

Perbandingan Algoritma YOLOv4 dan YOLOv4-tiny dalam Deteksi Korban Bencana Alam

Comparison of YOLOv4 and YOLOv4-tiny Algorithms in the Detection of Victims of Natural Disasters

M Adamu Islam Mashuri^{*1}, Faris Abdi El Hakim²

^{1,2}Universitas Negeri Surabaya, Jl. Ketintang, Ketintang, Kec. Gayungan, Surabaya, Jawa Timur 60231

¹mmashuri@unesa.ac.id, ²farishakim@unesa.ac.id

Format Kutipan: Mashuri, M.A.I & Hakim, F.A.E. (2024). Perbandingan Algoritma YOLOv4 dan YOLOv4-tiny dalam Deteksi Korban Bencana Alam. *Nusantara Journal of Science and Technology*, 1(2), hal. 30-37. <https://doi.org/10.69959/nujst.v1i2.119>

RIWAYAT ARTIKEL

Dikirim: 17 November 2024

Revisi Akhir: 28 November 2024

Diterbitkan: 30 November 2024

Tersedia Daring Sejak: 30 November 2024

KATA KUNCI

Kecerdasan Buatan

YOLOv4

YOLOv4-tiny

KEYWORDS

Artificial Intelligent

YOLOv4

YOLOv4-tiny

ABSTRAK

Saat ini teknologi kecerdasan buatan (*artificial intelligence*) banyak dibahas oleh para peneliti dan teknologi ini dapat membantu kita dalam kehidupan sehari-hari. Oleh karena itu terdapat banyak aplikasi dalam berbagai bidang, salah satunya adalah topik dalam penelitian ini yaitu deteksi korban bencana alam. Teknologi ini sangat dibutuhkan oleh tim penyelamat untuk mempercepat pencarian korban bencana alam karena alat yang saat ini digunakan hanya berupa alat berat sehingga membutuhkan waktu yang lama untuk mencari korban bencana alam. Dalam penelitian ini membandingkan kecepatan deteksi dan akurasi dalam mendeteksi korban bencana alam menggunakan algoritma You Only Look Once (YOLO) versi 4 dan YOLOv4-tiny. Kemudian melakukan pelatihan dengan parameter dan dataset yang sama tetapi menggunakan arsitektur yang berbeda. Dari hasil penelitian dapat disimpulkan bahwa algoritma YOLOv4-tiny lebih cepat dalam mendeteksi korban bencana dengan akurasi sebesar 75% sedangkan algoritma YOLOv4 membutuhkan waktu lebih lama untuk mendeteksi korban bencana namun memiliki akurasi sebesar 54%.

ABSTRACT

Currently, artificial intelligence technology is widely discussed by researchers and this technology can help us in our daily lives. So that there are many applications in various fields, one of which is the topic in our paper namely the detection of victims of natural disasters. This is really needed by the rescue team in speeding up the search for victims of natural disasters because the tools currently used are only heavy equipment, so it takes a long time to search for victims of natural disasters. In this paper we will compare the speed of detection and accuracy in detecting victims of natural disasters using the You Only Look Once (YOLO) version 4 and YOLOv4-tiny algorithms. We train with the same parameters and dataset but with a different architecture. From the results, we get the YOLOv4-tiny algorithm is faster in detecting disaster victims but has an accuracy of 75% whereas the YOLOv4 algorithm takes longer to detect victims of natural disasters but has an accuracy of 54%.

Artikel ini dapat diakses secara terbuka (*open access*) di bawah lisensi CC-BY-SA



PENDAHULUAN

Di era modern ini, perkembangan teknologi berlangsung dengan sangat pesat dan menghadirkan berbagai inovasi yang salah satunya adalah teknologi kecerdasan buatan atau *artificial intelligence* (AI). Teknologi ini mensimulasikan kecerdasan manusia yang diterapkan pada sistem komputer sehingga dapat menyelesaikan masalah di berbagai aspek serta sistem tersebut dirancang agar mampu belajar dan berpikir seperti manusia (Holzinger et al., 2019). Hal ini karena komputer telah dilatih untuk mengenali data sehingga dibutuhkan banyak data untuk membuat komputer dapat berpikir seperti manusia. Untuk melatih hal ini terdapat berbagai arsitektur yang dapat digunakan seperti yang digunakan dalam penelitian ini yaitu arsitektur YOLOv4 dan YOLOv4-tiny.

