

ANALISIS PERBANDINGAN ALGORITMA CNN DAN YOLO DALAM MENGIDENTIFIKASI KERUSAKAN JALAN

Nabila Khairunisa^{1*}, Carudin², Asep Jamaludin³

^{1,2,3} Universitas Singaperbangsa Karawang; Jl. HS.Ronggo Waluyo, Puseurjaya, Telukjambe Timur, Karawang, Jawa Barat 41361, Telp. (0267) 641177

Received: 25 Mei 2024

Accepted: 31 Juli 2024

Published: 7 Agustus 2024

Keywords:

Convolutional Neural Network, You Only Look Once, Uji-T Berpasangan, Kerusakan Jalan.

Correspondent Email:

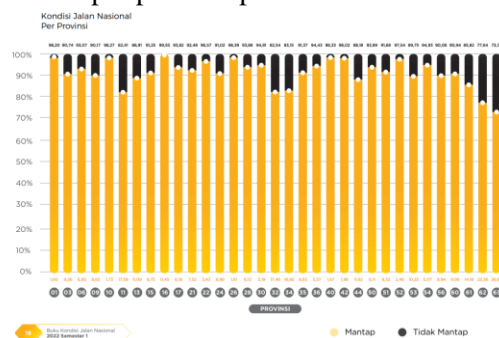
2010631170101@student.unsika.ac.id

Abstrak. Jalan merupakan infrastruktur penting dalam kehidupan masyarakat yang memiliki peran dalam mendukung pertumbuhan ekonomi, mobilitas, dan konektivitas antar wilayah. Penelitian ini bertujuan untuk membandingkan kinerja dua algoritma yaitu Convolutional Neural Network (CNN) dan You Only Look Once (YOLO), dalam mengidentifikasi kerusakan jalan. Tiga variabel instrumen dievaluasi, yaitu waktu pelatihan, kecepatan deteksi, dan akurasi pengujian. Metode analisis yang digunakan adalah Uji-T Berpasangan dengan tingkat signifikansi alpha sebesar 0.05. Hasil pengujian menunjukkan bahwa terdapat perbedaan signifikan dalam waktu pelatihan antara CNN dan YOLO ($P\text{-Value} < \alpha$), yang mengindikasikan bahwa salah satu algoritma memerlukan waktu pelatihan yang lebih sedikit daripada yang lain. Namun, tidak terdapat perbedaan signifikan dalam kecepatan deteksi maupun akurasi pengujian ($P\text{-Value} > \alpha$) antara kedua algoritma. Temuan ini menunjukkan bahwa meskipun ada perbedaan dalam waktu pelatihan antara CNN dan YOLO, keduanya memiliki kinerja yang setara dalam hal kecepatan deteksi dan akurasi pengujian dalam konteks pengidentifikasian kerusakan jalan. Oleh karena itu, pemilihan antara kedua algoritma ini dapat dipertimbangkan berdasarkan faktor-faktor lain seperti kebutuhan spesifik proyek dan kemampuan komputasi yang tersedia.

Abstract. Roads are important infrastructure in people's lives that have a role in supporting economic growth, mobility, and connectivity between regions. This research aims to compare the performance of two algorithms, namely Convolutional Neural Network (CNN) and You Only Look Once (YOLO), in identifying road damage. Three instrument variables were evaluated, namely training time, detection speed, and testing accuracy. The analysis method used was the Paired T-Test with an alpha significance level of 0.05. The test results show that there is a significant difference in training time between CNN and YOLO ($P\text{-Value} < \alpha$), which indicates that one algorithm requires less training time than the other. However, there was no significant difference in detection speed or testing accuracy ($P\text{-Value} > \alpha$) between the two algorithms. This finding suggests that despite the difference in training time between CNN and YOLO, they perform equally in terms of detection speed and testing accuracy in the context of road defect identification. Therefore, the selection between these two algorithms can be considered based on other factors such as project-specific requirements and available computational capabilities.

1. PENDAHULUAN

Jalan merupakan infrastruktur penting dalam kehidupan masyarakat yang memiliki peran dalam mendukung pertumbuhan ekonomi, mobilitas, dan konektivitas antar wilayah [1]. Namun, kerusakan jalan telah menjadi masalah di banyak negara. Kategori kerusakan jalan tersebut meliputi jalan berlubang, bergelombang, retakan atau patah-patah dan permukaan kasar [2]. Berikut merupakan grafik mengenai kondisi jalan nasional per provinsi pada Gambar 1 berikut:



Gambar 1. Kondisi Jalan Nasional Per Provinsi (Sumber: Kementerian PUPR (Buku Kondisi Jalan Nasional 2022))

Berdasarkan data tersebut, maka diperlukan pemantauan mengenai kondisi jalan yang dapat diaplikasikan dengan metode manual atau otomatis. Terdapat macam metode yang tersedia untuk mengidentifikasi kerusakan jalan secara otomatis, diantaranya yaitu Convolutional Neural Network (CNN) dan You Only Look Once (YOLO). Saat ini CNN merupakan salah satu metode deep learning yang mempunyai hasil paling relevan dalam bidang pengenalan citra [3]. CNN merupakan model jenis Deep Learning yang terinspirasi oleh bagaimana manusia memahami dunia visual, dan model ini mampu digunakan untuk menganalisis data gambar [4]. Sedangkan metode YOLO sendiri juga merupakan sebuah model dalam deep learning yang digunakan untuk mendeteksi objek dalam gambar dan video secara real-time [5].

Penelitian algoritma YOLO mengenai “Deteksi Kendaraan dengan Metode YOLO” menjelaskan bahwa YOLO cenderung kesulitan mendeteksi objek kecil atau objek yang

memiliki proporsi yang sangat kecil dalam gambar. Selain itu, YOLO memiliki kinerja yang kurang baik jika gambar memiliki resolusi rendah atau detail yang kurang jelas [6].

