INNOVATIVE: Journal Of Social Science Research
Volume 5 Nomor 4 Tahun 2025 Page 8593-8601
E-ISSN 2807-4238 and P-ISSN 2807-4246
Website: https://j-innovative.org/index.php/Innovative

Analisis Perbandingan Performa YOLO v11 Dan v12 menggunakan model N dan S

Gidion Albeth Anoraga Putra[1✉], Alz Danny Wowor[2]
Universitas Kristen Satya Wacana
Email: alzdanny.wowor@uksw.edu[1✉]

Abstract

Penelitian ini bertujuan untuk menganalisis perbandingan kinerja model deteksi objek YOLO (You Only Look Once) versi 11 dan versi 12, dengan fokus pada variasi N dan S pada kedua versi tersebut. Analisis dilakukan terhadap parameter waktu inferensi, kecepatan pemrosesan, penggunaan memori, dan ukuran model. Data diperoleh melalui eksperimen terhadap kedua versi dengan melakukan 100 epoch pelatihan pada masing-masing model. Hasil penelitian menunjukkan bahwa YOLO versi 11 secara umum memiliki waktu eksekusi yang lebih cepat dibandingkan versi 12, dengan total waktu eksekusi 219 detik untuk versi 11 N dan 228 detik untuk versi 11 S, sementara versi 12 N membutuhkan 303 detik dan versi 12 S membutuhkan 420 detik. Versi 11 juga menunjukkan penggunaan memori yang lebih efisien, sekitar 126-127 MB dibandingkan dengan versi 12 yang membutuhkan memori sekitar 3674-4309 MB. Penelitian ini menyimpulkan bahwa YOLO versi 11 menunjukkan keunggulan signifikan dalam efisiensi waktu pelatihan, penggunaan memori, dan waktu inferensi, menjadikannya pilihan optimal dalam hal sumber daya atau kebutuhan pemrosesan real-time. Sebaliknya, YOLO versi 12 menawarkan konsistensi kinerja yang lebih baik dan potensi akurasi yang lebih tinggi dengan biaya komputasi yang substansial.

Keywords: *YOLO, Deteksi Objek, Perbandingan Model, Deep Learning.*

Abstract

This research aims to analyze the performance comparison of the YOLO (You Only Look Once) object detection model versions 11 and 12, focusing on variations N and S in both versions. The analysis is conducted on the parameters of inference time, processing speed, memory usage, and model size. Data is obtained through experiments on both versions by performing 100 epochs of training on each model. The results indicate that YOLO version 11 generally has a faster execution time compared to version 12, with a total execution time of 219 seconds for version 11 N and 228 seconds for version 11 S, while version 12 N requires 303 seconds and version 12 S requires 420 seconds.Version 11 also demonstrates more efficient memory usage, around 126-127 MB compared to version 12 which requires memory around 3674-4309 MB. This study concludes that YOLO version 11 shows significant advantages in training time efficiency, memory usage, and inference time, making it the optimal choice in terms of resource or real-time processing needs. In contrast, YOLO version 12 offers better performance consistency and the potential for higher accuracy at a substantial computational cost.

