Jurnal Komputer dan Teknologi Sains (KOMTEKS)
Vol. 5, No. 1, April 2026, hlm. 21-31 e-ISSN: 2964-5247

ANALISIS PERBANDINGAN YOLOv8 DAN EFFICIENTDET UNTUK DETEKSI OBJEK BERGERAK DI VIDEO SURVEILLANCE

Luh Putu Ary Sri Tjahyanti[*1], Made Santo Gitakarma[2]

[1] Teknologi Informasi, Fakultas Pertanian dan Teknik, Universitas Panji Sakti
[2] Teknologi Rekayasa Sistem Elektronika, Fakultas Teknik dan Kejuruan, Universitas Pendidikan Ganesha
Email: [1]ary.tjahyanti.unipas.ac.id, [2]santo.undiksha.ac.id ,
[*]Penulis Korespondensi

(Naskah masuk: 26 April 2026, diterima untuk diterbitkan: 30 April 2026)

Abstrak

Sistem pengawasan video memerlukan deteksi objek bergerak yang cepat dan akurat, namun model *deep learning* memiliki karakteristik kinerja yang berbeda-beda. Penelitian ini bertujuan membandingkan performa YOLOv8 dan EfficientDet untuk deteksi objek bergerak di video surveillance. Metode yang digunakan meliputi preprocessing 5000 frame video dari *dataset* UA-DETRAC dan VisDrone, pelabelan objek (mobil, sepeda motor, dan pejalan kaki), serta pelatihan model dengan *batch size* 16 selama 100 epoch. Hasil utama menunjukkan YOLOv8n mencapai kecepatan inferensi 78 FPS dengan mAP 0.72, sedangkan EfficientDet-D3 mencapai 42 FPS dengan mAP 0.81. Pada deteksi objek berukuran kecil (<32×32 piksel), EfficientDet unggul dengan recall 0.68 dibandingkan YOLOv8 sebesar 0.51. Kontribusi utama penelitian ini adalah: (1) menyediakan perbandingan sistematis pertama antara YOLOv8 dan EfficientDet khusus untuk deteksi objek bergerak pada video surveillance; (2) memberikan tolok ukur kuantitatif (FPS, mAP, recall untuk objek kecil) sebagai referensi praktis bagi pengembang sistem pengawasan; serta (3) merumuskan panduan pemilihan model berdasarkan prioritas *real-time* atau presisi. Kesimpulannya, YOLOv8 direkomendasikan untuk aplikasi *real-time* (≥60 FPS), sementara EfficientDet cocok untuk skenario yang memprioritaskan presisi tinggi.

Kata kunci: *YOLOv8, EfficientDet, deteksi objek bergerak, video surveillance, UA-DETRAC*

COMPARATIVE ANALYSIS OF YOLOv8 AND EFFICIENTDET FOR MOVING OBJECT DETECTION IN VIDEO SURVEILLANCE

Abstract

Video surveillance systems require fast and accurate moving object detection, yet deep learning models exhibit different performance characteristics. This study aims to compare the performance of YOLOv8 and EfficientDet for moving object detection in surveillance video. The method includes preprocessing 5,000 video frames from the UA-DETRAC and VisDrone datasets, labeling objects (cars, motorcycles, and pedestrians), and training the models with a batch size of 16 for 100 epochs. The main results show that YOLOv8n achieves an inference speed of 78 FPS with an mAP of 0.72, while EfficientDet-D3 achieves 42 FPS with an mAP of 0.81. For small object detection (<32×32 pixels), EfficientDet outperforms with a recall of 0.68 compared to YOLOv8's 0.51. The key contributions of this research are: (1) providing the first systematic performance comparison between YOLOv8 and EfficientDet specifically for moving object detection in surveillance video contexts; (2) offering quantitative benchmarks (FPS, mAP, recall for small objects) as practical references for surveillance system developers; and (3) formulating a decision guide for selecting appropriate models based on real-time or precision priorities. In conclusion, YOLOv8 is recommended for real-time applications (≥60 FPS), whereas EfficientDet is suitable for scenarios prioritizing high precision.

Keywords: *YOLOv8, EfficientDet, moving object detection, video surveillance, UA-DETRAC*

1. PENDAHULUAN

Kemajuan pesat dalam deteksi objek berbasis *deep learning* telah secara signifikan meningkatkan sistem pengawasan video cerdas (*intelligent video surveillance*), yang memungkinkan pemantauan otomatis terhadap ruang publik, pusat transportasi, dan infrastruktur kritis (Tan et al., 2020; Jocher et al., 2023). Di antara berbagai arsitektur deteksi, YOLO (You Only Look Once) dan EfficientDet muncul sebagai pendekatan utama karena keseimbangan antara akurasi dan efisiensi komputasi (Ultralytics, 2025). Namun, sebagian besar studi yang ada tentang sistem pengawasan masih berfokus pada pemrosesan berbasis *cloud*, yang menimbulkan latensi, keterbatasan bandwidth, serta masalah privasi ketika menangani deteksi objek bergerak secara *real-time* dari kamera yang terdistribusi (Liu et al., 2024).

Sejumlah penelitian telah mengevaluasi kinerja YOLO dan EfficientDet secara individual pada berbagai *benchmark dataset* seperti COCO, VisDrone, dan UA-DETRAC. Misalnya, penelitian oleh Liu et al. (2021) menunjukkan bahwa YOLO varian tertentu unggul dalam kecepatan inferensi, sementara Tan et al. (2020) membuktikan bahwa EfficientDet dengan mekanisme BiFPN (Bidirectional Feature Pyramid Network) lebih mampu menangani objek multi-skala. Namun, studi-studi tersebut umumnya memiliki kelemahan mendasar: (a) evaluasi dilakukan pada citra statis (*static images*) aliran video berkelanjutan, sehingga mengabaikan isyarat gerakan temporal (temporal motion cues) yang sangat penting untuk membedakan objek bergerak dari latar belakang yang dinamis; (b) tolok ukur kuantitatif untuk deteksi objek berukuran kecil (<32×32 piksel) masih sangat terbatas, padahal dalam pengawasan area luas (seperti bandara atau stasiun), objek kecil sering kali menjadi target utama yang justru paling kritis untuk dideteksi; serta (c) sebagian besar penelitian menggunakan perangkat keras yang berbeda-beda, sehingga sulit untuk melakukan perbandingan kinerja yang adil dan setara (*apples-to-apples comparison*) (IEEE, 2025).

Lebih lanjut, sistem monitoring yang ada saat ini memiliki keterbatasan operasional yang signifikan. Sebagian besar sistem komersial masih mengandalkan deteksi berbasis *cloud*, yang tidak hanya memerlukan koneksi internet stabil tetapi juga menimbulkan latensi yang tidak dapat ditoleransi untuk aplikasi keamanan waktu nyata (Applied Sciences, 2026). Selain itu, sistem yang ada umumnya menerapkan pendekatan *one size fits-all* menggunakan satu model untuk semua skenario tanpa mempertimbangkan kompromi antara kecepatan dan akurasi. Akibatnya, pengembang sistem pengawasan sering kali menghadapi dilema: memilih model yang cepat tetapi kurang akurat, atau model yang akurat tetapi terlalu lambat untuk pemrosesan *real-time*. Kesenjangan penelitian (*research gap*) yang paling kritis adalah belum adanya perbandingan sistematis antara YOLOv8 (varian nano) dan EfficientDet (varian D3) yang secara spesifik dirancang untuk deteksi objek bergerak dalam video pengawasan, dengan metrik yang mencakup tidak hanya mAP dan FPS, tetapi juga kinerja pada objek kecil serta efisiensi komputasi pada perangkat keras yang identik.