Berdasarkan hasil penelitian sebelumnya, metode CNN dan YOLO memiliki kelebihan dan kekurangannya masing-masing. CNN terkenal karena kemampuannya dalam mengekstrak fitur hierarkis dari gambar, namun membutuhkan data pelatihan besar dan sumber daya komputasi yang besar. YOLO dikenal karena kemampuannya dalam deteksi objek real-time, terutama pada perangkat terbatas, namun sering kurang presisi dalam deteksi objek yang berdekatan atau dalam kondisi pencahayaan ekstrem.

Dibalik kelebihan dan kekurangan metode CNN dan YOLO dapat dikatakan bahwa kedua algoritma tersebut berhasil dalam mengidentifikasi objek, terutama mengidentifikasi kerusakan jalan. Oleh sebab itu, pada penelitian sekarang akan membahas mengenai analisis perbandingan kedua algoritma tersebut. Dimana, analisis perbandingan penting dilakukan untuk mengetahui kelebihan dan kekurangan masing-masing algoritma yang dapat mempengaruhi kinerja aplikasi identifikasi kerusakan jalan. Perbandingan ini dapat membantu dalam menentukan algoritma yang paling tepat untuk digunakan dalam suatu aplikasi tertentu. Metode yang akan digunakan dalam membandingkan kedua algoritma tersebut yaitu Uji-T Berpasangan dengan beberapa variabel seperti kecepatan deteksi, hasil akurasi, dan waktu pelatihan. Dalam konteks membandingkan dua algoritma, uji-T berpasangan dapat diterapkan untuk mengukur apakah terdapat perbedaan signifikan antara kinerja kedua algoritma tersebut. Dengan adanya penelitian ini diharapkan mampu berkontribusi bagi penelitian ilmiah dan memperluas pemahaman tentang algoritma. Penelitian ini juga dapat dijadikan sebagai referensi penelitian selanjutnya untuk mengembangkan model terhadap deteksi objek.

2. TINJAUAN PUSTAKA

2.1. Spesifikasi Hardware

Spesifikasi hardware mempengaruhi kinerja dan kemampuan sistem komputer. Komponen seperti prosesor, RAM, dan penyimpanan memengaruhi kecepatan pemrosesan data, kapasitas untuk menjalankan aplikasi, dan kemampuan untuk menangani tugas yang kompleks. Keberhasilan sebuah program atau aplikasi juga dapat dipengaruhi oleh kecocokan antara spesifikasi hardware yang digunakan dan kebutuhan program yang dirancang. Spesifikasi hardware yang digunakan dalam penelitian yaitu:

| | |
|-------------------------|--|
| <i>System Model</i> | Acer Aspire A514-54 |
| <i>Operating System</i> | Windows 11 64-bit |
| <i>BIOS</i> | V1.33 |
| <i>Processor</i> | 11 th Gen Intel® Core™ i5-1135G @ 2.40GHz (8 CPUs), ~2.4GHz |
| <i>Memory</i> | 8192MB RAM |
| <i>Page File</i> | 10768MB used, 5184MB available |

Tabel 1 Spesifikasi Hardware

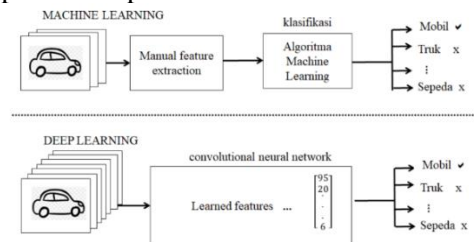
2.2. Computer Vision

Computer vision merupakan bidang ilmu komputer yang berfokus pada pengembangan algoritma dan teknologi yang memungkinkan komputer memahami informasi visual dari dunia fisik. Dalam konteks computer vision, informasi visual dapat berasal dari berbagai sumber, seperti gambar, video, citra medis, data sensor optik, dan lainnya [7]. Computer vision bertujuan untuk meniru cara kerja sistem visual manusia (human vision). Visual manusia sangat kompleks dengan menggunakan indera penglihatan untuk melihat sesuatu, dan gambar yang mereka lihat dikirim ke otak untuk ditafsirkan, sehingga manusia dapat memahami apa yang dilihat.

2.3. Deep Learning

Deep Learning merupakan salah satu subbidang dalam Machine Learning yang berfokus pada penggunaan neural networks dengan banyak lapisan (deep neural networks) dalam memproses data [8]. Pada umumnya deep learning memiliki struktur jaringan yang sama dengan neural network, yaitu tersusun atas input layer, hidden layer, dan output layer. Deep learning memiliki aturan umum yang

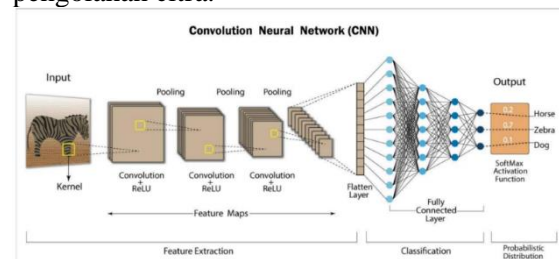
lebih banyak lapisan daripada neural network, oleh karena itu disebut deep. Salah satu kemungkinan deep learning adalah mengganti fungsi manual dengan algoritma yang efisien untuk pembelajaran Hirarkis Unsupervised, Semi-Supervised Feature Learning, dan Hierarchical Feature Extraction [9]. Perbedaan mengenai Machine learning dan Deep learning dapat dilihat pada Gambar 2 dibawah ini.



Gambar 2 Machine Learning vs Deep Learning

2.4. Convolutional Neural Network

CNN merupakan salah satu metode deep learning yang mempunyai hasil pengembangan dari Multilayer Perceptron yang dibuat untuk melakukan pengolahan data ke bentuk dua dimensi, seperti gambar, video, dan suara [2]. Teknik CNN terinspirasi dari cara manusia menghasilkan persepsi visual dan biasa digunakan pada data gambar (Nyoman et al., n.d.). Pada dasarnya CNN merupakan pengembangan dari metode Artificial Neural Network (ANN) konvensional. Pada ANN terdapat tiga sampai empat layer, sedangkan CNN terdiri hingga ratusan layer. CNN memiliki kemampuan feature learning lebih baik daripada metode yang lainnya. Pada Gambar 3 dibawah ini dapat dilihat mengenai alur dari proses CNN dalam melakukan pengolahan citra.