Arsitektur adalah topik yang sangat menarik dalam dunia AI karena arsitektur ini akan mempengaruhi proses deteksi dan kecepatan deteksi. Sehingga untuk mencapai sistem yang dapat mensimulasikan kecerdasan manusia perlu memperhatikan hal ini. Dengan membandingkan beberapa arsitektur, kita dapat memperoleh hasil yang paling optimal. Untuk mendukung hal tersebut diperlukan juga dataset untuk proses pelatihan dan pengujian.

Data pelatihan (*training*) ini akan digunakan dalam proses pelatihan yang diterapkan pada arsitektur yang telah dirancang. Selanjutnya hasil dari data pelatihan (*training*) ini dapat digunakan untuk melihat akurasi dengan menggunakan data pengujian. Dalam penelitian ini menggunakan dataset korban bencana alam. Hasil pelatihan dengan dataset ini akan diterapkan pada perangkat keras untuk

mendeteksi korban bencana alam. Perangkat keras ini digunakan untuk mendeteksi korban secara real-time. Karena saat bencana terjadi, akses menuju lokasi bencana akan sulit dilalui maka diperlukan perangkat keras yang dapat dipasang pada drone untuk mendeteksi korban setelah bencana terjadi. Dan untuk dapat membandingkan kinerja terbaik dari beberapa arsitektur ini perlu diperhatikan *hyperparameter* yang digunakan.

Dalam penelitian (Hartawan, Purboyo, & Setianingsih, 2019; Adamu, Zen, & Widyatra, 2022) dijelaskan perbandingan algoritma Convolutional Neural Network (CNN) dengan arsitektur YOLO, VGG-16, dan MobileNet. Arsitektur ini menggunakan dataset korban bencana alam dan memiliki akurasi terbaik. Namun permasalahan dalam penelitian ini adalah kita tidak mengetahui apakah arsitektur terbaik dapat memberikan hasil optimal saat diterapkan pada perangkat keras. Sebab, kita juga harus melihat kecepatan deteksi dan akurasi yang diperoleh setelah diterapkan pada perangkat keras. Oleh karena itu, analisis dari sisi perangkat keras perlu dilakukan untuk mendapatkan hasil yang maksimal.

Untuk mengatasi masalah ini, penelitian ini mengusulkan untuk membandingkan arsitektur YOLOv4 dan YOLOv4-tiny saat diterapkan pada perangkat keras. Pada penelitian ini akan melatih dan menerapkan arsitektur ini pada perangkat keras untuk mendeteksi, menghitung, dan mengklasifikasikan korban bencana alam. Dataset yang digunakan dibagi menjadi 2 kelas yaitu *Emergency* dan *Need Help*. Setiap arsitektur akan dilatih dengan dataset tersebut dan menggunakan *hyperparameter* yang sama. Hasil dari pelatihan ini akan dibandingkan berdasarkan akurasi dan kecepatan deteksi.

METODE PENELITIAN

Sistem dalam penelitian ini menggunakan dua arsitektur yang berbeda dimana arsitektur ini nantinya akan diterapkan pada perangkat keras yang dilengkapi dengan sensor termal. Penelitian sebelumnya telah menguji sensor termal ini untuk mendeteksi manusia (Lehr, Schluter, & Kruger, 2019). Saat ini mencari arsitektur terbaik untuk mendeteksi korban bencana alam menggunakan kamera dan Raspberry Pi. Arsitektur terbaik dari penelitian ini akan digunakan untuk penelitian selanjutnya.