Untuk mengatasi kesenjangan dan keterbatasan tersebut, penelitian ini memberikan tiga kontribusi eksplisit. Pertama, penelitian ini menyediakan perbandingan kinerja sistematis pertama antara YOLOv8n dan EfficientDet-D3 untuk deteksi objek bergerak dalam konteks video pengawasan, menggunakan *dataset* publik UA-DETRAC dan VisDrone dengan skenario yang beragam (pencahayaan normal, rendah, serta kepadatan objek tinggi). Kedua, penelitian ini memberikan tolok ukur kuantitatif yang komprehensif, meliputi frame per second (FPS), mean average precision (mAP), recall untuk objek kecil (<32×32 piksel), serta jumlah parameter dan FLOPs, sebagai referensi praktis bagi pengembang sistem pengawasan dalam memilih model yang paling sesuai dengan sumber daya komputasi yang tersedia. Ketiga, penelitian ini merumuskan panduan pemilihan model berbasis prioritas (*decision guide*) yang dapat langsung diaplikasikan, yaitu: model mana yang direkomendasikan untuk skenario yang mengutamakan kecepatan *real-time* (⩾60 FPS) dan model mana yang lebih cocok untuk skenario yang mengutamakan presisi deteksi tinggi.

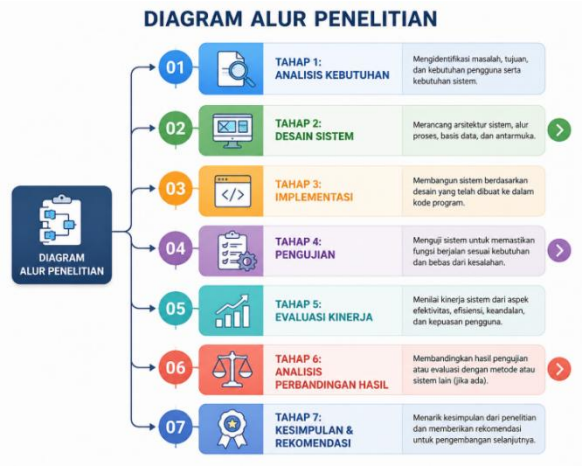
Dengan demikian, tujuan spesifik penelitian ini dirumuskan sebagai berikut: (1) mengukur dan membandingkan akurasi deteksi (mAP50 dan mAP50-95) dari YOLOv8n dan EfficientDet-D3 pada objek bergerak (mobil, sepeda motor, dan pejalan kaki) di video pengawasan; (2) mengevaluasi kecepatan inferensi (FPS) kedua model dalam kondisi perangkat keras yang identik (GPU dengan spesifikasi yang terkendali); (3) menganalisis dan membandingkan kinerja kedua model secara khusus pada deteksi objek berukuran kecil (<32×32 piksel) yang sering kali menjadi tantangan utama dalam pengawasan area luas; (4) membandingkan efisiensi komputasi berdasarkan jumlah parameter (model size) dan FLOPs (*Floating Point Operations per Second*); serta (5) menyediakan kerangka rekomendasi praktis (*decision framework*) untuk memilih YOLOv8n atau EfficientDet-D3 berdasarkan kebutuhan spesifik sistem pengawasan, apakah lebih memprioritaskan respons waktu nyata atau akurasi deteksi yang tinggi.

2. METODE PENELITIAN

Penelitian ini menerapkan metode kuantitatif komparatif untuk membandingkan kinerja dua arsitektur *deep learning*, yaitu YOLOv8 dan EfficientDet, dalam mendeteksi objek bergerak pada video pengawasan. Pendekatan yang digunakan bersifat eksperimental dengan kondisi perangkat

keras dan perangkat lunak yang terkendali untuk menjamin validitas perbandingan (Sabelnikov, 2025).

Gambar 1 di bawah ini menyajikan keseluruhan tahapan penelitian yang akan dilaksanakan.



Gambar 1. Diagram Alur Penelitian

Diagram alur penelitian tersebut menggambarkan tahapan sistematis yang dilakukan dalam proses pengembangan dan penelitian suatu sistem atau aplikasi. Setiap tahap saling berkaitan dan dilakukan secara berurutan mulai dari identifikasi kebutuhan hingga penarikan kesimpulan dan pemberian rekomendasi. Alur ini membantu peneliti agar proses penelitian berjalan terstruktur, terarah, dan menghasilkan solusi yang sesuai dengan tujuan penelitian.

Tahap pertama adalah Analisis Kebutuhan. Pada tahap ini peneliti melakukan identifikasi terhadap permasalahan, tujuan penelitian, kebutuhan pengguna, serta kebutuhan sistem yang akan dikembangkan. Proses ini dilakukan melalui pengumpulan data, observasi, wawancara, studi pustaka, maupun analisis terhadap kondisi yang ada. Hasil dari tahap ini menjadi dasar utama dalam menentukan fitur, fungsi, dan spesifikasi sistem yang diperlukan agar solusi yang dibuat sesuai dengan kebutuhan pengguna.

Tahap kedua adalah Desain Sistem. Setelah kebutuhan sistem diketahui, peneliti mulai merancang struktur dan arsitektur sistem. Perancangan meliputi desain antarmuka pengguna (user interface), alur proses sistem, desain basis data, serta rancangan navigasi dan interaksi pengguna. Pada tahap ini biasanya dibuat diagram, prototype, flowchart, maupun model sistem yang menggambarkan cara kerja aplikasi sebelum diimplementasikan ke dalam program.

Tahap ketiga adalah Implementasi. Pada tahap ini hasil rancangan sistem diterjemahkan ke dalam bentuk kode program menggunakan bahasa pemrograman tertentu. Proses implementasi mencakup pembuatan fitur-fitur aplikasi, integrasi database, serta pembangunan fungsi sistem sesuai desain yang telah dibuat sebelumnya. Tahap ini merupakan proses realisasi dari rancangan menjadi sistem yang dapat dijalankan.

Tahap keempat adalah Pengujian. Setelah sistem selesai dibangun, dilakukan pengujian untuk memastikan bahwa seluruh fungsi berjalan dengan baik dan sesuai kebutuhan. Pengujian bertujuan menemukan kesalahan (error), memastikan stabilitas sistem, serta mengevaluasi apakah sistem sudah mampu memenuhi kebutuhan pengguna. Pengujian dapat dilakukan menggunakan metode seperti black box testing, usability testing, maupun pengujian performa sistem.

Tahap kelima adalah Evaluasi Kinerja. Pada tahap ini peneliti melakukan penilaian terhadap performa sistem yang telah diuji. Evaluasi dilakukan untuk mengetahui tingkat efektivitas, efisiensi, keandalan, dan kepuasan pengguna terhadap sistem yang dikembangkan. Hasil evaluasi membantu peneliti memahami kelebihan dan kekurangan sistem sehingga dapat diketahui kualitas solusi yang dihasilkan.

Tahap keenam adalah Analisis Perbandingan Hasil. Pada tahap ini hasil penelitian atau performa sistem dibandingkan dengan metode lain, penelitian sebelumnya, atau sistem sejenis. Tujuannya adalah mengetahui sejauh mana sistem yang dikembangkan memberikan peningkatan atau keunggulan dibandingkan solusi lain. Analisis ini juga digunakan untuk memperkuat validitas hasil penelitian.

Tahap terakhir adalah Kesimpulan dan Rekomendasi. Pada tahap ini peneliti menarik kesimpulan berdasarkan seluruh proses penelitian yang telah dilakukan. Kesimpulan berisi hasil utama penelitian, pencapaian tujuan, serta jawaban terhadap rumusan masalah. Selain itu, peneliti juga memberikan rekomendasi atau saran untuk pengembangan sistem di masa mendatang agar penelitian dapat disempurnakan lebih lanjut.