.Keywords: *YOLO, Object Detection, Model Comparison, Deep Learning*

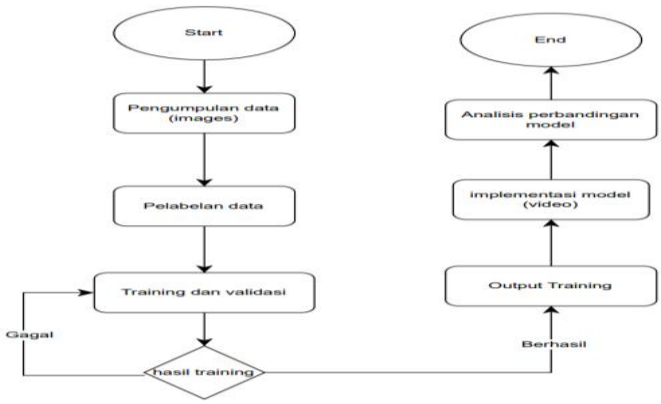
PENDAHULUAN

Dalam dunia yang serba modern ini, teknologi deteksi object dalam bidang *computer vision* telah mengalami perkembangan pesat seiring dengan evolusi teknologi *deep learning,* dengan algoritma *You Only Look Once (*YOLO) merupakan salah satu pendekatan yang paling popular, karena memiliki kelebihan dalam kecepatan eksekusi tinggi dengan Tingkat akurasi yang presisi, keunggulan utama YOLO terletak pada pendekatan regresi langsung. Algoritma YOLO telah menjadi terobosan yang signifikan dalam teknologi deteksi mobject *real-time* sejak diperkenalkan oleh Redmon pada tahun 2016. Mulai dari *versi* sampai dengan yang terbaru, YOLO telah mengalami peningkatan yang signifikan dalam hal inovasi dan performa . Contoh penerapan secaral *real-time*nya seperti sistem pengawasan otomatis, deteksi objek pada perangkat *mobile*, dan sistem *autonomus driving* yang memerlukan pertimbangan cermat terhadap efisiensi waktu inferensi dan penggunaan memori. Sebaiknya, aplikasi yang memperioritaskan akurasi maksimal mungkin dapat menoleransi konsumsi sumber daya yang lebih tinggi . Tentunya hal ini akan sangat penting karena kecepatan inferensi akan menjadi lebih kritikal kalau dibandingkan dengan akurasi deteksi . Penelitian tentang Evaluasi yang komperhensif dengan *versi* yang berbeda telah dilakukan sebelumnya, namun hanya mencakup YOLO v5,v8,v9,v10, dan v11 saja, dan hasilnya yaitu terjadi peningkatan yang konsisten seiring dengan evolusi setiap *versi* YOLO.

Fokus Penelitian ini ada pada perbandingan antara dua versi dari algoritma YOLO, yaitu 11 dan 12 yang merupakan versi masih baru, dengan masing-masing memiliki model N dan S yang masih terbatas dalam literatur. Penelitian ini bertujuan untuk melakukan analisis mendalam terhadap parameter kinerja komputasi kedua versi YOLO seperti waktu eksekusi, penggunaan memori, kecepatan inferensi, dan bagaimana pengaruh ukuran terhadap kinerja pada masing masing versi YOLO . Kontribusi utama penelitian ini adalah menyediakan analisis komprehensif mengenai performa dari YOLO v11 dan v12, memberikan panduan praktis untuk pemilihan model berdasarkan kebutuhan aplikasi spesifik, dan mengidentifikasi karakteristik unik masing-masing versi dalam konteks efisiensi sumber daya.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan Kuantitatif untuk menganalisis perbandingan Kinerja antara YOLO *versi* 11 dan 12, dengan masing-masing *versi* terdapat dua *model* yaitu model N dan *model* S. Analisis ini mancakup beberapa tahapan yang digunakan untuk setiap model, mulai dari pengumpulan data yang berupa image, pelabelan data secara manual menggunakan roboflow, *training* dan validasi,output training, Implementasi model, dan analisis perbandingan model. Semua tahapan ini dapat dilihat pada Gambar 1.



Gambar 1: Alur penelitian

Pengumpulan data

Pengumpulan data dimulai dari pencarian *image*(gambar) dan video yang sesuai dengan objek yang ingin dideteksi, pada penelitian ini penulis menggunakan data plat nomer sebagai objek yang ingin dideteksi*, image* yang dikumpulkan sebanyak 199 *images*. Dan juga 5 video untuk menguji implementasi deteksi objek dengan ukuran video sebesar 1,69 mb, 1,91 mb , 2,50 mb, 7,18 mb, dan 22,80 mb.

Pelabelan data

Pelabelan data dilakukan secara manual pada satu per satu images yang sudah

dikumpulkan menggunakan roboflow untuk membuat *model computer vision*nya.

Training dan validasi

Training dan validasi dilakukan untuk mengevaluasi dan melatih model supaya algoritma YOLO dapat mengenali objek yang akan dideteksi, penelitian ini menggunakan 100 epoch pada semua model untuk memperoleh hasil yang lebih optimal.

output training

Berdasarkan Hasil training yang telah dilakukan maka akan dihasilkan output seperti berapa lama waktu eksekusi per epoch dan total waktu pelatihan, yang nantinya akan digunakan untuk menganalisis perbandingan antar *model*.