Gambar 3 Arsitektur Model CNN

Menurut [10] terdapat beberapa parameter yang digunakan dalam algoritma CNN, diantaranya yaitu:

1. Epoch: jumlah iterasi dari proses pelatihan model pada data training. Semakin banyak epoch, semakin baik model akan

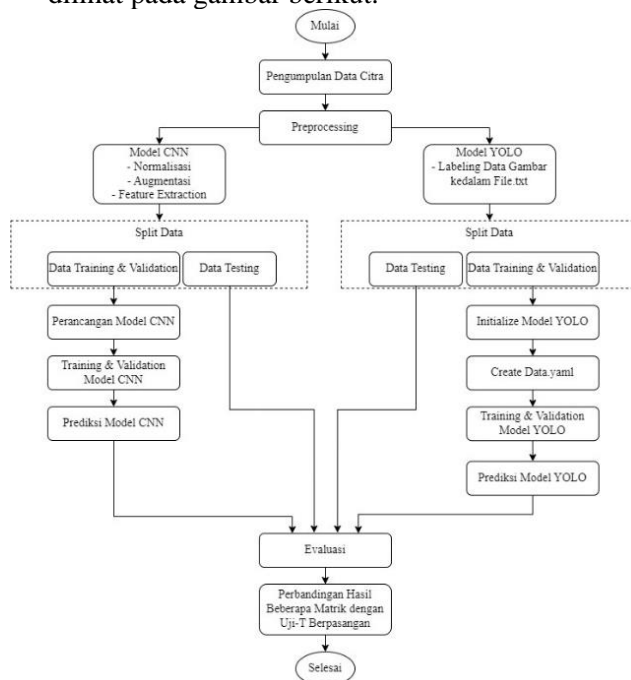
n = Jumlah sampel

Interpretasi:

1. Untuk menginterpretasikan uji t-test terlebih dahulu harus menentukan:
 - Nilai signifikan α
 - DF (Degree of Freedom) = $N - k$, khusus untuk uji-t berpasangan $df = N - 1$
2. Bandingkan nilai t_{hit} dengan $t_{tab} = \alpha; n-1$
3. Apabila:
 - $t_{hit} > t_{tab}$: berbeda secara signifikan (H_0 ditolak)
 - $t_{hit} < t_{tab}$: tidak berbeda secara signifikan (H_0 diterima)

3. METODE PENELITIAN

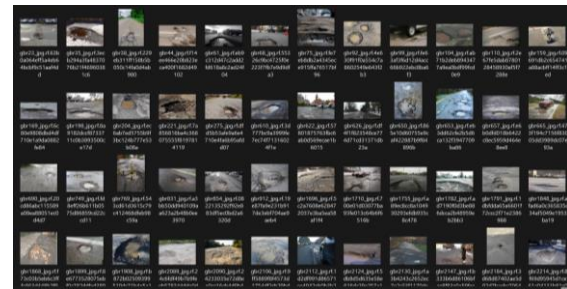
Metode penelitian yang digunakan yaitu membandingkan dua algoritma dengan dataset yang sama. Alur dari metode tersebut dapat dilihat pada gambar berikut.



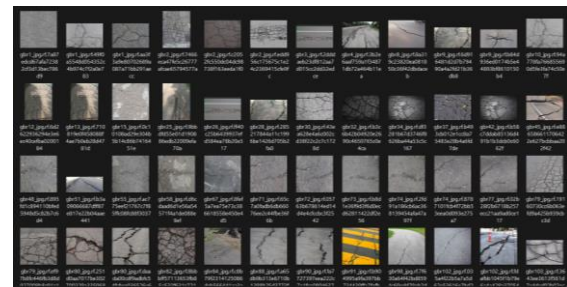
Gambar 5 Metode Penelitian

3.1. Pengumpulan Data

Pengumpulan dataset merupakan tahapan awal dalam proses pengembangan model deteksi objek. Dataset ini diambil dari berbagai sumber, sudut pandang, dan dalam kondisi yang berbeda-beda. Gambar 6 dan 7 menunjukkan mengenai pengumpulan dataset pada setiap kategorinya.



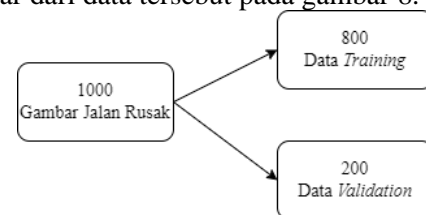
Gambar 6 Dataset Jalan Berlubang



Gambar 7 Dataset Jalan Retak

3.2. Pembagian Dataset

Pembagian dataset merupakan salah satu hal penting untuk memastikan model dapat mempelajari data dengan baik saat melakukan testing di situasi dunia nyata. Dataset dibagi menjadi training, validation, dan testing. Dengan membagi dataset, model dapat mempelajari variasi data dan menghasilkan prediksi yang akurat. Beberapa teknik partisi dataset seperti berikut, 90:10, 80:20, dan 70:20:10. Pada penelitian ini dataset dibagi menjadi 80% untuk training, 20% untuk validation dan untuk testing menggunakan data di luar dari data tersebut pada gambar 8.