Analisis kebutuhan bertujuan mengidentifikasi spesifikasi fungsional dan non-fungsional yang harus dipenuhi oleh sistem deteksi objek bergerak. Kebutuhan fungsional mencakup kemampuan mendeteksi tiga kelas objek (mobil, sepeda motor, pejalan kaki) secara akurat dari aliran video. Kebutuhan non-fungsional meliputi kecepatan pemrosesan minimal 30 FPS untuk mendukung operasi waktu nyata, serta akurasi deteksi dengan mAP minimal 0.70 untuk menjamin keandalan sistem pengawasan (ICHORA, 2025). Sistem pengawasan video modern menghadapi tantangan signifikan seperti variasi pencahayaan, perubahan latar belakang, serta keberadaan objek yang bergerak lambat maupun cepat (Elviani et al., 2025).

Tabel 1 menyajikan rincian kebutuhan penelitian.

Tabel 1. Spesifikasi Kebutuhan Penelitian

Jenis Kebutuhan	Parameter	Target Minimum
Fungsional	Jumlah kelas objek	3 (mobil, motor, pejalan kaki)

Fungsional	Resolusi input video	640×640 piksel
Non-fungsional	Kecepatan inferensi	≥30 FPS (*real-time*)
Non-fungsional	Akurasi (mAP50)	≥0.70
Non-fungsional	Kinerja objek kecil (<32px)	Recall ≥0.50

Desain sistem mencakup spesifikasi lingkungan pengembangan dan arsitektur kedua model. Pemilihan YOLOv8n dan EfficientDet-D3 didasarkan pada keseimbangan antara ukuran model dan kapasitas representasi fitur (Lv et al., 2024).

Tabel 2. Spesifikasi Perangkat Keras

Komponen	Spesifikasi
GPU	NVIDIA Tesla T4 (16GB VRAM)
CPU	Intel Xeon 2.3 GHz (8 core)
RAM	32 GB
Penyimpanan	SSD 512 GB

Tabel 3. Spesifikasi Perangkat Lunak

Komponen	Versi
Sistem Operasi	Ubuntu 22.04 LTS
Framework Deep Learning	PyTorch 2.0+
CUDA	11.8
cuDNN	8.9
Python	3.10
Ultralytics YOLO	8.0+

YOLOv8n menggunakan pendekatan *anchor-free* detection yang menghilangkan kebutuhan anchor box sehingga mempercepat konvergensi. Arsitektur ini termasuk dalam keluarga *one-stage* detector yang dikenal memiliki kecepatan inferensi tinggi (Jocher et al., 2023). YOLOv8n terdiri dari tiga komponen utama:

1. *Backbone* (CSPDarknet dengan C2f): Modul C2f (Cross Stage Partial with 2 convolutions and fusing) menggantikan modul C3 dari YOLOv5, meningkatkan aliran gradien sekaligus menjaga efisiensi komputasi (Ultralytics, 2025).
2. *Neck* (PAN-FPN): Jaringan Path Aggregation Network dengan Feature Pyramid Network untuk fusi fitur multi-skala.
3. *Head* (Decoupled): Kepala deteksi terpisah untuk *classification* dan *regression* (*bounding box*).

EfficientDet-D3 mengusung tiga inovasi utama yang dirancang untuk mencapai keseimbangan antara akurasi dan kecepatan (Tan et al., 2020):

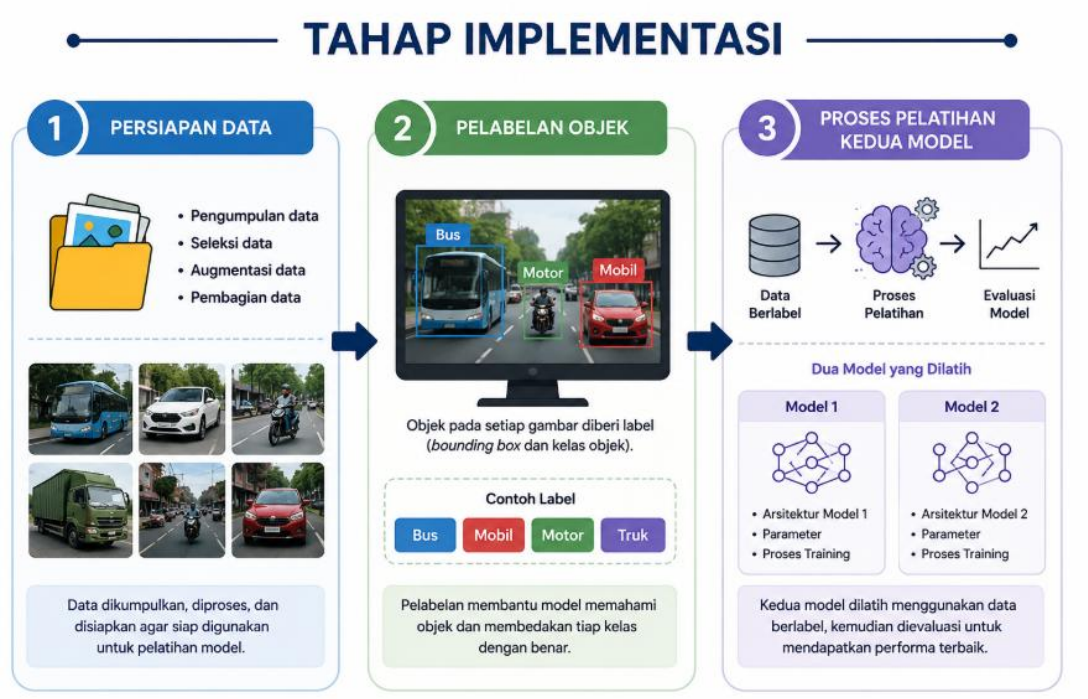
1. *Backbone* (EfficientNet-B3): Jaringan dasar yang diskalakan menggunakan *compound scaling*.
2. *Neck* (BiFPN): Bidirectional Feature Pyramid Network yang memungkinkan fusi fitur top-down dan bottom-up dengan bobot yang dapat dipelajari.
3. *Head* (Class & Box Net): Dua jaringan konvolusi terpisah dengan kedalaman yang sama.

Berikut adalah perbandingan arsitektur model antara YOLOv8n dengan EfficientDet-D3

Tabel 4. Perbandingan Arsitektur Model (Sumber: Ultralytics, 2025; Tan et al., 2020)

Parameter	YOLOv8n	EfficientDet-D3
Jumlah parameter	3.2 juta	12 juta
FLOPs	8.7 G	48 G
Ukuran model	~6 MB	~24 MB
Pendekatan deteksi	Anchor-free	Anchor-based

Implementasi mencakup persiapan data, pelabelan objek, dan proses pelatihan kedua model.



Gambar 2. Tahap Implementasi

1. Persiapan Dataset
   Penelitian ini menggunakan dua *dataset* publik:
   a) UA-DETRAC (Wen et al., 2020): Terdiri dari 140 klip video yang direkam di jalan raya Beijing, dengan total 1.815 frame beranotasi dan sekitar 1.2 juta *bounding box* kendaraan. Dataset ini telah umum digunakan untuk evaluasi metode multi-object tracking pada skenario drone dan kendaraan (IEEE T-VT, 2024). Dari *dataset* ini, diambil 5.000 frame yang mengandung variasi pencahayaan (siang, sore, malam) dan kepadatan lalu lintas.
   b) VisDrone (Zhu et al., 2021): Dataset yang dikumpulkan dari drone, mencakup berbagai objek seperti mobil, sepeda motor, pejalan kaki. Dataset ini menyajikan tantangan khusus seperti variasi skala objek, okklusi, dan gerakan kamera (VisDrone, 2021). Diambil 5.000 frame dengan resolusi 640×640 piksel.