Implementasi model

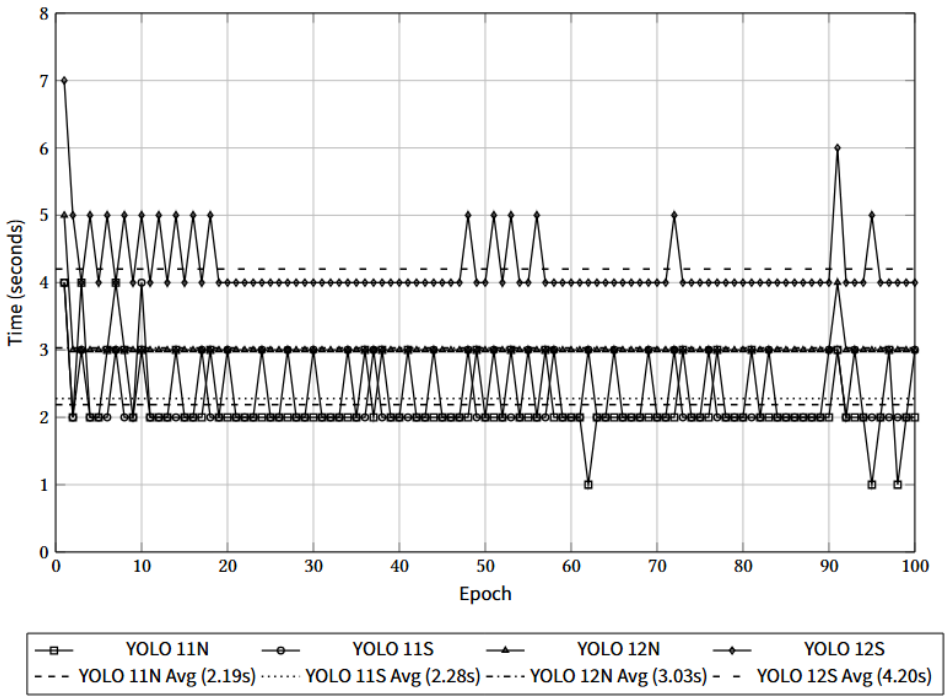
Setelah hasil training selesai, yang selanjutnya dilakukan Adalah mengimplementasikan model yang telah selesai di *training* untuk dapat mendeteksi objek berupa video yang akan menghasilkan data seperti kecepatan *model*, inferensi, dan penggunaan memori pada setiap *model*, yang selanjutnya akan digunakan untuk melakukan analisis perbandingan antar *model*.

HASIL DAN PEMBAHASAN

Setelah melakukan training dan implementasi *model* pada setiap model YOLO, maka telah dihasilkan data yang dapat digunakan untuk membandingkan performa antara YOLO v11 dan 12 dengan dua *model* yaitu *model* N dan S sebagai berikut:

Analisis Waktu Eksekusi

Hasil pengukuran waktu eksekusi menunjukan perbedaan yang signifikan antara kedua *versi* YOLO. Data waktu eksekusi per-epoch menunjukan bahwa YOLO 11 memiliki performa yang lebuh superior dalam hal kecepatan pemrosesan dibandingkan dengan *versi* 12, seperti yang ada pada gambar 2.