Metodologi pada sistem ini dapat dilihat pada Gambar 1 bahwa sistem ini akan mendeteksi dan mengklasifikasikan korban bencana alam menggunakan arsitektur YOLOv4 dan YOLOv4-tiny. Sesuai dengan alur keseluruhan sistem bahwa pencarian dan pengumpulan dataset dilakukan terlebih dahulu. Dataset yang digunakan terdiri dari dua kelas dimana setiap kelas dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian. Pembagian ini sesuai dengan aturan umum yang biasanya digunakan sehingga dengan pembagian ini dapat melihat hasil akurasi dari dataset pengujian sebelum diterapkan pada perangkat keras.

Setelah itu dilakukan augmentasi untuk meningkatkan dataset. Proses augmentasi ini mencakup rotasi dan pemotongan sehingga untuk setiap data akan dihasilkan dua data baru. Setelah proses augmentasi selesai, arsitektur YOLOv4 dan YOLOv4-tiny diimplementasikan dengan beberapa *hyperparameter* untuk proses pelatihan. Biasanya proses pelatihan membutuhkan waktu yang cukup lama untuk menghasilkan data terbaik. Setelah file hasil pelatihan diperoleh maka file tersebut dapat diterapkan pada perangkat keras untuk analisis. Tata letak keseluruhan metodologi sistem yang digunakan dalam penelitian ini ditunjukkan pada Gambar 1.



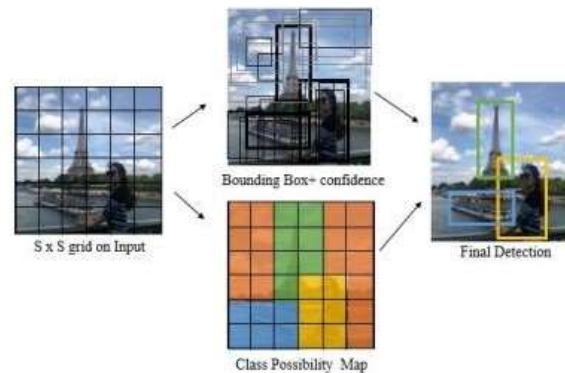
Gambar 1. Metodologi Sistem

Untuk mendapatkan hasil terbaik dan sesuai dengan tujuan utama, diperlukan beberapa teori pendukung, yaitu:

A. Algoritma You Only Look Once (YOLO)

YOLO adalah metode untuk mendeteksi objek yang membutuhkan input untuk dapat mendeteksi setiap objek dalam gambar input. YOLO memproses gambar secara real-time pada 45 frame per detik. YOLO dapat mewakili objek-objek yang sangat umum (Hutauruk, Matulatan, & Hayaty, 2020; Redmon & Farhadi, 2018). Sistem YOLO membagi gambar input menjadi grid $S \times S$. Setiap sel grid memprediksi bounding box dan memprediksi nilainya. Nilai ini akan digunakan untuk mengetahui seberapa akurat model dalam mendeteksi objek dalam kotak tersebut. Setiap bounding box terdiri dari 5 prediksi: x , y , w , h , dan confidence. Koordinat (x, y) mewakili pusat grid relatif terhadap batas

sel grid. (w, h) atau lebar dan tinggi mewakili pusat kotak relatif terhadap gambar. Terakhir adalah confidence yang mewakili *Intersection over Union* (IoU) antara kotak prediksi dan kotak *ground truth* (Khairunnas, Mulyanto, & Zaini, 2021). Proses ini dapat dilihat pada Gambar 2.

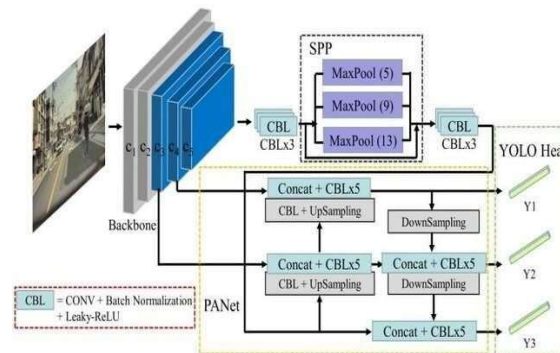


Gambar 2. Urutan Deteksi Objek pada YOLO

Secara umum, proses deteksi YOLO seperti yang ditunjukkan di atas namun dalam YOLO terdapat beberapa versi dari sistem pemodelannya. Salah satu contohnya adalah dalam penelitian ini menggunakan arsitektur YOLOv4 dan YOLOv4-tiny.