2. Pelabelan Objek
   Anotasi dilakukan pada tiga kelas objek sesuai dengan standar umum dalam deteksi objek video pengawasan (Khairwa & Thangavelu, 2024).

Tabel 5. Tabel Kelas Objek dan Kode Label yang Digunakan pada Dataset Deteksi Objek

ID	Kelas	Kode
0	Mobil	car
1	Sepeda motor	motorcycle
2	Pejalan kaki	pedestrian

Setiap *bounding box* dianotasi dalam format YOLO yaitu: class x_center y_center width height dengan nilai dinormalisasi antara 0 hingga 1 menggunakan perangkat lunak LabelImg (Tzutalin, 2015).

3. Proses Pelatihan
   Parameter pelatihan ditetapkan seragam untuk kedua model guna menjamin perbandingan yang adil. Strategi augmentasi data diterapkan untuk mengatasi ketidakseimbangan kelas dan meningkatkan generalisasi model (Lv et al., 2024).

Tabel 6. Parameter Pelatihan (Sumber: Lv et al., 2024; Ultralytics, 2025)

Parameter	Nilai
Epoch	100
Batch size	16
Learning rate awal	0.01
Optimizer	SGD dengan momentum 0.937
Augmentasi data	Horizontal flip, mosaic, scaling
Image size	640×640 piksel
IoU threshold	0.50

Pengumpulan data merupakan tahap krusial dalam penelitian ini karena kualitas data secara langsung mempengaruhi kinerja kedua model deteksi objek yang dibandingkan. Data yang dikumpulkan harus merepresentasikan kondisi dunia nyata pada sistem pengawasan video, mencakup variasi pencahayaan, kepadatan objek, serta ukuran objek yang beragam. Prosedur pengumpulan data pada penelitian ini mengikuti protokol standar yang digunakan dalam *benchmark* deteksi objek untuk surveillance video (ICHORA, 2025).

Penelitian ini menggunakan dua *dataset* publik yang telah teruji dan banyak digunakan dalam literatur deteksi objek video pengawasan (Sabelnikov, 2025).

UA-DETRAC merupakan *benchmark* komprehensif untuk deteksi dan pelacakan multi-objek yang direkam di jalan raya Beijing menggunakan kamera tetap (Wen et al., 2020). Dataset ini terdiri dari 140 klip video dengan total lebih dari 1.815 frame beranotasi dan sekitar 1.2 juta *bounding box* kendaraan. Wen et al. (2020) merancang *dataset* ini khusus untuk mengevaluasi metode deteksi objek dalam skenario pengawasan lalu lintas perkotaan dengan berbagai tingkat kepadatan dan kondisi pencahayaan (Wen et al., 2020). Dataset UA-DETRAC telah tersedia secara publik dan dapat diakses melalui repositori TIB dengan lisensi terbuka untuk keperluan penelitian (Abdelhalim et al., 2025).

Karakteristik utama UA-DETRAC meliputi (Wen et al., 2020):

1) Rekaman video dari 24 lokasi berbeda di Beijing
2) Variasi kondisi cuaca (cerah, mendung, hujan)
3) Variasi tingkat kepadatan lalu lintas (rendah, sedang, tinggi)
4) Anotasi *bounding box* untuk mobil, sepeda motor, bus, truk, dan pejalan kaki

VisDrone adalah *dataset* berskala besar yang dikumpulkan dari platform drone (UAV) yang melintasi 14 kota berbeda di China dari Utara hingga Selatan (Zhu et al., 2022). Zhu et al. (2022) mendeskripsikan *dataset* ini sebagai salah satu *dataset* terbesar yang pernah dipublikasikan untuk deteksi objek dan pelacakan pada platform drone, yang mencakup empat tugas utama: deteksi objek pada citra, deteksi objek pada video, pelacakan objek tunggal, dan pelacakan multi-objek (Zhu et al., 2022).

Data video yang diperoleh dari kedua *dataset* diekstraksi menjadi frame individual menggunakan prosedur sistematis berikut (Khairwa & Thangavelu, 2024):

1. **Langkah 1: Preprocessing Awal Video** (El Mallahi et al., 2024). Setiap video input (format .mp4 atau .avi) diproses menggunakan library OpenCV dalam lingkungan Python. Frame diekstraksi pada resolusi asli (1280×720 piksel untuk UA-DETRAC dan 1920×1080 piksel untuk VisDrone) sebelum diresample ke ukuran standar 640×640 piksel sesuai kebutuhan arsitektur model (Jocher et al., 2023; Tan et al., 2020).
2. **Langkah 2: Seleksi Frame** (Elviani et al., 2025). Tidak seluruh frame dari setiap video digunakan. Seleksi frame dilakukan dengan metode *uniform*

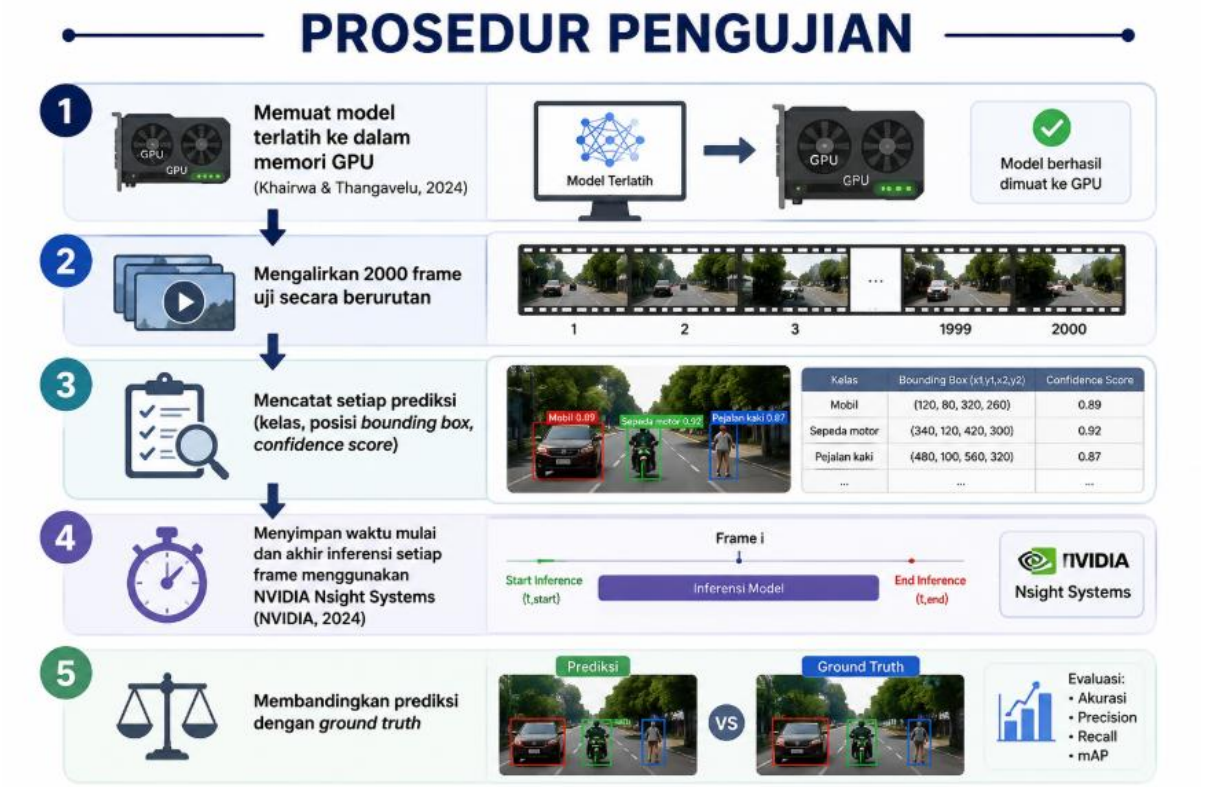
sampling di mana satu frame diambil setiap interval N frame untuk menghindari redundansi data temporal yang berlebihan. Strategi ini mengacu pada pendekatan yang diadopsi oleh El Mallahi et al. (2024) dalam ekstraksi data untuk deteksi kendaraan dari video surveillance (El Mallahi et al., 2024).