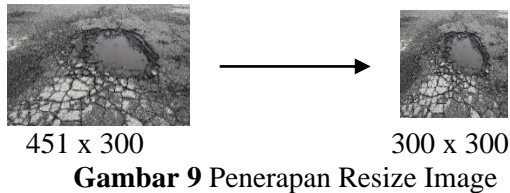
Gambar 8 Pembagian Dataset

3.3. Implementasi Model CNN

1. Resize Image

Tahap ini bertujuan untuk melakukan standarisasi dengan mengubah skala pixel tiap gambar agar memiliki ukuran yang sama. Hal tersebut karena gambar akan dijadikan sebagai input model yang akan dilatih. Model akan menerima input gambar dengan ukuran tertentu,

oleh karena itu diperlukan adanya penentuan ukuran gambar untuk seluruh dataset seperti Gambar berikut:



Gambar 9 Penerapan Resize Image

2. Augmentasi

Augmentasi dataset dilakukan dengan mengubah data pelatihan yang sudah ada sebagai bagian dari proses augmentasi dataset. Tujuan dari augmentasi dataset ini adalah mengurangi kemungkinan terjadinya overfitting, di mana model cenderung menghafal dataset secara berlebihan. Beberapa teknik augmentasi diantaranya yaitu rotation range, width shift range, height shift range, horizontal flip, vertical flip, shear range, dan zoom range.

```
# Membuat objek ImageDataGenerator untuk augmentasi data training
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

#Membuat objek ImageDataGenerator tanpa augmentasi untuk data validation
val_datagen = ImageDataGenerator(rescale=1./255)
```

Gambar 10 Penerapan Augmentasi

3. Feature Extraction

Setelah melalui langkah preprocessing data, langkah berikutnya adalah melakukan ekstraksi fitur pada citra. Proses ini melibatkan dua tahap, di mana pertama-tama dilakukan ekstraksi citra dengan memanfaatkan data augmentasi dari data pelatihan dan data validasi. Selanjutnya, disusunlah komposisi ukuran citra yang akan diterapkan pada model CNN. Dalam penelitian ini, ukuran citra yang digunakan adalah 128 x 128. Untuk implementasinya dapat dilihat pada gambar 11.

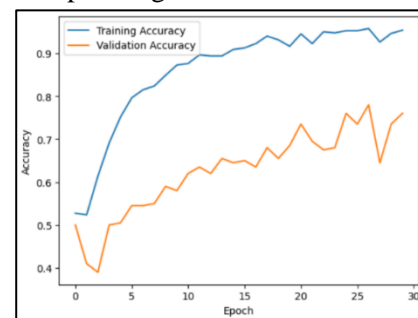
```
[ ] ## Menggunakan objek ImageDataGenerator untuk memuat data dari direktori
train_generator = train_datagen.flow_from_directory(
    train_path,
    target_size=(128, 128),
    batch_size=32,
    class_mode='categorical'
)

val_generator = val_datagen.flow_from_directory(
    val_path,
    target_size=(128, 128),
    batch_size=32,
    class_mode='categorical'
)
```

Gambar 11 Penerapan Feature Extraction

4. Evaluasi Model

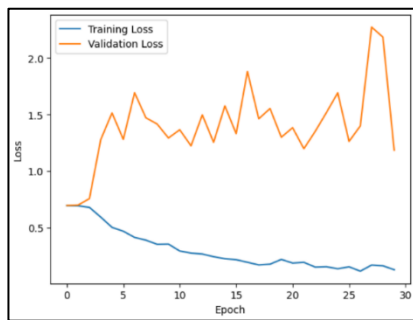
Berikut Gambar 12 akurasi training dan validation pada algoritma CNN.



Gambar 12 Akurasi Training & Validation

Grafik diatas menunjukkan akurasi pelatihan dengan warna biru dan validasi dengan warna oranye pada model Machine Learning setiap epoch. Keduanya dimulai dari 0,5 dan meningkat seiring waktu. Akurasi pelatihan mencapai puncak sekitar 0,9 pada epoch 25, sementara validasi mencapai puncak sekitar 0,8 pada epoch 15. Kesenjangan validasi adalah ukuran generalisasi model. Dalam kasus ini, kesenjangan relatif kecil, menunjukkan model tidak terlalu overfitting, tetapi perlu observasi lebih lanjut. Secara keseluruhan, model belajar dengan baik, tapi masih ada ruang untuk perbaikan. Evaluasi pada dataset pengujian akan memberikan gambaran yang lebih jelas.

Berikut Gambar 13 loss training dan validation pada algoritma CNN.



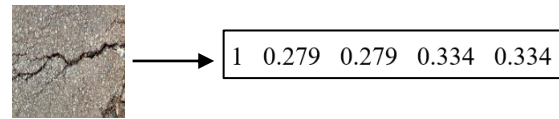
Gambar 13 Loss Training & Validation

Grafik diatas memperlihatkan loss training dan loss validation model ML pada setiap epoch. Loss training (biru) awalnya lebih tinggi dari loss validation (oranye), namun keduanya menurun seiring bertambahnya epoch. Training loss menurun lebih cepat, menandakan model memperbaiki akurasi dari data pelatihan. Namun, setelah sekitar epoch 15, training loss mulai meningkat lagi, sementara validation loss terus menurun, menunjukkan overfitting. Titik ini, disebut titik siku, menandakan waktu yang tepat untuk menghentikan pelatihan agar model tidak overfitting pada data baru. Meskipun model mungkin sedikit overfitting, coba tambahkan beberapa epoch lagi dengan hati-hati.

3.4. Implementasi Model YOLO

1. Labeling

Pada tahap ini dilakukan labeling image JPG yang diubah ke dalam format TXT yang merupakan proses menandai atau memberi anotasi pada image untuk menunjukkan lokasi atau batasan dari objek tertentu di dalamnya. Dalam konteks identifikasi objek, label ini seringkali mencakup informasi mengenai lokasi dan ukuran dari objek yang diidentifikasi. Secara khusus dalam format TXT yang digunakan untuk pelabelan image, setiap baris pada file TXT biasanya berisi informasi tentang satu objek, termasuk kelas objek, koordinat pusat, dan dimensi (lebar dan tinggi) objek. Format ini akan digunakan dalam algoritma YOLO (You Only Look Once). Contoh format label dalam file TXT bisa dilihat pada Gambar 4.10.