1. YOLOv4

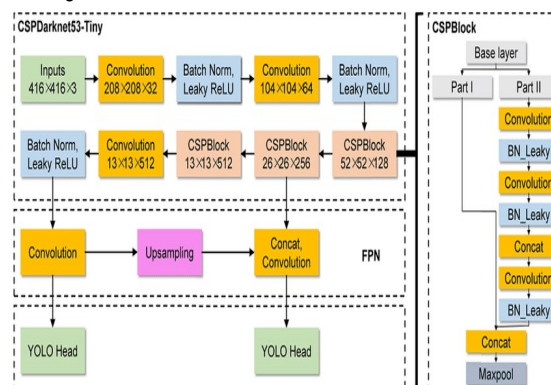
Untuk arsitektur YOLOv4 itu sendiri, menggunakan Backbone, Neck, dan Head. Pada Backbone menggunakan ResNet untuk mengekstraksi fitur. Kemudian beberapa lapisan diterapkan untuk mengekstraksi fitur, semakin banyak lapisan dalam arsitektur dapat menghasilkan tingkat fitur yang berbeda dengan semantik yang lebih tinggi (Jeong-ah, Ju-Yeong, & Se-ho, 2020), dan neck berfungsi sebagai lapisan tambahan yang digunakan untuk mengekstraksi peta fitur dari berbagai tahap pada backbone. Head digunakan untuk mendeteksi objek dan untuk *bounding box*. Pada head menggunakan 2 kelas yaitu emergency dan need help.



Gambar 3. Arsitektur YOLOv4

2. Arsitektur YOLOv4-tiny

YOLOv4-tiny adalah versi kecil dari YOLOv4 dimana YOLOv4-tiny memiliki jaringan yang lebih sederhana dibandingkan dengan YOLOv4. Seluruh jaringan YOLOv4 memiliki sekitar 60 juta parameter sedangkan YOLOv4-tiny hanya memiliki 6 juta (Kun & Yan, 2021). Jadi arsitektur ini dapat melakukan pelatihan lebih cepat dan melakukan deteksi lebih cepat. Karena arsitektur ini hanya memiliki 2 YOLO Heads sedangkan YOLOv4 memiliki 3 YOLO Heads.



Gambar 4. Arsitektur YOLOv4-tiny

B. Software

Dalam penelitian ini menggunakan layanan Google Colaboratory untuk melatih dataset sebelum diterapkan pada perangkat keras. Perangkat lunak ini menyediakan "Jupyter Notebook" tanpa server untuk pengembangan interaktif dan disediakan secara gratis (Bisong, 2019). Infrastruktur Google Colab memiliki spesifikasi berupa prosesor Xeon dengan hyper-thread single-core @ 2.3GHz CPU, GPU yang didukung oleh Nvidia Tesla K80 dengan 296 inti CUDA, 12GB DDR5 VRAM, 12,6GB RAM, dan disk sebesar 29GB. Selain itu juga menggunakan VNC Viewer untuk Raspberry Pi (Rohit & Tank, 2018).

C. Desain Sistem

Alat yang digunakan terdiri dari Raspberry Pi 4 4GB dan Raspberry Pi Camera 16MP IMX519. Raspberry Pi 4 4GB yang dipilih memiliki spesifikasi tertinggi yang akan mempercepat proses deteksi. Selain itu, Raspberry Pi Camera 16MP akan digunakan sebagai input awal yang akan diproses oleh algoritma yang telah dilatih. Pemilihan ini mempertimbangkan kualitas gambar dan bobot perangkat keras yang akan dipasang pada drone.



