

## BAB II

### TINJAUAN PUSTAKA DASAR TEORI

#### 2.1 Tinjauan Pustaka

Penelitian yang berjudul “Pengenal Plat Kendaraan Berbasis Android menggunakan Viola Jones dan *Kohonen Neural Network*” milik Aryo Michael membahas tentang membangun sistem pengenalan plat kendaraan roda dua berbasis *android* dengan metode *viola jones*, kemudian segmentasi karakter plat menggunakan metode morfologi, dan pengenalan karakter plat dengan *kohonen neural network*. Hasil penelitian yang diperoleh menunjukkan bahwa pengenalan plat kendaraan menggunakan *kohonen neural network* berdasarkan pengujian yang dilakukan menunjukkan persentase keberhasilan pengenalan karakter pada plat kendaraan bermotor pada kondisi yang baik sebesar 78,57% sedangkan pada plat yang kurang baik sebesar 57,14% (Michael, 2019).

Penelitian yang berjudul “Deteksi Dan Pengenalan Plat Karakter Nomor Kendaraan Menggunakan *OpenCV* Dan *Deep Learning* Berbasis Python” milik Mochammad Zakiyamani membahas tentang sistem pengenalan karakter pada plat nomor kendaraan Indonesia menggunakan metode *Convolutional Neural Network* (CNN) untuk mengenali objek pada gambar selayaknya manusia dengan pembelajaran pada sebuah komputer dengan menggunakan jaringan saraf tiruan. Pengenalan karakter plat nomor kendaraan Indonesia merupakan salah satu jenis *deep learning* karena dapat mengenali berbagai karakter huruf dan angka. Hasil penelitian yang diperoleh dengan data yang digunakan sebanyak 40 citra mobil dan 36 kelas karakter yang terdiri dari huruf dan angka, pengujian plat kendaraan dengan metode CNN dengan tingkat akurasi mencapai 96% dengan tingkat kesalahan 11,78% (Zakiyamani, 2022).

Penelitian yang berjudul “Perbandingan Akurasi Pengenalan Karakter Plat Nomor Menggunakan *Tesseract* Dan Data Latih *Emnist*” milik Trisiwi Indra Cahyani membahas tentang pengenalan karakter dapat menggunakan *Optical Character Recognition* (OCR) yang melakukan metode *template matching* pada huruf dan angka. Menggunakan *Convolutional Neural Network* dengan melatih data *Emnist* untuk melakukan pengenalan karakter, sebagai perbandingan penggunaan

metode OCR menggunakan *Tesseract* dan CNN dalam melakukan pengenalan karakter. Dengan data yang diuji sebanyak 58 citra mobil dengan 36 kelas karakter yang terdiri dari huruf dan angka. Pengujian pengenalan karakter menggunakan CNN pada data latih Emnist menghasilkan kinerja yang kurang baik dengan 11 citra memiliki akurasi diatas 75%. Penelitian ini menghasilkan pengenalan karakter terbaik pada *Tesseract*-OCR menggunakan segmentasi karakter pada plat nomor dengan 44 citra memiliki akurasi diatas 75% (Cahyani, 2022).

Penelitian yang berjudul “Pengenalan Pola Karakter Plat Nomor Kendaraan Menggunakan Algoritma Momentum *Backpropagation Neural Network*” milik Donny Avianto membahas tentang pengenalan plat nomor menggunakan metode Momentum *Backpropagation Neural Network* untuk mengenali karakter dari suatu citra plat nomor kendaraan di Indonesia. Citra plat nomor akan diubah menjadi citra biner. Citra biner kemudian disegmentasi untuk mengisolasi karakter-karakter yang akan dikenali. Terakhir dimensi citra hasil segmentasi akan direduksi menggunakan *Haar Wavelet*. Uji coba pada penelitian kali ini melibatkan 276 karakter yang terdiri dari huruf dan angka pada plat nomor kendaraan di Indonesia. Hasil uji coba menunjukkan 268 karakter diantaranya mampu dikenali dengan benar. Dengan kata lain metode yang digunakan memiliki tingkat akurasi hingga 97,10% (Avianto, 2016).