Gambar 14 Penerapan Labeling

2. Initialize YOLO Model

Tahap ini melibatkan pembuatan model deteksi objek YOLO dan menginisialisasi berbagai parameter dan bobot model tersebut. Inisialisasi terjadi saat akan membuat objek model dari suatu library atau framework yang menyediakan implementasi YOLO. Pada kasus ini digunakan YOLO v8 sebagai model untuk identifikasi, dimana model tersebut di import dari ultralytics yang merupakan perpustakaan (library) sumber terbuka yang menyediakan implementasi YOLO yang kuat dan mudah digunakan. Untuk mengimport model YOLO v8 dapat menggunakan script pada Tabel 2 berikut.

| Script | Keterangan |
|---|---|
| <code>!pip install ultralytics -q</code> | Menginstall library ultralytics melalui manajer paket Python "pip". |
| <code>From ultralytics import YOLO</code> | Import model YOLO dari library ultralytics dengan membuat instance dari kelas YOLO. |
| <code>Model = YOLO("yolov8m.pt")</code> | Membuat objek model YOLO v8 yang didalamnya sudah berisikan parameter dan arsitektur model. |

Tabel 2 Script Import Model YOLO v8

3. Create Data.yaml

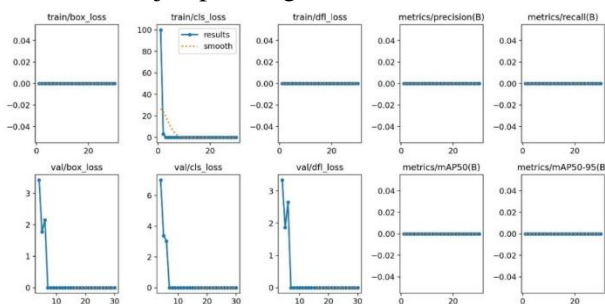
File data.yaml berfungsi sebagai file konfigurasi untuk melatih model YOLO menggunakan library Ultralytics. File ini menyimpan informasi penting terkait dataset pelatihan dan konfigurasi pelatihan. Untuk membuat data.yaml perlu menggunakan perintah shell `!touch data.yaml`. Dimana nantinya untuk isi dari file Data.yaml tersebut dapat dilihat pada gambar 15.

```
Train:
/content/drive/MyDrive/Skripsi/Dataset
YOLO/Training
val:
/content/drive/MyDrive/Skripsi/Dataset
YOLO/Validation
nc: 2
names: ["Pothole", "Crack"]
```

Gambar 15 Penerapan Create Data.yaml

4. Evaluasi Model

Grafik pada Gambar 4.16 di bawah merupakan hasil eksperimen untuk melatih kinerja model pembelajaran mesin untuk tugas deteksi objek pada algoritma YOLO.



Gambar 16 Hasil Eksperimen Model YOLO

Pada grafik diatas menjelaskan bahwa sumbu X menunjukkan epoch, atau jumlah iterasi pelatihan yang telah dilakukan model. Sumbu Y menunjukkan nilai dari beberapa metrik kinerja. Grafik tersebut dibagi menjadi dua bagian, yaitu grafik untuk data pelatihan (train) dan grafik untuk data pengujian (val).

Pada grafik pelatihan, kita dapat melihat bahwa nilai loss untuk semua metrik menurun seiring dengan bertambahnya epoch. Hal ini menunjukkan bahwa model deteksi objek tersebut semakin akurat dalam mendeteksi objek seiring dengan bertambahnya jumlah data pelatihan.

Pada grafik pengujian, kita dapat melihat bahwa nilai mAP50 dan mAP50-95 untuk model deteksi objek tersebut terus meningkat seiring dengan bertambahnya epoch. Hal ini menunjukkan bahwa model tersebut semakin baik dalam mendeteksi objek pada data pengujian.

4. HASIL DAN PEMBAHASAN

4.1. Metrik Perbandingan

Metrik perbandingan digunakan dalam mengukur efektivitas maupun kinerja suatu algoritma pada sistem yang dibuat. Terdapat beberapa metrik evaluasi yang digunakan dalam penelitian ini, yaitu:

1. Waktu Pelatihan










Waktu training yaitu total waktu yang diperlukan untuk melatih model pada data training. Semakin cepat waktu training, maka akan semakin bagus dalam pengembangan dan pemeliharaan model. Waktu training dapat dipengaruhi oleh kompleksitas model, jumlah data training, dan spesifikasi hardware.


| Epoch | CNN (s) | YOLO (s) |
|--------------|-------------|-------------|
| 1 | 221 | 70 |
| 2 | 110 | 25.64 |
| 3 | 106 | 23.92 |
| 4 | 109 | 24.03 |
| 5 | 112 | 23.69 |
| 6 | 109 | 23.47 |
| 7 | 108 | 24.27 |
| 8 | 106 | 24.63 |
| 9 | 108 | 23.47 |
| 10 | 116 | 23.80 |
| 11 | 105 | 23.47 |
| 12 | 107 | 24.27 |
| 13 | 105 | 24.03 |
| 14 | 108 | 23.47 |
| 15 | 114 | 23.80 |
| 16 | 106 | 24.03 |
| 17 | 107 | 23.69 |
| 18 | 105 | 23.58 |
| 19 | 106 | 23.80 |
| 20 | 116 | 23.69 |
| 21 | 108 | 26.59 |
| 22 | 107 | 23.36 |
| 23 | 109 | 23.47 |
| 24 | 106 | 23.58 |
| 25 | 113 | 24.27 |
| 26 | 109 | 23.58 |
| 27 | 110 | 23.58 |
| 28 | 110 | 23.58 |
| 29 | 109 | 23.69 |
| 30 | 109 | 23.47 |
| Total | 3374 s | 763.92 s |
| Menit | 56.23 Menit | 12.73 Menit |

Tabel 3 Waktu Pelatihan pada Setiap Epoch

2. Kecepatan Deteksi

Kecepatan deteksi digunakan untuk mengukur seberapa cepat model dalam mengenali objek, pola, atau fitur dalam gambar. Pada matrik ini, respon cepat sangat diperlukan dalam mendeteksi. Kecepatan deteksi dapat dipengaruhi oleh kompleksitas model, ukuran masukan gambar, dan spesifikasi hardware.