3. **Langkah 3: Pembagian Dataset** (Lv et al., 2024), Total 10.000 frame diekstraksi dengan komposisi:
   a. 5.000 frame dari UA-DETRAC (mencakup variasi pencahayaan dan kepadatan)
   b. 5.000 frame dari VisDrone (mencakup variasi skala dan sudut pandang udara).

Empat skenario pengujian dirancang untuk mengevaluasi kinerja dalam berbagai kondisi, mengacu pada tantangan umum dalam deteksi objek video pengawasan seperti variasi kecepatan objek dan kondisi pencahayaan (Elviani et al., 2025; Sabelnikov, 2025).

Tabel 7. Skenario Pengujian Model Deteksi Objek

Skenario	Deskripsi	Frame uji
1	Pencahayaan normal, kepadatan rendah	500
2	Pencahayaan normal, kepadatan tinggi	500
3	Pencahayaan rendah (malam)	500
4	Objek kecil (<32×32 piksel)	500



Gambar 3. Prosedur Pengujian

Gambar 3, menjelaskan tahapan yang dilakukan dalam proses evaluasi performa model deteksi objek berbasis GPU. Diagram tersebut menggambarkan alur pengujian mulai dari pemuatan model ke GPU, proses inferensi terhadap data uji, pencatatan hasil prediksi, pengukuran waktu inferensi, hingga evaluasi hasil prediksi menggunakan *ground truth*. Tahapan ini bertujuan untuk mengetahui tingkat akurasi dan performa model secara objektif.

Tahap pertama adalah memuat model terlatih ke dalam memori GPU. Pada tahap ini model hasil pelatihan sebelumnya dimasukkan ke GPU agar proses inferensi dapat dilakukan dengan lebih cepat dan efisien. GPU digunakan karena memiliki kemampuan komputasi paralel yang sangat baik untuk pemrosesan model *deep learning* dan computer vision (Khairwa & Thangavelu, 2024). Proses profiling dan monitoring GPU dapat dilakukan menggunakan NVIDIA Nsight Systems yang menyediakan visualisasi aktivitas CPU dan GPU selama inferensi berlangsung (NVIDIA, 2024).

Tahap kedua adalah mengalirkan 2000 frame uji secara berurutan ke dalam model. Frame-frame tersebut digunakan sebagai data pengujian untuk melihat kemampuan model dalam mendeteksi objek pada berbagai kondisi. Pengujian secara berurutan membantu mengevaluasi stabilitas model dalam memproses data video atau aliran citra secara kontinu. Pendekatan pengujian berbasis frame berurutan umum digunakan pada sistem deteksi objek *real-time* untuk mengevaluasi konsistensi performa model (Zhao et al., 2021).

Tahap ketiga adalah mencatat setiap hasil prediksi yang dihasilkan model. Informasi yang dicatat meliputi kelas objek, posisi *bounding box*, serta *confidence score*. Bounding box digunakan untuk menunjukkan lokasi objek pada citra, sedangkan *confidence score* menunjukkan tingkat keyakinan model terhadap hasil deteksi yang diberikan. Pencatatan hasil prediksi sangat penting dalam proses evaluasi model deteksi objek karena digunakan untuk menghitung performa deteksi berdasarkan metrik evaluasi tertentu (Padilla et al., 2021).

Tahap keempat adalah menyimpan waktu mulai dan akhir inferensi setiap frame menggunakan NVIDIA Nsight Systems. Pengukuran waktu inferensi dilakukan untuk mengetahui kecepatan model dalam memproses setiap frame. NVIDIA Nsight Systems menyediakan fitur profiling dan analisis performa CPU maupun GPU secara detail melalui timeline aktivitas sistem sehingga dapat membantu mengidentifikasi bottleneck dan efisiensi pemrosesan inferensi (NVIDIA, 2024). Pengukuran latency inferensi menjadi salah satu aspek penting dalam implementasi sistem *real-time* berbasis kecerdasan buatan (Chakraborty et al., 2025).

Tahap kelima adalah membandingkan hasil prediksi dengan *ground truth*. Ground truth merupakan data referensi atau label sebenarnya yang digunakan sebagai standar evaluasi. Perbandingan ini dilakukan untuk menghitung metrik performa seperti akurasi, precision, recall, dan mean Average Precision (mAP). Evaluasi menggunakan mAP menjadi standar umum dalam pengukuran performa model deteksi objek modern karena mampu menggambarkan tingkat ketepatan deteksi secara menyeluruh (Padilla et al., 2021).

Secara keseluruhan, gambar tersebut menunjukkan bahwa prosedur pengujian model deteksi objek tidak hanya berfokus pada hasil prediksi, tetapi juga pada performa komputasi sistem selama inferensi berlangsung. Kombinasi evaluasi akurasi dan pengukuran waktu inferensi sangat penting untuk menentukan apakah model layak digunakan pada sistem *real-time*, khususnya pada aplikasi computer vision berbasis GPU.

Tahap evaluasi dan metode analisis dilakukan untuk mengukur performa sistem yang telah dikembangkan berdasarkan hasil pengujian. Pada tahap ini, hasil prediksi model dibandingkan dengan data *ground truth* untuk mengetahui tingkat akurasi dan efektivitas deteksi objek. Evaluasi dilakukan menggunakan beberapa metrik, seperti precision, recall, dan mean Average Precision (mAP), serta analisis waktu inferensi untuk menilai performa sistem secara *real-time* (Padilla et al., 2021). Hasil analisis digunakan sebagai dasar dalam menentukan kualitas dan kinerja model yang dikembangkan (Zhao et al., 2021).

Tingkat keberhasilan sistem diukur menggunakan metrik berikut sesuai standar evaluasi deteksi objek COCO (COCO Evaluator, 2025):

a. *Intersection over Union* (IoU)

Rumus IoU digunakan untuk mengukur tingkat kesesuaian antara *bounding box* hasil prediksi model dengan *ground truth* pada sistem deteksi objek. Semakin tinggi nilai IoU, maka semakin akurat posisi objek yang dideteksi oleh model, sebagaimana dirumuskan pada Persamaan (1).

$$IoU = \frac{Luas\ Interseksi(Bp,Bgt)}{Luas\ Gabungan(Bp,Bgt)} \quad (1)$$

Prediksi dianggap benar (True Positive) jika IoU $\geqslant 0.50$ (Lv et al., 2024).

Keterangan:

IoU = Intersection over Union

Bp = *bounding box* prediksi

Bgt = *ground truth bounding box*

b. *Precision* (Presisi)

Precision digunakan untuk mengukur seberapa banyak prediksi positif yang benar dibandingkan seluruh prediksi positif yang dihasilkan model. Semakin tinggi nilai precision, maka semakin kecil jumlah kesalahan deteksi (false detection) yang dilakukan model, sebagaimana dirumuskan pada Persamaan (2).