Penelitian yang berjudul “Identifikasi Pengenalan Karakter Plat Nomor Kendaraan Menggunakan Jaringan Syaraf Tiruan Berbasis Citra Digital” milik Sufiatul Maryana membahas tentang mengidentifikasi pengenalan karakter plat nomor kendaraan menggunakan Jaringan Syaraf tiruan berbasis citra digital, pengenalan karakter dan keberhasilan proses klasifikasi ditentukan dari keberhasilan ekstraksi fitur tiap karakter sehingga dengan bentuk karakter yang beragam akan menambah tingkat kesulitan dalam proses pengenalan. Pengenalan pola pada plat nomor kendaraan menggunakan metode Jaringan Syaraf Tiruan Data yang diambil sebanyak 160 yang terdiri atas 40 citra wilayah Bogor (F), 40 citra wilayah Jakarta (B), 40 citra wilayah Bandung (D) dan 40 citra wilayah Purwakarta (T). Data dari masing-masing jenis plat nomor dibagi menjadi dua bagian, 30 buah untuk data latih dan 10 buah untuk data uji. Dari uji coba berdasarkan 40 data latih

yang dilakukan terdapat 31 data uji yang terdefinisi atau akurasi sebesar 77,5% (Maryana, Qur'ania, dan Putra 2019).

## **2.2 Dasar Teori**

### **2.2.1. Tanda Nomor Kendaraan Bermotor**

Tanda Nomor Kendaraan Bermotor (TNKB) adalah tanda atau simbol yang berupa huruf, angka, kombinasi huruf, angka yang memuat kode Tanda nomor kendaraan Bermotor harus memenuhi syarat bentuk, ukuran, bahan, warna, dan cara pemasangan. Tanda Nomor Kendaraan Bermotor diatur dengan peraturan Kepolisian Negara Republik Indonesia Setiap kendaraan di Indonesia yang telah melakukan registrasi pasti memiliki sebuah Tanda Nomor Kendaraan Bermotor (TNKB). Aturan atau regulasi mengenai plat kendaraan ini berbeda-beda disetiap satu negara dengan negara yang lainnya. Di Indonesia, Plat nomor kendaraan selalu dibuat seragam atau sama antara satu jenis kendaraan dengan kendaraan lain, baik dalam ukuran plat, huruf pada plat nomor pun selalu sejenis (Nur'aini dan Rusdiana 2021). Seperti pada gambar 2.1



Gambar 2.1 Tanda Nomor Kendaraan Bermotor

### **2.2.2. Automatic Number Plate Recognition**

*Automatic Number Plate Recognition* (ANPR) adalah teknik yang dirancang untuk membaca plat nomor kendaraan dengan menggunakan pengambilan gambar berkecepatan tinggi dengan penerangan yang mendukung, deteksi karakter dalam gambar yang disediakan, verifikasi urutan karakter seperti yang ada pada plat nomor kendaraan pengenalan karakter hingga mengubah gambar menjadi teks, dijadikan sekumpulan metadata yang mengidentifikasi gambar yang berisi pelat nomor kendaraan dan kode yang terkait dengan plat tersebut (Adak, 2023).

### 2.2.3. OpenCV

OpenCV adalah sebuah pustaka yang bersifat *open source* yang dikembangkan oleh intel yang fokus untuk menyederhanakan pemrograman terkait dengan pengolahan citra digital. OpenCV memiliki banyak fitur terkait visi komputer (*computer vision*) antara lain : pengenalan wajah, deteksi wajah, *Kalman filtering*, dan berbagai jenis metoda AI (*Artificial Intellegence*) dan dapat di jalan di berbagai bahasa pemograman, dan juga mendukung berbagai platform sistem operasi seperti Windows, Linux, Mac OS, iOS dan Android (Iman dan Syamsuddin 2020). Secara resmi OpenCV diluncurkan pada tahun 1999, yang merupakan program penelitian milik *Intel Corporation* untuk mendukung aplikasi *CPU-intensive*. OpenCV adalah platform populer untuk mengimplementasikan algoritma deteksi dan pengenalan objek (Hasan dan Sallow, 2021).