Gambar 5. Alat yang digunakan

D. Klasifikasi Data dan Dataset

Dataset yang digunakan dalam penelitian ini adalah dataset untuk korban bencana alam. Dataset ini diperoleh dari IEEE Dataset, dataset Fall dari Kaggle serta dikombinasikan dengan dataset kustom yang dikumpulkan. Kemudian membagi dataset ini menjadi 2 kelas yang berbeda yaitu kelas *Emergency* dan kelas *Need Help*. Pada kelas *Emergency* mendefinisikan bahwa korban berada dalam kondisi cedera serius dan membutuhkan pertolongan segera sedangkan pada kelas *Need Help* mendefinisikan bahwa korban mengalami cedera ringan. Dataset yang digunakan untuk eksperimen dapat dilihat pada tabel di bawah ini.

Tabel 1. Dataset pada korban bencana alam

Nama label	Data training	Data testing
Emergency	692	173
Need Help	692	173

E. Hyperparameter

Proses pelatihan dengan *hyperparameter* ini akan sama untuk setiap algoritma yang dibandingkan. Beberapa *hyperparameter* yang digunakan adalah sebagai berikut:

Tabel 2. Dataset pada korban bencana alam

Hyperparameter	Nilai
Learning rate	0,0001
Epoch	6
Max batch	6000
Input Size	224, 224

Dari *hyperparameter* diatas dapat disimpulkan bahwa semakin kecil learning rate maka hasil pelatihan akan semakin optimal. Kemudian menggunakan max batch sebanyak 6000 dengan 1000 steps dan 6 epoch akan menghasilkan data hasil pelatihan setiap 1000 iterasi. Karena menggunakan 6 epoch dalam 1 epoch akan dilakukan 1000 iterasi sehingga data hasil pelatihan yang dihasilkan berjumlah 6 data. Data hasil pelatihan terbaik akan dipilih nantinya. Ukuran input adalah ukuran gambar yang akan dilatih.

Data hasil pelatihan ini diperoleh dengan menerapkan sebuah arsitektur dimana dalam penelitian ini akan memperoleh data hasil pelatihan dengan membandingkan arsitektur:

1. Arsitektur YOLOv4
2. Arsitektur YOLOv4-tiny

Dalam perbandingan arsitektur yang diusulkan terdapat 2 jenis pembandingan untuk mendapatkan hasil yang optimal:

- **Evaluasi Model:** Hal ini dilakukan untuk melihat hasil dari pelatihan yang telah dilakukan. Dalam penelitian ini digunakan pendekatan confusion matrix yang menampilkan hasil precision, accuracy, recall, F1-score, dan mAP.
- **Kecepatan Deteksi:** Hal ini sangat berguna jika diterapkan untuk mendeteksi objek darurat.

Selanjutnya pengujian dilakukan dengan memasukkan data dengan hasil terbaik ke dalam program untuk mengevaluasi model. Evaluasi model ini menggunakan pendekatan confusion matrix sehingga diperoleh beberapa data yaitu:

1. Akurasi merupakan prediksi yang benar terhadap seluruh data. Nilai ini akan digunakan untuk melihat arsitektur mana yang memiliki akurasi lebih tinggi. Akurasi dapat dihitung dengan rumus pada persamaan 1.

$$Akurasi = \frac{TP+TN}{TP+FN+FP+TN} \quad (1)$$

2. *Precision* merupakan matrik yang digunakan untuk mengevaluasi tingkat keakuratan prediksi positif suatu model terhadap total keseluruhan prediksi positif yang dihasilkan.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

3. *Recall* disebut juga sensitivitas yaitu perbandingan antara prediksi benar positif dengan keseluruhan data positif. *Recall* dapat dihitung secara manual dengan menggunakan rumus.