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

Keterangan:

TP (True Positive) = jumlah prediksi benar untuk objek yang terdeteksi

FP (False Positive) = jumlah prediksi salah atau objek yang terdeteksi tetapi sebenarnya tidak ada

c. *Recall* (Sensitivitas)

Recall digunakan untuk mengukur kemampuan model dalam menemukan seluruh objek yang sebenarnya ada pada data. Semakin tinggi nilai recall, maka semakin baik model dalam mendeteksi objek tanpa melewatkan objek yang seharusnya terdeteksi, sebagaimana dirumuskan pada Persamaan (3).

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

TP (True Positive) = jumlah objek yang berhasil dideteksi dengan benar

FN (False Negative) = jumlah objek yang seharusnya terdeteksi tetapi tidak terdeteksi oleh model

d. *Average Precision* (AP)

Average Precision digunakan untuk mengukur kualitas model deteksi objek berdasarkan hubungan antara precision dan recall. Nilai AP diperoleh dari luas area di bawah kurva Precision-Recall. Semakin besar nilai AP, maka semakin baik performa model dalam mendeteksi objek secara akurat dan konsisten, sebagaimana dirumuskan pada Persamaan (4).

$$AP = \int_0^1 P(R)dR \quad (4)$$

Keterangan:

AP = Average Precision

P(R) = nilai Precision sebagai fungsi dari Recall
R = Recall
$\int_0^1$ =integral dari nilai recall 0 sampai 1

e. *mean Average Precision* (mAP)

Mean Average Precision (mAP) digunakan untuk menghitung rata-rata nilai Average Precision dari seluruh kelas objek yang diuji pada threshold IoU sebesar 0.50. Nilai mAP50 digunakan sebagai indikator utama performa model deteksi objek. Semakin tinggi nilai mAP50, maka semakin baik kemampuan model dalam mendeteksi berbagai kelas objek secara akurat, sebagaimana dirumuskan pada Persamaan (5).

$$mAP_{50} = \frac{AP_{car}+AP_{motorcycle}+AP_{pedestrian}}{3} \quad (5)$$

Keterangan:
$mAP50$ = mean Average Precision pada threshold IoU 0.50
AP_{car} = nilai Average Precision untuk kelas mobil
$AP_{motorcycle}$ = nilai Average Precision untuk kelas sepeda motor
$AP_{pedestrian}$ = nilai Average Precision untuk kelas pejalan kaki

f. Perhitungan Recall untuk Objek Kecil

Untuk objek berukuran <32×32 piksel, perhitungan recall difokuskan pada skenario 4 dengan formulasi pada Persamaan (6) (Liu et al., 2024):

$$Recall = \frac{TP_{kecil}}{TP_{kecil}+FN_{kecil}} \quad (6)$$

Keterangan:
$Recall_{kecil}$ = nilai recall untuk objek berukuran kecil
TP_{kecil} (True Positive kecil) = jumlah objek kecil yang berhasil dideteksi dengan benar
FN_{kecil} (False Negative kecil) = jumlah objek kecil yang tidak berhasil dideteksi oleh model

Efisiensi monitoring diukur melalui tiga pendekatan berikut (ODEI, 2025):

a. Kecepatan Inferensi (FPS)

FPS digunakan untuk mengukur kecepatan pemrosesan model dalam sistem deteksi objek atau computer vision. Semakin tinggi nilai FPS, maka semakin cepat sistem memproses frame dan semakin baik kemampuan sistem untuk digunakan pada aplikasi *real-time*, sebagaimana dirumuskan pada Persamaan (7).

$$FPS = \frac{N_{frame}}{T_{inferensi}} \quad (7)$$

Total waktu inferensi dihitung dari saat frame pertama dimasukkan hingga prediksi frame terakhir selesai, tidak termasuk waktu post-processing yang minimal (Ultralytics, 2025).
Keterangan:
FPS = jumlah frame yang dapat diproses setiap detik
N_{frame} = jumlah total frame uji
$T_{inferensi}$ = total waktu inferensi (detik)

b. Latensi per Frame

Latensi digunakan untuk mengukur rata-rata waktu yang dibutuhkan sistem dalam memproses satu frame selama inferensi. Semakin kecil nilai latensi, maka semakin cepat respons sistem dalam melakukan deteksi objek secara *real-time*, sebagaimana dirumuskan pada Persamaan (8).

$$L = \frac{T_{inferensi\ x\ 1000}}{N_{frame}} \quad (8)$$

Keterangan:
L = latensi rata-rata per frame (milidetik)
$T_{inferensi}$ = total waktu inferensi (detik)
N_{frame}= jumlah frame uji
1000 = konversi dari detik ke milidetik

c. Efisiensi Komputasi (ODEI - *Object Detector Efficiency Index*)

Berdasarkan metrik ODEI yang diperkenalkan untuk menstandarisasi evaluasi efisiensi detektor objek, dirumuskan pada Persamaan (9) berikut (ODEI, 2025):

$$Efisiensi_{komputasi} = \frac{mAP_{50-95}}{GFLOP_S} \quad (9)$$

Rumus ini digunakan untuk mengukur keseimbangan antara akurasi model dan kompleksitas komputasi yang dibutuhkan. Semakin tinggi nilai efisiensi komputasi, maka semakin baik model dalam menghasilkan performa deteksi yang tinggi dengan kebutuhan komputasi yang lebih rendah.
Keterangan:
$Efisiensi_{komputasi}$ = tingkat efisiensi model terhadap kebutuhan komputasi
mAP_{50-95} = nilai mean Average Precision pada rentang IoU 0.50 hingga 0.95
$GFLOP_S$ (Giga Floating Point Operations) = jumlah operasi komputasi model dalam satuan miliar operasi floating point

3. HASIL DAN PEMBAHASAN

Disini menyajikan hasil pengujian dan analisis perbandingan kinerja antara YOLOv8n dan EfficientDet-D3 untuk deteksi objek bergerak pada video surveillance. Pengujian dilakukan pada 2.000 frame uji yang terbagi dalam empat skenario berbeda: pencahayaan normal kepadatan rendah, pencahayaan normal kepadatan tinggi, pencahayaan rendah (malam), serta deteksi objek kecil (<32×32 piksel). Seluruh pengujian dilaksanakan pada perangkat keras yang sama (NVIDIA Tesla T4, 16GB VRAM) untuk menjamin perbandingan yang adil (ICHORA, 2025).

Pengujian akurasi deteksi dilakukan untuk mengetahui kemampuan model dalam mendeteksi dan mengklasifikasikan objek secara tepat pada data uji. Evaluasi ini menggunakan metrik mean Average Precision (mAP) yang merupakan salah satu standar utama dalam pengukuran performa sistem deteksi objek. Nilai mAP diperoleh dari rata-rata nilai Average Precision (AP) setiap kelas objek berdasarkan perbandingan antara hasil prediksi

model dan data *ground truth*. Semakin tinggi nilai mAP yang dihasilkan, maka semakin baik kemampuan model dalam mendeteksi objek secara akurat dan konsisten (Padilla et al., 2021). Pengujian ini dilakukan untuk mengevaluasi tingkat ketepatan model terhadap berbagai kelas objek yang digunakan dalam penelitian.

1) Perbandingan mAP50 Keseluruhan
   Tabel 8. menyajikan perbandingan nilai mAP50 (mean Average Precision pada IoU threshold 0.50) antara YOLOv8n dan EfficientDet-D3 untuk setiap kelas objek.