Gambar 2.2 Logo OpenCV

### 2.2.4. Object Detection Algorithm

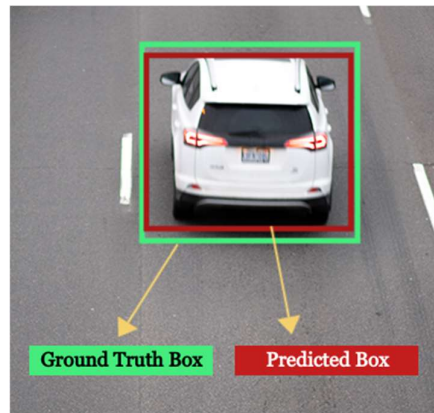
*Object detection algorithm* merupakan salah satu algoritma computer vision yang berfungsi untuk menentukan di mana lokasi objek pada suatu citra berada dengan *bounding box*, sekaligus mengklasifikasikan objek tersebut ke dalam sebuah kelas yang telah ditentukan. Metrik yang paling cocok dan sering digunakan untuk mengukur kemampuan algoritma deteksi objek adalah *mean Average-Precision* (mAP) (Wiranda dan Putra 2022).

Untuk menentukan dan membandingkan kinerja prediktif model deteksi objek yang berbeda, memerlukan standar kuantitatif metrik. Dua metrik yang paling umum adalah metrik *Intersection over Union* (IoU) dan *Average Precision* (AP).

#### 1) *Intersection over Union* (IoU)

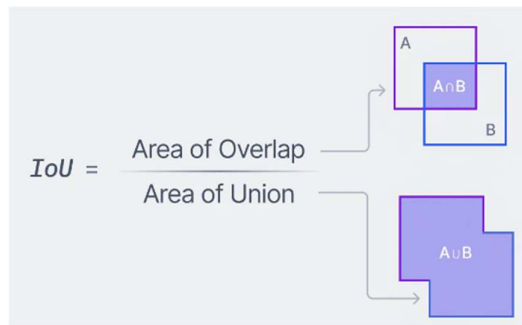
IoU adalah ukuran tumpang tindih antara kotak pembatas yang diprediksi

dan kotak pembatas kebenaran dasar. Dihitung dengan mengambil luas perpotongan kedua kotak dan membaginya dengan luas penyatuannya, seperti pada gambar 2.3.



Gambar 2.3 Nilai IoU

Nilai IoU yang lebih tinggi menunjukkan kecocokan yang lebih baik antara *predicted Box* dan *Ground Truth Box*. Penghitung tumpukan *ground-truth bounding box* dengan *predicted bounding box*. Secara matematis ditulis sebagai berikut :



Gambar 2.4 Rumus IoU

$$\text{Intersection over Union (IoU)} = \frac{|A \cap B|}{|A \cup B|} \quad 2.2$$

$$\text{Intersection over Union (IoU)} = \frac{TP}{TP + FN + FP}$$

Keterangan :

TP = *True Positive*

FN = *False Negative*

FP = *False Positive*

## 2) *Average Precision*

*Average Precision* adalah metrik untuk mengukur keakuratan pendeteksi

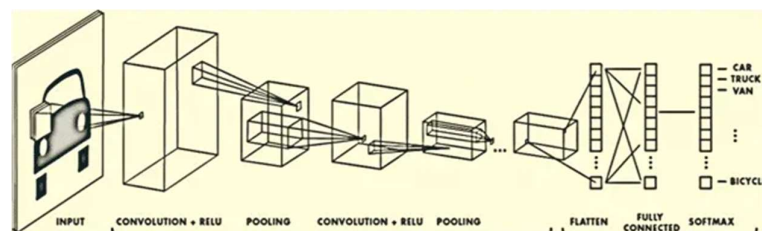
objek lebih cepat seperti R-CNN, SSD, dan lain-lain. *Average Precision* menghitung nilai *precision* dengan nilai *recall* dari 0 hingga 1.