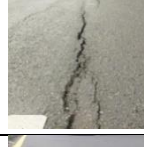



| No | Gambar | CNN (s) | YOLO v8 (s) |
|----|---|---------|-------------|
| 1 |  | 1 | 1 |
| 2 |  | 2 | 1 |
| 3 |  | 2 | 1 |
| 4 |  | 2 | 2 |
| 5 |  | 1 | 2 |
| 6 |  | 1 | 1 |
| 7 |  | 1 | 1 |
| 8 |  | 2 | 1 |
| 9 |  | 2 | 2 |




| | | | |
|--------------|--|-------------|-------------|
| 10 |  | 1 | 2 |
| Total | | 17 s | 14 s |

Tabel 4 Kecepatan Deteksi pada setiap Objek Pengujian

3. Akurasi Pengujian

Akurasi prediksi yaitu perbandingan antara jumlah prediksi yang benar dengan total jumlah sampel. Memberikan gambaran seberapa bagus model dapat menghasilkan hasil yang benar. Akurasi berguna dalam banyak kasus, tetapi bisa kurang efektif pada distribusi kelas yang tidak seimbang.

| Gambar | CNN | | YOLO | |
|---|---------|---------|---------|---------|
| | Deteksi | Akurasi | Deteksi | Akurasi |
|  | Ya | 100% | Ya | 98% |
|  | Ya | 99.9 % | Ya | 90% |
|  | Ya | 99.92 % | Ya | 85% |
|  | Ya | 72.69 % | Ya | 92% |
|  | Ya | 99.12 % | Ya | 100% |
|  | Ya | 99.99 % | Ya | 94% |
|  | Ya | 99.50 % | Ya | 98% |

| | | | | |
|---|----|---------|----|------|
|  | Ya | 100% | Ya | 99% |
|  | Ya | 98.03 % | Ya | 100% |
|  | Ya | 99.94 % | Ya | 100% |
| Total | | 969.09 | | 956 |

Tabel 4 Hasil Akurasi pada setiap Objek Pengujian

$$Akurasi\ CNN = \left(\frac{10}{969.09} \right) \times 100\%$$

$$Akurasi\ CNN = 96.9\%$$

$$Akurasi\ YOLO = \left(\frac{10}{956} \right) \times 100\%$$

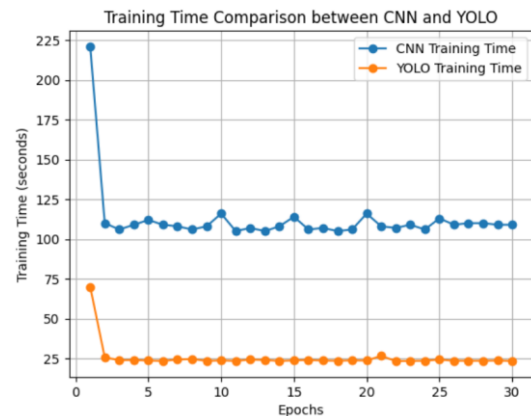
$$Akurasi\ YOLO = 95.6\%$$

4.2. Perbandingan Kinerja Uji-T

Pengujian dilakukan dengan Uji-T Berpasangan berbasis python sebagai bahasa pemrograman yang akan digunakan. Pengujian hipotesis ini dilakukan untuk menguji ada atau tidaknya perbedaan antara algoritma CNN dan YOLO dalam mengidentifikasi kerusakan jalan. Berikut adalah hasil uji hipotesis dengan menggunakan uji-t berpasangan pada kinerja algoritma CNN dan YOLO.

1. Waktu Pelatihan

Berikut merupakan grafik perbandingan waktu pelatihan (training) pada algoritma CNN dan YOLO dalam mengidentifikasi kerusakan jalan, yang dapat dilihat pada Gambar 4.17, dimana terdapat perbedaan waktu pelatihan yang sangat signifikansi antara kedua algoritma tersebut.



Gambar 17 Grafik Perbandingan Waktu Pelatihan CNN dan YOLO

Berikut merupakan hasil perbandingan kinerja pada waktu pelatihan dengan menggunakan Uji-T berpasangan, pada Tabel 5.

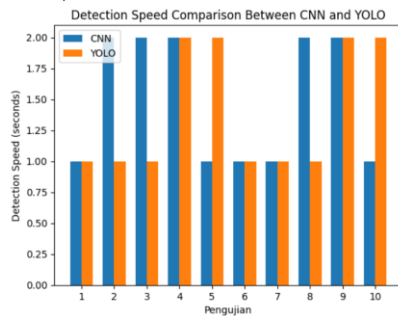
| Hasil Uji | Nilai |
|-------------|------------------------------------|
| Alpha | 0.05 |
| Statistik T | 38.213620970705904 |
| P-Value | 2.3313124186271488e ⁻²⁶ |

Tabel 5 Perbandingan Kinerja pada Waktu Pelatihan

Pada pengujian Uji-T, peneliti menggunakan alpha sebesar 0.05. Alpha tersebut digunakan untuk membantu peneliti dalam mengambil keputusan tentang signifikansi statistik dari perbedaan antara dua algoritma. Untuk Statistik T didapatkan nilai sebesar 38.213620970705904. Nilai tersebut mengukur seberapa besar perbedaan rata-rata antara dua sampel. Statistik T yang tinggi menunjukkan bahwa perbedaan antara waktu training kedua algoritma tersebut cukup besar. Dari hasil Uji-T pada tabel diatas menunjukkan bahwa P-Value < alpha (2.3313124186271488e-26 < 0.05), dimana perbedaan pada waktu training signifikan dan menunjukkan bahwa pengujian menolak hipotesis 0 (H0). P-value adalah probabilitas untuk mendapatkan hasil uji lebih ekstrim secara acak jika hipotesis 0 benar. P-value yang sangat rendah, seperti 2.3313124186271488e-26 menunjukkan bahwa kecepatan training algoritma YOLO jauh lebih cepat daripada algoritma CNN.