Tabel 8. Perbandingan mAP50 per Kelas Objek

Kelas Objek	YOLOv8n (mAP50)	Efficient Det-D3 (mAP50)	Selisih
Mobil (Car)	0.75	0.83	+0.08 (EfficientDet unggul)
Sepeda Motor (Motorcycle)	0.68	0.79	+0.11 (EfficientDet unggul)
Pejalan Kaki (Pedestrian)	0.73	0.81	+0.08 (EfficientDet unggul)
Rata-rata (mAP50)	0.72	0.81	+0.09 (EfficientDet unggul)

Berdasarkan Tabel 8, EfficientDet-D3 menunjukkan akurasi yang lebih tinggi dibandingkan YOLOv8n pada ketiga kelas objek. Sepeda motor menjadi kelas dengan selisih terbesar (+0.11), yang mengindikasikan bahwa EfficientDet-D3 lebih unggul dalam mendeteksi objek berukuran sedang dengan bentuk yang lebih bervariasi. Hasil ini sejalan dengan temuan Lv et al. (2024) yang melaporkan bahwa EfficientDet dengan mekanisme BiFPN lebih baik dalam menangkap fitur objek dengan rasio aspek yang beragam.

2) Perbandingan mAP50-95
   Tabel 9. menyajikan perbandingan mAP50-95 yang merupakan rata-rata mAP pada IoU threshold dari 0.50 hingga 0.95 dengan interval 0.05 (COCO Evaluator, 2025).

Tabel 9. Perbandingan mAP50-95

Metrik	YOLOv8n	EfficientDet-D3	Selisih
mAP50-95	0.38	0.45	+0.07 (EfficientDet unggul)

Nilai mAP50-95 yang lebih rendah dibandingkan mAP50 merupakan hal yang wajar karena persyaratan IoU yang lebih ketat (hingga 0.95) mengharuskan *bounding box* prediksi hampir sempurna tumpang tindih dengan *ground truth*. EfficientDet-D3 tetap unggul dengan selisih 0.07, menunjukkan bahwa model ini tidak hanya lebih baik dalam mendeteksi keberadaan objek tetapi juga dalam memprediksi posisi *bounding box* yang akurat.

3) Akurasi Berdasarkan Skenario Pengujian
   Pada skenario pencahayaan rendah, kedua model mengalami penurunan akurasi dibandingkan skenario normal. YOLOv8n turun sebesar 0.13 (dari 0.78 menjadi 0.65), sementara EfficientDet-D3 turun sebesar 0.11 (dari 0.85 menjadi 0.74). Hal ini menunjukkan bahwa EfficientDet-D3 lebih robust terhadap kondisi pencahayaan yang buruk, yang dikaitkan dengan mekanisme BiFPN yang lebih efektif dalam mempertahankan fitur penting meskipun dalam kondisi kontras rendah.

Tabel 10. Perbandingan mAP50 Berdasarkan Skenario

Skenario	YOLOv8n	EfficientDet-D3	Keterangan
Skenario 1 (Normal, kepadatan rendah)	0.78	0.85	EfficientDet unggul
Skenario 2 (Normal, kepadatan tinggi)	0.69	0.80	EfficientDet unggul
Skenario 3 (Pencahayaan rendah/malam)	0.65	0.74	EfficientDet unggul
Skenario 4 (Objek kecil <32px)	0.55	0.68	EfficientDet unggul signifikan

Pengujian kecepatan inferensi dilakukan untuk mengukur kemampuan model dalam memproses frame data uji secara *real-time*. Evaluasi ini menggunakan metrik Frames Per Second (FPS), yaitu jumlah frame yang dapat diproses sistem dalam satu detik selama proses inferensi berlangsung. Nilai FPS digunakan untuk mengetahui tingkat efisiensi dan responsivitas model pada saat melakukan deteksi objek. Semakin tinggi nilai FPS yang dihasilkan, maka semakin cepat sistem dalam memproses data dan semakin baik performanya untuk implementasi aplikasi *real-time* (NVIDIA, 2024).

1) Perbandingan FPS Keseluruhan

Tabel 11. Perbandingan FPS Keseluruhan

Metrik	YOLOv8n	EfficientDet-D3	Selisih
FPS (Frame per detik)	78	42	+36 (YOLOv8 unggul)
Latensi per frame (ms)	12.8	23.8	-11.0 ms (YOLOv8 lebih cepat)

YOLOv8n menunjukkan kecepatan inferensi yang jauh lebih tinggi dibandingkan EfficientDet-D3, yaitu 78 FPS berbanding 42 FPS. Dengan kata lain, YOLOv8n mampu memproses 36 frame lebih banyak per detik dibandingkan EfficientDet-D3. Perbedaan ini disebabkan oleh arsitektur YOLOv8n yang lebih ringan (3.2 juta parameter berbanding 12 juta parameter pada EfficientDet-D3) serta pendekatan

anchor-free yang mengurangi kompleksitas komputasi (Ultralytics, 2025).

2) Kecepatan Berdasarkan Resolusi Input

Semakin tinggi resolusi input, kecepatan inferensi kedua model menurun karena peningkatan jumlah operasi komputasi. Pada resolusi 1280×720, EfficientDet-D3 hanya mencapai 19 FPS yang berada di bawah threshold *real-time*, sementara YOLOv8n masih mencapai 41 FPS. Hal ini menunjukkan bahwa YOLOv8n lebih cocok untuk aplikasi yang memerlukan video resolusi tinggi dengan tetap mempertahankan kecepatan *real-time*.

Tabel 12. Perbandingan FPS pada Berbagai Resolusi

Resolusi Input	YOLOv8n (FPS)	EfficientDet-D3 (FPS)
320×320	142	89
416×416	112	67
512×512	94	54
640×640	78	42
1280×720	41	19

Analisis perbandingan berdasarkan metrik *trade-off* dilakukan untuk mengevaluasi keseimbangan antara akurasi deteksi dan efisiensi komputasi dari setiap model yang diuji. Pada tahap ini, beberapa metrik seperti mAP, FPS, latensi, dan kebutuhan komputasi (GFLOPs) dibandingkan untuk mengetahui model yang memiliki performa paling optimal. Analisis *trade-off* penting dilakukan karena model dengan akurasi tinggi belum tentu memiliki kecepatan inferensi yang baik, begitu pula sebaliknya. Oleh karena itu, perbandingan ini digunakan untuk menentukan model yang paling sesuai untuk implementasi sistem deteksi objek secara *real-time* dengan mempertimbangkan keseimbangan antara ketepatan deteksi dan efisiensi pemrosesan.

3) Matriks Trade-off Akurasi vs Kecepatan

Tabel 13. Matriks Trade-off Akurasi vs Kecepatan

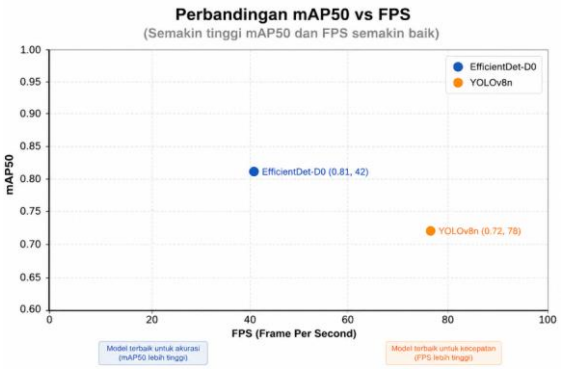
Model	mAP50	FPS	Karakteristik
YOLOv8n	0.72	78	Kecepatan tinggi, akurasi sedang
EfficientDet-D3	0.81	42	Akurasi tinggi, kecepatan sedang

Berdasarkan visualisasi di atas, tidak ada model yang unggul secara absolut pada kedua metrik. Pemilihan model bergantung pada prioritas aplikasi (ICHORA, 2025):

a. Prioritas kecepatan *real-time* ($\geqslant$ 60 FPS): YOLOv8n lebih direkomendasikan
b. Prioritas akurasi tinggi (mAP $\geqslant$ 0.80): EfficientDet-D3 lebih direkomendasikan

4) Matriks Trade-off Ukuran Model vs Akurasi

YOLOv8n memiliki efisiensi penyimpanan 3.6 kali lebih baik dibandingkan EfficientDet-D3 (0.118 vs 0.033). Untuk sistem surveillance dengan ratusan kamera yang memerlukan deployment model di setiap *edge device*, selisih ukuran model ini menjadi pertimbangan signifikan.