### 2.2.5. EasyOCR

*EasyOCR* adalah modul *Python* untuk ekstraksi teks dari gambar. Ini adalah *OCR* umum yang dapat membaca teks pada *billboard* hingga teks padat pada dokumen. Saat ini mendukung lebih dari 80 bahasa dan terus berkembang. Jaidev AI, sebuah perusahaan yang berspesialisasi dalam layanan Pengenalan Karakter Optik, membuat dan *maintains* paket *EasyOCR*. *Python* dan perpustakaan *PyTorch* digunakan untuk mengimplementasikan *EasyOCR*. Jika memiliki perangkat dengan *GPU* berkemampuan *CUDA*, *deep learning library* milik *PyTorch* dapat meningkatkan deteksi teks dan kecepatan *OCR* secara signifikan (Chawla, Gupta, dan Shushrutha, 2022).

### 2.2.6. Convolutional neural networks

*Convolutional neural networks* (CNN) merupakan algoritma *deep learning* yang memiliki fitur *weight sharing*, yang mengurangi jumlah parameter jaringan yang dapat dilatih dan pada gilirannya membantu jaringan untuk meningkatkan generalisasi dan menghindari *overfitting* serta implementasi *large-scale network* jauh lebih mudah dengan CNN dibandingkan dengan *neural networks* lainnya (Alzubaidi, 2021), seperti pada gambar 2.4



Gambar 2.5 Arsitektur CNN

Lapisan convolution merupakan inti dari setiap CNN. Lapisan ini digunakan untuk mengekstrak informasi dari inputnya melalui penggunaan sejumlah filter yang secara otomatis diajarkan untuk mendeteksi fitur tertentu dalam suatu gambar.. Setiap filter akan memindai input dari sudut kiri atas ke sudut kanan bawah, masing-masing membuat *feature map* (Tabian, Fu dan Khodaei, 2019).

Lapisan *pooling* melakukan *down-sampling* pada lebar dan tinggi,

mengurangi dimensi inputnya dan, karenanya, mengurangi jumlah parameter yang akan dihitung. Ini mengurangi kompleksitas dan kemungkinan *overfitting*. *Pooling* beroperasi pada setiap irisan kedalaman input secara terpisah, *down-sampling* semuanya dengan cara yang sama. Setiap irisan akan dibagi menjadi sejumlah *patch*, yang luasnya sama dengan ukuran filter yang ditetapkan oleh pengguna saat mendefinisikan lapisan *pooling*. Output dari lapisan *pooling* akan menjadi volume yang lebih kecil, tetapi kedalamannya sama dengan input. Ada beberapa jenis lapisan *pooling*, yang dikategorikan berdasarkan cara operasi ini dilakukan (Brownlee, 2019).

- a. *Average Pooling*: Menghitung rata-rata angka dalam setiap *patch* dan mengirimkannya ke posisi yang sesuai dalam output seperti pada Gambar 2.6.

|   |   |                 |                 |                 |
|---|---|-----------------|-----------------|-----------------|
| 1 | 0 | 0               | 1               | 0               |
| 1 | 1 | 0               | 0               | 1               |
| 0 | 0 | 1 <sub>x0</sub> | 1 <sub>x1</sub> | 0 <sub>x0</sub> |
| 1 | 0 | 0 <sub>x0</sub> | 0 <sub>x0</sub> | 1 <sub>x1</sub> |
| 0 | 1 | 1 <sub>x0</sub> | 1 <sub>x1</sub> | 0 <sub>x1</sub> |

Gambar 2.6 Operasi konvolusi

- b. *Max Pooling*: Untuk setiap patch, nilai maksimum dikirimkan ke *output*. Jenis ini memiliki kinerja yang lebih baik dan digunakan di semua lapisan penggabungan CNN seperti pada Gambar 2.7.

|   |   |   |   |
|---|---|---|---|
| 1 | 6 | 3 | 1 |
| 7 | 3 | 5 | 2 |
| 9 | 4 | 1 | 3 |
| 5 | 8 | 4 | 1 |