Gambar 2: Lama Waktu Pelatihan per Epoch Pada Setiap model YOLO

Berdasarkan data hasil pengujian, waktu pemrosesan per epoch untuk setiap varian YOLO menunjukkan variasi yang signifikan. Pada YOLO *versi* 11, rentang waktu pemrosesan per *epoch* untuk 11 N berkisar antara 1-4 detik, dengan mayoritas berada pada 2 detik. Untuk 11 S, rentang waktu berkisar antara 2-4 detik, dengan mayoritas juga berada pada 2 detik, sementara itu, pada YOLO versi 12, terdapat peningkatan waktu pemrosesan yang konsisten. Untuk 12 N, waktu pemrosesan berkisar antara 3-5 detik, dengan mayoritas berada pada 3 detik. Untuk 12 S, waktu pemrosesan lebih tinggi lagi, berkisar antara 4-7 detik, dengan mayoritas berada pada 4 detik. Dari awal pengujian (*epoch* 1), terlihat bahwa *versi* 12 memerlukan waktu pemrosesan yang lebih lama dibandingkan *versi* 11. YOLO 11 N dan 11 S memulai dengan 4 detik, sementara 12 N memulai dengan 5 detik dan 12 S memulai dengan 7 detik. Setelah beberapa epoch, waktu pemrosesan cenderung stabil pada masing-masing varian.

Total Waktu Pelatihan

Statistik waktu eksekusi yang tercantum dalam Tabel 1 menunjukkan bahwa YOLO 11N memiliki rata-rata waktu eksekusi 2,19 detik per epoch, sementara YOLO 12S memerlukan waktu paling lama dengan rata-rata 4,20 detik per epoch.

Tabel 1. Statistik waktu eksekusi pelatihan

Statistik	YOLO 11 N	YOLO 11 S	YOLO 12 N	YOLO 12 S
Rata-rata(detik)	2.19	2.28	3.03	4.20
Minimum	1	2	3	4
Maksimum	4	4	5	7
Total (100 *Epoch)*	219	228	303	420
Total (Menit*)*	3.65	3.80	5.05	7.00

Dari tabel 1, terlihat bahwa YOLO versi 11 memiliki total waktu pemrosesan yang lebih cepat dibandingkan dengan YOLO versi 12. Selisih waktu antara 11 N dan 12 N adalah 84 detik (1,4 menit), sementara selisih waktu antara 11 S dan 12 S lebih besar yaitu 192 detik (3,2 menit). Hal ini menunjukkan bahwa peningkatan versi dari 11 ke 12 mengakibatkan peningkatan waktu pemrosesan yang signifikan, terutama pada varian S.

Perbandingan Penggunaan Memori

Penggunaan memori menunjukkan perbedaan dramatis antara kedua versi, seperti yang ditampilkan dalam Tabel 2. YOLO versi 12 memiliki penggunaan memori yang substansial lebih tinggi dibandingkan versi 11.

Tabel 2. Perbandingan penggunaan memori

Ukuran Data (MB)	YOLO 11 N	YOLO 11 S	YOLO 12 N	YOLO 12 S
1.69	126.40	127.41	3674.93	4309.43
1.91	126.40	127.41	3674.43	4309.44
2.50	126.40	127.41	3674.80	4309.43
7.18	126.40	127.41	3675.15	4309.46
22.80	126.40	127.41	3675.09	4309.43

Dari tabel 2, Data menunjukkan bahwa YOLO versi 11 memiliki penggunaan memori yang stabil dan efisien, tidak terpengaruh oleh ukuran data input. Peningkatan penggunaan memori dari versi 11 ke versi 12 sangat signifikan, di mana YOLO 12 N menggunakan memori sekitar 29 kali lebih besar dibandingkan dengan YOLO 11 N, dan YOLO 12 S menggunakan memori sekitar 34 kali lebih besar dibandingkan dengan YOLO 11 S, yang tentumya dapat menjadi keterbatasan signifikan dengan sumber daya.

Perbandingan Kecepatan (*millisecond*)

Pengukuran kecepatan pemrosesan yang diukur dalam *milisecond* pada 5 ukuran *file* yang berbeda untuk masing-masing *model* menunjukkan hasil yang

menarik, dimana YOLO *versi* 12 sedikit lebih unggul dalam hal kecepatan, seperti yang ditampilkan dalam Tabel 3.