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

4. *F1-Score* adalah perbandingan rata-rata harmonis antara *precision* dan *recall*. Jika dituliskan dalam bentuk rumus, adalah sebagai berikut:

$$F1-Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)} \quad (4)$$

HASIL PENELITIAN DAN PEMBAHASAN

Untuk simulasi pada penelitian ini menggunakan data pelatihan terbaik dan kemudian menerapkannya langsung pada perangkat yang terdiri dari Raspberry Pi 4 dengan RAM 4GB dan Pi Camera 16MP. Area pengujian dilakukan di dalam rumah. Kemudian membandingkan kinerja arsitektur YOLOv4 dan YOLOv4-tiny. Simulasi ini mencakup hasil pelatihan, hasil deteksi menggunakan input dari video secara real-time, perhitungan jumlah objek yang terdeteksi, dan yang terakhir adalah pengujian kecepatan deteksi. Simulasi ini mencakup hasil pelatihan, hasil deteksi menggunakan input video secara real-time, perhitungan jumlah objek yang terdeteksi, dan pengujian kecepatan deteksi.

A. Hasil *training* menggunakan arsitektur YOLOv4 dan YOLOv4-tiny

Proses *training* ini dilakukan menggunakan Google Colab dengan beberapa *hyperparameter* seperti yang ditunjukkan pada Tabel 3. Setelah *training* akan mendapatkan beberapa data dari hasil *training*. Data terbaik akan digunakan dalam program untuk mendeteksi korban bencana alam secara real-time. Proses pelatihan ini di Google Colab agar tidak membebani Raspberry Pi 4 jika pelatihan dilakukan pada perangkat keras. Hasil pelatihan yang diperoleh adalah sebagai berikut:

Tabel 3. Hasil Evaluasi menggunakan YOLOv4 dan YOLOv4-tiny

Matriks	YOLOv4	YOLOv4-tiny
Akurasi	92,29%	75%
Precision	86%	71%
Recall	88%	73%
F1-Score	87%	73%
mAp	92,20%	75%

Dari hasil yang diperoleh dapat disimpulkan bahwa matriks YOLOv4 lebih optimal dibandingkan YOLOv4-tiny, hal ini terjadi karena YOLOv4-tiny hanya menggunakan beberapa parameter.

B. Pengujian dengan input video

Uji ini dilakukan untuk melihat hasil deteksi dan perhitungan input berupa video. Video ini menunjukkan orang yang berjalan yang direkam oleh CCTV seperti yang ditunjukkan pada Gambar 6. Dari video ini akan dilakukan analisis apakah program sudah dapat mendeteksi dan menghitung jumlah orang yang terdeteksi.

:



Gambar 6. Hasil deteksi dengan input video

Tabel 4. Hasil testing dengan input video

No	Jumlah Aktual	YOLOv4	YOLOv4-tiny
1	10	9	7
2	9	9	7
3	13	9	8
4	12	8	7
5	11	9	8
6	13	10	8
7	14	10	8
8	15	9	7
9	14	11	9
10	14	11	8
Total	125	95	77

Dari hasil yang diperoleh pada Tabel V dapat disimpulkan bahwa deteksi dengan YOLOv4 lebih baik dibandingkan dengan YOLOv4-tiny, karena tabel menunjukkan bahwa ada beberapa objek yang tidak terdeteksi menggunakan YOLOv4-tiny. Dari tabel tersebut dapat melakukan analisis *confusion matrix* melalui perhitungan manual seperti di bawah ini.

- $Recall = \frac{TP}{TP+FN} = \frac{95}{95} = 1$
- $Precision = \frac{TP}{TP+fp} = \frac{95}{125} = 0.76$
- $F1-Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)} = \frac{1,52}{1,76} = 0,86$
- $Akurasi = \frac{TP}{TP+FN+FP} \times 100\% = \frac{95}{125} \times 100\% = 76\%$

Tabel 5. Confusion Matrix

Matriks	Nilai rata-rata	
	YOLOv4	YOLOv4-tiny
Akurasi	92,29%	75%
Precision	86%	71%
Recall	88%	73%
F1-Score	87%	73%

Dan berdasarkan data pada Tabel VI dapat menghitung akurasi dari uji dengan input video. Hasil akurasi yang diperoleh adalah 76% untuk YOLOv4 dan 61% untuk YOLOv4-tiny. Dari hasil tersebut dapat dilihat bahwa YOLOv4 menghasilkan hasil yang baik. Dalam proses deteksi namun hasil akurasi yang diperoleh dari kedua arsitektur tersebut masih belum optimal karena dalam kondisi tertentu seperti orang yang membawa anak kecil dan orang yang setengah tubuhnya tidak terlihat di kamera maka sistem tidak berfungsi secara optimal.