Max Pooling  
with filter  
size (2,2)

|   |   |
|---|---|
| 7 | 5 |
| 9 | 4 |

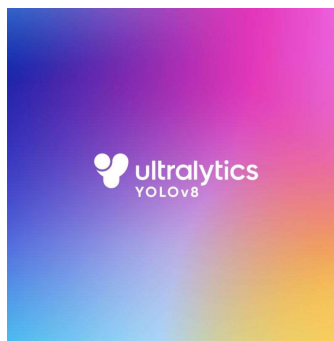
Gambar 2.7 Max Pooling dengan ukuran filter (2,2)

*Fully Connected Layer* tersebut adalah layer yang biasanya digunakan dalam penerapan MLP (*Multi Layer Perceptron*) dan bertujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara *linear*. Setiap *neuron* pada *convolution layer* perlu ditransformasi menjadi data satu dimensi

terlebih dahulu sebelum dapat dimasukkan ke dalam sebuah *fully connected layer*. Karena hal tersebut menyebabkan data kehilangan informasi spasialnya dan tidak reversibel, *fully connected layer* hanya dapat diimplementasikan di akhir jaringan (Suartika, Wayan, Yudhi, 2019).

#### 1) *You Only Look Once*

YOLOv8 merupakan salah satu *state-of-the-art object detection algorithm* yang membagi citra masukan ke dalam suatu grid berukuran  $S \times S$ . Ukuran dari grid cell tersebut tergantung pada input size yang digunakan pada suatu arsitektur. Pada YOLOv8, jika *input size*  $416 \times 416$ , maka ukuran *grid size* adalah  $13 \times 13$ ,  $26 \times 26$ , dan  $52 \times 52$ . Setiap sel bertugas untuk memprediksi objek yang ada di dalam sel tersebut dengan *bounding box* beserta *confidence* yang merupakan nilai probabilitas keberadaan suatu objek pada *bounding box* tersebut. Kemudian, setelah *bounding box* dipetakan berdasarkan nilai *confidence* yang dihasilkan, YOLOv8 akan memprediksi kelas dari objek yang terdapat pada *bounding box* tersebut beserta probabilitasnya, sehingga terbentuklah *class probability map* (Asyhar, Wibowo, dan Budiman 2020).



Gambar 2.8 *You Only Look Once*

#### 2.2.7. *Python*

*Python* adalah bahasa pemrograman yang sangat populer dengan banyak fitur hebat untuk mengembangkan Kecerdasan Buatan. Banyak pengembang Kecerdasan Buatan di seluruh dunia menggunakan *Python*, *Python* memiliki banyak paket seperti *TensorFlow*, *Keras*, dan *Theano* yang membantu *data scientist* dalam mengembangkan algoritme *deep learning*. *Python* memberikan bantuan yang unggul sehubungan dengan algoritma *deep learning* (Teoh dan Rong 2022). *Python* dapat digunakan untuk berbagai keperluan, seperti pengembangan aplikasi web,



aplikasi desktop, IoT, dan berbagai aplikasi lainnya. *Python* juga memiliki integrasi dengan sistem database dan mampu membaca serta mengubah file, sehingga sering digunakan untuk *prototyping* atau pengembangan perangkat lunak dengan cepat dan reliabel (Rahman., 2023).



Gambar 2.9 Python

*Python* mendukung berbagai paradigma pemrograman, terutama namun tidak terbatas pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang dimiliki oleh *Python* adalah sebagai bahasa pemrograman dinamis dengan manajemen memori otomatis. Seperti halnya bahasa pemrograman dinamis lainnya, *Python* sering digunakan sebagai bahasa skrip, meskipun pada kenyataannya, penggunaannya lebih luas dan mencakup berbagai konteks pemanfaatan yang tidak terbatas pada bahasa skrip. *Python* dapat digunakan untuk berbagai tujuan dalam pengembangan perangkat lunak dan dapat dijalankan di berbagai platform sistem operasi (Muhamad Zulkarnaen, 2024).