2. Kecepatan Deteksi

Pada kecepatan deteksi, waktu yang diperlukan kedua algoritma untuk mendeteksi objek terbilang cukup cepat dan hanya membutuhkan waktu satu sampai dua detik saja. Perbedaan antara algoritma CNN dan YOLO tidak signifikan seperti grafik pada Gambar 18, berikut.



Gambar 18 Grafik Perbandingan Kecepatan Deteksi CNN dan YOLO

Berikut merupakan hasil perbandingan kinerja pada kecepatan deteksi dengan menggunakan Uji-T berpasangan, pada Tabel 4.15.

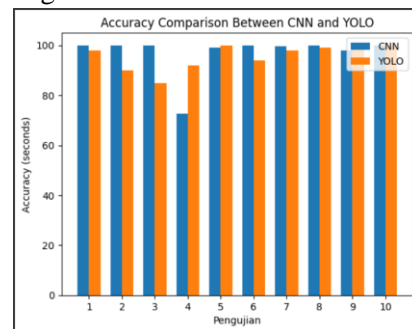
| Hasil Uji | Nilai |
|-------------|--------------------|
| Alpha | 0.05 |
| Statistik T | 0.4285714285714286 |
| P-Value | 0.6783097418055797 |

Tabel 6 Perbandingan Kinerja pada Kecepatan Deteksi

Pada pengujian digunakan nilai alpha sebesar 0.05 yang memberikan suatu trade-off yang umumnya dianggap wajar antara sensitivitas dan spesifisitas uji statistik. Statistik T dari perbandingan kecepatan deteksi menghasilkan nilai sebesar 0.4285714285714286. Dimana dengan nilai yang rendah menunjukkan bahwa perbedaan antara kecepatan deteksi dua algoritma tersebut relatif kecil. Sedangkan untuk P-Value dengan nilai sebesar 0.6783097418055797 atau P-Value > alpha. Dengan P-Value yang cukup tinggi, maka pengujian menerima hipotesis nol (H_0). Artinya, dalam konteks kecepatan deteksi, hasil uji tidak menunjukkan perbedaan yang signifikan antara dua algoritma yang diuji.

3. Akurasi Pengujian

Pada pengujian, terdapat 10 objek untuk dilakukan pengujian pada masing-masing algoritma. Kedua algoritma cukup baik dalam mendeteksi objek, hal tersebut dapat dilihat pada Gambar 19. Akurasi pada setiap pengujian dapat dikatakan sangat bagus karena dapat mendeteksi hingga 100%. Perbedaan akurasi antara algoritma CNN dan YOLO juga tidak terlalu signifikan.



Gambar 19 Grafik Perbandingan Akurasi Pengujian CNN dan YOLO

Berikut merupakan hasil perbandingan kinerja pada akurasi pengujian dengan menggunakan Uji-T berpasangan, pada Tabel 4.16.

| Hasil Uji | Nilai |
|-------------|---------------------|
| Alpha | 0.05 |
| Statistik T | 0.46147441328117766 |
| P-Value | 0.6554087075248882 |

Tabel 7 Perbandingan Kinerja pada Akurasi Pengujian

Uji-T pada akurasi pengujian dilakukan dengan menggunakan nilai alpha sebesar 0.05, yang merupakan standar yang umumnya diterima dan memperhitungkan secara seimbang sensitivitas dan spesifisitas uji statistik. Statistik T untuk perbandingan akurasi pengujian menghasilkan nilai sebesar 0.46147441328117766, menunjukkan bahwa perbedaan antara kecepatan deteksi kedua algoritma tersebut relatif kecil. Selain itu, P-Value yang diperoleh sebesar 0.6554087075248882, atau lebih besar dari alpha. Dengan nilai P-Value yang cukup tinggi,

pengujian menerima hipotesis 0 (H_0), yang mengindikasikan bahwa tidak ada perbedaan yang signifikan dalam akurasi pengujian antara kedua algoritma yang diuji. Dapat dikatakan bahwa kedua algoritma memiliki performa yang serupa dalam konteks akurasi objek yang diuji.

4.3. Pembahasan

1. Waktu Pelatihan

Hasil pengujian menunjukkan bahwa P-Value untuk waktu pelatihan sebesar $2.3313124186271488e-26$ dengan alpha sebesar 0.05, yang berarti $P\text{-Value} < 0.05$. Hal ini menunjukkan bahwa terdapat perbedaan yang sangat signifikan pada waktu pelatihan antara algoritma CNN dan YOLO. Dengan demikian, menunjukkan bahwa pengujian menolak hipotesis 0 (H_0), dimana H_0 yaitu tidak ada perbedaan yang signifikan antara kedua algoritma tersebut. Temuan ini menegaskan bahwa kecepatan pelatihan algoritma YOLO secara konsisten lebih cepat dibandingkan algoritma CNN.

2. Kecepatan Deteksi

Dalam analisis kecepatan deteksi dengan alpha sebesar 0.05, diperoleh nilai P-Value sebesar 0.6783097418055797, atau $P\text{-Value} > 0.05$. Dengan nilai P-Value yang tinggi tersebut, pengujian menerima hipotesis nol (H_0) yang menunjukkan bahwa hasil uji menemukan perbedaan yang sangat minim antara kedua algoritma yang diuji. Hal tersebut disebabkan karena perbedaan rata-rata kecepatan deteksi kedua algoritma hanya sebesar 0.3 second.

3. Akurasi Pengujian

Untuk akurasi pengujian menunjukkan bahwa $P\text{-Value} > \alpha$, dengan alpha 0.05 dan P-Values sebesar 0.6554087075248882. Oleh karena itu, hasil pengujian menunjukkan bahwa hipotesis nol (H_0) diterima, yang menyatakan bahwa tidak ada perbedaan yang signifikan dalam akurasi pengujian antara kedua algoritma yang diuji. Dengan demikian, dapat disimpulkan bahwa kinerja kedua algoritma

memiliki kesamaan dalam hal akurasi deteksi objek yang diuji. Hal ini didukung oleh penelitian sebelumnya yang dilakukan oleh (Amanda Putri et al., 2023), yang menunjukkan bahwa kedua algoritma memiliki tingkat akurasi yang sangat baik.