C. Pengujian deteksi dengan kamera secara *real-time*

Uji ini dilakukan untuk melihat hasil deteksi dan perhitungan input berupa video. Video ini menunjukkan orang yang berjalan direkam oleh CCTV seperti yang ditunjukkan pada Gambar 6. Dari video ini akan dilakukan analisis apakah program sudah dapat mendeteksi dan menghitung jumlah orang yang terdeteksi.

Uji ini dilakukan untuk mengetahui apakah sistem dapat mendeteksi objek secara real-time menggunakan Raspberry Pi 4 dan kamera web. Untuk posisi kamera itu sendiri ditempatkan pada ketinggian 2 meter dan mengatur sudut kamera pada 70° ke bawah. Kemudian akan menempatkan objek dengan beberapa posisi dan jarak dari korban ke kamera. Hasil uji adalah sebagai berikut.

Tabel 6. Hasil pengujian menggunakan kamera secara *real-time*

Test	Jumlah korban dalam frame	Jumlah korban terdeteksi	
		YOLOv4	YOLOv4-tiny
1	2 NH	2 NH	1 NH
2	1 NH, 1 EM	1 NH, 1 EM	1 NH, 1 EM
3	2 EM	2 EM	1 EM
4	3 NH	3 NH	2 NH
5	2 NH, 1 EM	2 NH	-
6	1 NH, 2 EM	1 NH, 2 EM	1 NH, 1 EM
7	5 NH	2 NH	1 NH
8	4 NH, 1 EM	3 NH	3 NH
9	4 NH, 1 EM	3 NH, 1 EM	2 NH
10	4 NH, 1 EM	3 NH, 1 EM	3 NH, 1 EM

Catatan : EM = Emergency, NH = Need Help

Dari tabel di atas dapat disimpulkan bahwa YOLOv4 lebih unggul dalam mendeteksi korban bencana alam secara real-time. Sementara itu YOLOv4-tiny terlihat sangat lemah dalam proses deteksi pada setiap pengujian. Setelah mendapatkan tabel di atas, kita dapat menganalisis hasil dari *confusion matrix*.

Tabel 7. Hasil kalkulasi *confussion matrix*

Matriks	Nilai rata-rata	
	YOLOv4	YOLOv4-tiny
Akurasi	75%	54%
Precision	77%	54%
Recall	95%	95%
F1-Score	85%	68%

Dari hasil pengujian yang dilakukan menggunakan webcam dapat dilihat bahwa akurasi terbaik diperoleh dari YOLOv4 yaitu sebesar 75%. Namun dari hasil deteksi yang diperoleh dapat disimpulkan bahwa kamera tidak mampu mendeteksi jika korban terhalang oleh korban lain sehingga akurasi yang didapat masih belum optimal.

D. Pengujian kecepatan deteksi

Seperti yang terlihat pada Tabel VI bahwa YOLOv4-tiny memiliki rata-rata waktu klasifikasi sebesar 8,6 detik dengan sekitar 2 detik lebih cepat dibandingkan YOLOv4. Hal ini menunjukkan bahwa YOLOv4-tiny sangat cepat dan mengungguli YOLOv4 dalam uji kecepatan. Namun YOLOv4-tiny tidak mampu mendeteksi objek meskipun terdapat objek yang terlihat jelas dalam gambar seperti yang ditunjukkan pada Gambar 7. Sementara itu YOLOv4 meskipun jauh lebih lambat dengan hanya 1 frame per detik akan tetapi mampu mengklasifikasikan dengan benar ke dalam kelas Need Help yaitu dengan akurasi sebesar 85%.