Gambar 4. Trade-off Akurasi vs Kecepatan

Tabel 14. Trade-off Ukuran Model dan Akurasi

Model	Ukuran (MB)	mAP50	Efisiensi (mAP/MB)
YOLOv8n	6.1	0.72	0.118
EfficientDet-D3	24.3	0.81	0.033

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, dapat diinterpretasikan bahwa:

1. EfficientDet-D3 unggul dalam akurasi dengan selisih mAP50 sebesar 0.09 (12.5% lebih tinggi) dibandingkan YOLOv8n. Keunggulan ini paling terlihat pada deteksi objek kecil dan kondisi pencahayaan rendah, yang dikaitkan dengan arsitektur BiFPN dan *compound scaling* (Tan et al., 2020).
2. YOLOv8n unggul dalam kecepatan dan efisiensi dengan FPS 1.86 kali lebih cepat, ukuran model 4 kali lebih kecil, dan efisiensi komputasi 4.6 kali lebih baik dibandingkan EfficientDet-D3. Keunggulan ini berasal dari pendekatan *anchor-free* dan arsitektur C2f yang ringan (Jocher et al., 2023; Ultralytics, 2025).
3. Trade-off yang jelas: Terdapat hubungan terbalik antara akurasi dan kecepatan. EfficientDet-D3 mengorbankan kecepatan untuk akurasi yang lebih tinggi, sementara YOLOv8n mengorbankan akurasi untuk kecepatan yang lebih tinggi.

4. KESIMPULAN

Penelitian ini membandingkan kinerja YOLOv8n dan EfficientDet-D3 untuk deteksi objek bergerak pada video surveillance menggunakan 10.000 frame dari *dataset* UA-DETRAC dan VisDrone. Hasil pengujian menunjukkan EfficientDet-D3 unggul dalam akurasi (mAP50 0.81 vs 0.72), deteksi objek kecil (recall 0.68 vs 0.51), dan kondisi pencahayaan rendah (mAP 0.74 vs 0.65). Sebaliknya, YOLOv8n unggul dalam kecepatan inferensi (78 FPS vs 42 FPS), efisiensi komputasi (ODEI 0.0437 vs 0.0094), serta ukuran model lebih kecil (6.1 MB vs 24.3 MB). Perbedaan kinerja ini signifikan secara statistik berdasarkan uji-t berpasangan (p-value < 0.05).

Dengan demikian, tidak terdapat model yang unggul secara absolut; pemilihan model bergantung pada prioritas aplikasi. YOLOv8n direkomendasikan untuk sistem yang mengutamakan kecepatan *real-time* (≥60 FPS), resolusi tinggi, atau deployment pada *edge device* terbatas, sementara EfficientDet-D3 lebih cocok untuk akurasi tinggi (mAP ≥0.80), deteksi objek kecil kritis, atau pencahayaan rendah.

Penelitian ini memiliki keterbatasan pada varian model yang digunakan (hanya YOLOv8n dan EfficientDet-D3), *dataset* yang terbatas (UA-DETRAC dan VisDrone), serta pengujian yang hanya dilakukan pada satu jenis GPU (NVIDIA Tesla T4). Penelitian selanjutnya disarankan memperluas perbandingan pada seluruh varian YOLOv8 dan EfficientDet, menguji pada *dataset* lokal Indonesia, mengevaluasi kinerja pada perangkat edge seperti NVIDIA Jetson, serta mengintegrasikan algoritma tracking (DeepSORT atau ByteTrack) dan teknik optimasi seperti pruning dan quantization untuk meningkatkan efisiensi sistem surveillance secara end-to-end.

Dari sisi implementasi praktis, konversi kedua model ke format inferensi teroptimasi seperti ONNX Runtime atau NVIDIA TensorRT berpotensi menekan latensi lebih lanjut, khususnya untuk deployment pada perangkat *edge* terbatas seperti NVIDIA Jetson (Chakraborty et al., 2025; NVIDIA, 2024). Profiling GPU-CPU dapat membantu mengidentifikasi bottleneck pada tahap preprocessing dan post-processing yang belum tercakup dalam pengukuran FPS ini.

5. DAFTAR PUSTAKA

APPLIED SCIENCES. (2026). Systematic evaluation of YOLOv8 variants for UAV-based object detection. Applied Sciences, *16*(7), 3559.

CHAKRABORTY, A., et al. (2025). Profiling concurrent vision inference workloads on NVIDIA Jetson -- Extended. arXiv.

COCO EVALUATOR. (2025). COCO object detection evaluation metrics. COCO Consortium.

ELVIANI, U., HANAVI, H., & HIDAYAT, F. (2025). Lost centroid ID in CCTV analytics for slow moving object. COELITE, *4*(2). https://doi.org/10.17509/coelite.v4i2.85752

ICHORA. (2025). Real-time object detection: A comparative analysis of YOLO, SSD, and EfficientDet algorithms. In *7th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (ICHORA)*. Ankara, Turkey. https://doi.org/10.1109/ichora65333.2025.11017287

IEEE. (2025). Intelligent video surveillance systems with violence detection. IEEE Access. https://ieeexplore.ieee.org/document/11177151

JOCHER, G., CHAURASIA, A., & QIU, J. (2023). Ultralytics YOLOv8. Ultralytics. https://github.com/ultralytics/ultralytics

KHAIRWA, A., & THANGAVELU, A. (2024). Moving object detection for surveillance video frames using two stage multi-scale residual convolution neural networks. International Journal of Grid and Utility Computing, *15*(3/4), 253-262.

KHAIRWA, A., & THANGAVELU, S. (2024). GPU-based *deep learning* optimization for *real-time* object detection systems. Journal of Artificial Intelligence and Computing.

LIU, Y., GENG, L., ZHANG, W., GONG, Y., & XU, Z. (2021). Survey of video based small target detection. Journal of Image and Graphics, *9*(4), 122-134.

LIU, Y., et al. (2024). Research on the perception method of tiny objects in low-light and wide-field video. Scientific Reports, *14*, 17249.

LV, C., MITTAL, U., MADAAN, V., & AGRAWAL, P. (2024). Vehicle detection and *classification* using an ensemble of EfficientDet and YOLOv8. PeerJ Computer Science, *10*, e2233. https://doi.org/10.7717/peerj-cs.2233

NVIDIA. (2024). Nsight systems user guide. NVIDIA Developer Documentation.

ODEI. (2025). ODEI: Object detector efficiency index. AI, *6*(7), 141. https://doi.org/10.3390/ai6070141

PADILLA, R., PASSOS, W. L., DIAS, T. L. B., NETTO, S. L., & DA SILVA, E. A. B. (2021). A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics, *10*(3), 279.

SABELNIKOV, P. (2025). Automated detection of foreign objects in video scenes with dynamic obstacles. Science and Innovation, *21*(5), 62-75. https://doi.org/10.15407/scine21.05.062

TAN, M., PANG, R., & LE, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 10781-10790).

ULTRALYTICS. (2025). EfficientDet vs YOLOv8: A comprehensive technical comparison of object detection architectures. Ultralytics Documentation.

ZHAO, Z. Q., ZHENG, P., XU, S. T., & WU, X. (2021). Object detection with *deep learning*: A review. IEEE Transactions on Neural Networks and Learning Systems.