#### 2.2.8. *NumPy*

*NumPy* adalah *library* untuk bahasa pemrograman *Python*, *NumPy* memberikan dukungan untuk himpunan dan *matriks multidimensi* yang besar, dan dilengkapi koleksi sejumlah besar fungsi matematika tingkat tinggi untuk beroperasi pada himpunan ini yang menyediakan komputasi objek array multidimensi dan objek turunan seperti *masked arrays* dan *matrix*. *NumPy* memberikan berbagai fungsi operasi aljabar, statistik dasar, simulasi acak, matematika, logika, manipulasi bentuk, pengurutan, pemilihan, I/O, transformasi Fourier diskrit, dan lain-lain (Harris . 2020).

Saat *generating array*, *NumPy* akan menggunakan *bit depth* atau *color depth* dengan *environment Python* secara *default*. Jika bekerja dengan *Python 64-bit*,

maka elemen array akan menggunakan presisi 64-bit secara default. Presisi ini membutuhkan banyak memori dan tidak selalu diperlukan. Dan dapat menentukan *bit depth* saat membuat array dengan menyetel parameter tipe data (*dtype*) ke `int`, `numpy.float16`, `numpy.float32`, atau `numpy.float64` (Lawrence, 2019).

### **2.2.9. Pandas**

*Pandas* adalah *library* untuk analisis dan manipulasi data. *Pandas* dapat membaca data dalam banyak format, termasuk CSV, TSV, Excel, JSON, dan SQL. Secara umum, ada dua struktur yang berguna untuk manipulasi data di *Pandas* yaitu *Series* dan *DataFrame*. *Series Pandas* pada dasarnya adalah struktur satu dimensi, seperti kolom di Excel. *Dataframe Pandas* adalah struktur dua dimensi yang struktur datanya berupa tabel dengan baris dan kolom (Palupi, Ihsanto, dan Nugroho 2023).

*Pandas* membantu dalam menyusun data dalam file .CSV dalam berbagai kategori seperti “*frame\_number*”, “*track\_id*”, “*car\_bbox*”, “*car\_bbox\_score*”, “*license\_plate\_bbox*”, “*license\_plate\_bbox\_score*”, “*license\_plate\_number*” dan “*license\_text\_score*”. *Pandas* juga mendukung bahasa *querying* untuk ekstraksi informasi dari kerangka data dan dapat mengekstrak kolom dan/atau baris tertentu menggunakan *querying* (Agarwal, 2019).

### **2.2.10. Flowchart**

*Flowchart* merupakan suatu jenis diagram yang merepresentasikan algoritma atau langkah-langkah instruksi yang berurutan dalam sistem sebagai bukti dokumentasi untuk menjelaskan gambaran logis sebuah sistem yang akan dibangun kepada *programmer*. Dengan begitu, *flowchart* dapat membantu untuk memberikan solusi terhadap masalah yang bisa saja terjadi dalam membangun sistem. Pada dasarnya, *flowchart* digambarkan dengan menggunakan simbol-simbol. Setiap simbol mewakili suatu proses tertentu. Sedangkan untuk menghubungkan satu proses ke proses selanjutnya digambarkan dengan menggunakan garis penghubung (Rosaly dan Prasetyo 2020).

### **2.1.11. Metode Waterfall**

Metode *waterfall* merujuk pada pendekatan yang sistematis dan berurutan dalam pengembangan perangkat lunak. Proses ini melibatkan langkah-langkah

secara berurutan, dimulai dari spesifikasi kebutuhan pengguna, kemudian melalui tahap perencanaan, seperti perencanaan, pemodelan, konstruksi sistem, hingga akhirnya menyerahkan sistem kepada pengguna. Inti dari metode *waterfall* adalah bahwa pengerjaan satu sistem dilakukan secara berurutan atau linear. Dengan kata lain, setiap tahap harus diselesaikan sepenuhnya sebelum melanjutkan ke tahap berikutnya, menghindari pengulangan garis waktu untuk mencapai kemajuan (Kurniawan, 2023).