5. KESIMPULAN

Berdasarkan eksperimen dan pengujian yang telah dilakukan, terdapat beberapa point penting yang dapat disimpulkan diantaranya yaitu, Implementasi algoritma CNN memerlukan waktu pelatihan sekitar 56.23 menit untuk 30 epoch, dengan kecepatan deteksi mencapai 1.7 second per pengujian, dan mencapai akurasi rata-rata sebesar 96.9%.

Implementasi algoritma YOLO menunjukkan waktu pelatihan singkat, hanya sekitar 12.73 menit untuk 30 epoch, dengan kecepatan deteksi rata-rata sebesar 1.4 second per pengujian, dan akurasi pengujian rata-rata sebesar 95.6%.

Dalam hasil Uji-T Berpasangan, pada pengujian waktu pelatihan terjadi penolakan terhadap H_0 , artinya terdapat perbedaan yang sangat signifikan antara waktu pelatihan algoritma CNN dan YOLO. Namun, untuk kecepatan deteksi dan akurasi pengujian, temuan menunjukkan bahwa H_0 diterima, yang artinya tidak ada perbedaan untuk kecepatan deteksi dan akurasi pengujian antara kedua algoritma tersebut.

UCAPAN TERIMA KASIH

Terima kasih kepada semua pihak yang telah memberikan dukungan dan motivasi selama melakukan penelitian ini. Tanpa adanya dukungan dan motivasi tersebut, penulis tidak dapat menyelesaikan penelitian sampai akhir.

DAFTAR PUSTAKA

- [1] B. Sasmito, B. H. Setiadji, and R. Isnanto, "Deteksi Kerusakan Jalan Menggunakan Pengolahan Citra Deep Learning di Kota Semarang," *TEKNIK*, vol. 44, no. 1, pp. 7–14, May 2023, doi: 10.14710/teknik.v44i1.51908.
- [2] J. Rekayasa Sistem Komputer and F. H. MIPA Universitas Tanjungpura Jalan Hadari Nawawi Pontianak Telp, "Coding : Jurnal Komputer dan Aplikasi Agung Wira Mulia, [2] Ikhwani Ruslianto, [3] Dwi Marisa Midyanti [1] [2] [3]."

- [3] A. Riyandi, T. Widodo, S. Uyun, and U. Islam Negeri Sunan Kalijaga Yogyakarta, "Classification of Damaged Road Images Using the Convolutional Neural Network Method Klasifikasi Pada Citra Jalan Rusak Menggunakan Metode Convolutional Neural Network," *Jurnal Informatika dan Teknologi Informasi*, vol. 19, no. 2, pp. 147–158, 2022, doi: 10.31515/telematika.v19i2.6460.
- [4] N. Nyoman, C. Sumartha, G. Pasek, S. Wijaya, and F. Bimantoro, "Klasifikasi Citra Lubang Pada Permukaan Jalan Beraspal Dengan Metode Convolutional Neural Networks (Image Classification of Potholes on Paved Road Surfaces with the Convolutional Neural Networks (CNN) Method)."
- [5] K. A. Baihaqi and Y. Cahyana, "Application of Convolution Neural Network Algorithm for Rice Type Detection Using Yolo v3," 2021.
- [6] M. Fauzan Arif, A. Nurkholis, S. Laia, and P. Rosyani, "Deteksi Kendaraan Dengan Metode YOLO," *Jurnal Artificial Intelligent dan Sistem Penunjang Keputusan*, vol. 01, no. 01, 2023, [Online]. Available: <https://jurnalmahasiswa.com/index.php/aidanspk>
- [7] A. Esteva *et al.*, "Deep learning-enabled medical computer vision," *npj Digital Medicine*, vol. 4, no. 1. Nature Research, Dec. 01, 2021. doi: 10.1038/s41746-020-00376-2.
- [8] Y. Nurhadi, D. Iskandar Mulyana, and Y. Akbar, "Klasifikasi Rumput Liar Menggunakan Algoritme Deep Learning Dengan Dense Convolutional Neural Network".
- [9] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning", doi: 10.1007/s12525-021-00475-2/Published.
- [10] N. R. Fauziyya, "Metoda Convolutional Neural Network (CNN) untuk Pendeteksi Tangga pada Alat Pemandu Arah bagi Penyandang Tunanetra," *Telekontran : Jurnal Ilmiah Telekomunikasi, Kendali dan Elektronika Terapan*, vol. 8, no. 2, pp. 145–153, Apr. 2021, doi: 10.34010/telekontran.v8i2.4709.
- [11] J. H. Sri Wisna *et al.*, "Jurnal Sustainable: Jurnal Hasil Penelitian dan Industri Terapan," vol. 09, no. 01, pp. 8–14, 2020.
- [12] I. M. D. Maleh, R. Teguh, A. S. Sahay, S. Okta, and M. P. Pratama, "Implementasi Algoritma You Only Look Once (YOLO) Untuk Object Detection Sarang Orang Utan Di Taman Nasional Sebangau," *Jurnal Informatika*, vol. 10, no. 1, pp. 19–27, Mar. 2023, doi: 10.31294/inf.v10i1.13922.
- [13] S. Amanda Putri, G. Ramadhan, Z. Alwildan, R. Afriansyah, and P. Manufaktur Negeri Bangka Belitung, "Perbandingan Kinerja Algoritma YOLO Dan RCNN Pada Deteksi Plat Nomor Kendaraan," 2023.
- [14] D. Hernikawati, "Analisis Dampak Pandemi COVID-19 terhadap Jumlah Kunjungan pada Situs E-Commerce di Indonesia Menggunakan Uji T Berpasangan," *Jurnal Studi Komunikasi dan Media*, vol. 25, no. 2, p. 191, Dec. 2021, doi: 10.31445/jskm.2021.4389.