Gambar 7. Hasil klasifikasi gambar real-time pada skenario untuk YOLOv4



Gambar 8. Hasil klasifikasi gambar real-time pada skenario untuk YOLOv4-tiny

Tabel 8. Hasil kalkulasi *confussion matrix*

Matriks	Nilai rata-rata	
	YOLOv4	YOLOv4-tiny
Waktu klasifikasi	10.6 s	8.6 s
Klasifikasi objek	2	0
Klasifikasi objek yang benar	2	0
Akurasi	85%	0

SIMPULAN DAN SARAN

Dalam mendeteksi objek menggunakan Convolutional Neural Network yang diimplementasikan ke dalam perangkat keras untuk mendeteksi diperlukan perbandingan untuk menentukan algoritma dan arsitektur yang paling optimal dalam mendeteksi objek. Hal ini menjadi faktor penting karena perangkat mungkin tidak bekerja secara optimal jika langsung menggunakan algoritma yang diinginkan tanpa analisis terlebih dahulu. Oleh karena itu diperlukan perbandingan untuk melakukan penelitian lebih lanjut. Dari hasil perbandingan pada penelitian ini dapat disimpulkan bahwa YOLOv4 lebih baik dalam mendeteksi objek tetapi proses deteksinya lebih lambat dibandingkan YOLOv4-tiny. Pada deteksi menggunakan kamera secara real-time, YOLOv4 mencapai akurasi sebesar 75% sedangkan YOLOv4-tiny hanya mencapai akurasi sebesar 54%. Untuk perbandingan kecepatan deteksi menggunakan YOLOv4-tiny lebih cepat 2 detik dibandingkan YOLOv4 namun deteksi objek menggunakan YOLOv4 lebih baik daripada YOLOv4-tiny.

DAFTAR PUSTAKA

- Adamu, M., Zen, M., & Widyatra, R. (2022). Smart victims detection in natural disaster using deep learning. In Proceedings of the International Electronics Symposium (IES), August 2022.
- Bisong, E. (2019). Training a neural network.
- Hartawan, D. R., Purboyo, T. W., & Setianingsih, C. (2019). Disaster victims detection system using convolutional neural network (CNN) method. In Proceedings of the International Conference on Industry 4.0, Artificial Intelligence, and Communication Technology (IAICT).
- Holzinger, A., Langs, G., Denk, H., Zatloukal, K., & Muller, H. (2019). Causability and explainability of artificial intelligence in medicine. WIREs Data Mining and Knowledge Discovery, 9. <https://doi.org/10.1002/widm.1305>
- Hutauruk, J. S. W., Matulatan, T., & Hayaty, N. (2020). Vehicle detection in real time using the Android-based YOLO method. Journal Name, 9(1), 8–14.
- Jeong-ah, K., Ju-Yeong, S., & Se-ho, P. (2020). Comparison of Faster-RCNN, YOLO, and SSD for real-time vehicle type recognition. In Proceedings of the IEEE International Conference on Consumer Electronics – Asia (ICCE-Asia).
- Khairunnas, M., Mulyanto, E., & Zaini, A. (2021). Making a human object detection module using the YOLO method for mobile robots. Jurnal Teknik ITS, 10.
- Kun, J., & Yan, C. (2021). SAR image ship detection based on deep learning. In Proceedings of the International Conference on Computer Engineering and Intelligent Control (ICCEIC), March 2021.
- Lapian, A. K., Sompie, S., & Manembu, P. (2021). You only look once (YOLO) implementation for signature pattern classification. Jurnal Teknik Informatika, 16(3), 337–346.
- Lehr, J., Schluter, M., & Kruger, J. (2019). Classification of similar objects of different sizes using a reference object by means of convolutional neural network. In Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA).
- Malhotra, P., & Garg, E. (2020). Object detection techniques: A comparison. In Proceedings of the International Conference on Smart Structures and Systems (ICSSS).
- Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement.
- Rohit, S. L., & Tank, B. V. (2018). IoT-based health monitoring system using Raspberry Pi – Review. In Proceedings of the International Conference on Inventive Communication and Computational Technologies (ICICCT).