional Neural Network (CNN). In *Seminar Informatika Aplikatif Polinema* (pp. 48-53).

Prihatiningsih, S., Andriani, F., & Nugraha, N. (2019). Analisa Performa Pengenalan Tulisan Tangan Angka Berdasarkan Jumlah Iterasi Menggunakan Metode Convolutional Neural Network. *Jurnal Ilmiah Teknologi dan Rekayasa*, *24*(1), 58-66.

.