Tabel 3. Perbandingan kecepatan pemrosesan (millisecond)

Ukuran Data (MB)	YOLO 11 N	YOLO 11 S	YOLO 12 N	YOLO 12 S
1.69	2.6	2.5	2.4	2.4
1.91	3.4	3.6	3.3	3.3
2.50	2.6	2.7	2.5	2.5
7.18	3.7	3.5	3.3	3.4
22.80	4.0	3.8	3.7	3.5
Rata-rata(ms)	3.26	3.22	3.04	3.02

Menariknya, meskipun YOLO versi 12 memiliki waktu pemrosesan per *epoch* dan penggunaan memori yang lebih besar, data kecepatan menunjukkan bahwa *versi* 12 sedikit lebih cepat dibandingkan dengan *versi* 11 pada beberapa ukuran file. Ini mengindikasikan bahwa YOLO *versi* 12 memiliki optimasi tertentu yang meningkatkan kecepatan pemrosesan meskipun menggunakan lebih banyak sumber daya. Selisih kecepatan antar ukuran *file* menunjukkan pola yang serupa pada semua varian YOLO, dengan rata-rata selisih berkisar antara 0,275 ms hingga 0,35 ms. Selisih terbesar terjadi pada transisi dari ukuran file 1,91 MB ke 2,5 MB dan dari 2,5 MB ke 7,18 MB.

Perbandingan Waktu Inferensi (*millisecond)*

dalam hal waktu inferensi, YOLO versi 11 menunjukkan performa yang superior dengan rata-rata waktu inferensi sekitar 11,6 ms, sementara versi 12 memerlukan sekitar 18,5 ms, seperti yang ditunjukkan dalam Tabel 4.

Tabel 4. Perbandingan waktu inferensi (millisecond)

Ukuran Data (MB)	YOLO 11 N	YOLO 11 S	YOLO 12 N	YOLO 12 S
1.69	9.7	9.2	15.7	16.4
1.91	11.3	11.9	18.4	19.4
2.50	9.8	9.9	15.9	16.1
7.18	12.8	12.5	19.1	19.4
22.80	14.3	14.9	22.3	22.1
Rata-rata(ms)	11.58	11.68	18.28	18.68

Jika dilihat pada Tabel 4, terlihat bahwa YOLO *versi* 12 memiliki waktu inferensi yang lebih lama dibandingkan dengan YOLO *versi* 11.Dimana Rata-rata waktu inferensi untuk YOLO 11 N adalah sekitar 11,58 ms, untuk 11 S sekitar 11,68 ms, untuk 12 N sekitar 18,28 ms, dan untuk 12 S sekitar 18,68 ms. Hal ini menunjukkan bahwa waktu inferensi *versi* 12 sekitar 1,6 kali lebih lama dibandingkan dengan versi 11. Selisih waktu inferensi antar ukuran *file* menunjukkan pola yang cukup konsisten pada semua

varian YOLO, dengan rata-rata selisih berkisar antara 1,15 ms hingga 1,65 ms. Selisih terbesar terjadi pada transisi dari ukuran file 1,91 MB ke 2,5 MB dan dari 2,5 MB ke 7,18 MB, yang serupa dengan pola pada kecepatan.

SIMPULAN

Penelitian ini berhasil mengidentifikasi karakteristik kinerja yang berbeda antara YOLO versi 11 dan 12. YOLO versi 11 menunjukkan keunggulan signifikan dalam efisiensi waktu pelatihan, penggunaan memori, dan waktu inferensi, menjadikannya pilihan optimal dalam hal sumber daya atau kebutuhan pemrosesan real-time. Sebaliknya, YOLO versi 12 menawarkan konsistensi kinerja yang lebih baik dan potensi akurasi yang lebih tinggi dengan biaya komputasi yang substansial. Penelitian ini memberikan gambaran analisis perbandingan yang akan berguna dalam memilih model YOLO yang sesuai dengan kebutuhan aplikasi spesifik mereka. Untuk aplikasi dengan sumber daya terbatas atau kebutuhan respons cepat, YOLO versi 11 tetap menjadi pilihan yang lebih tepat. Sementara untuk aplikasi yang memprioritaskan akurasi dan konsistensi dengan ketersediaan sumber daya yang memadai, YOLO versi 12 dapat menjadi alternatif yang layak dipertimbangkan.

Untuk penelitian selanjutnya, disarankan untuk melakukan analisis terhadap akurasi deteksi objek dari kedua versi YOLO dan menguji kedua model pada dataset yang lebih bervariasi untuk menilai kemampuan generalisasi mereka.

DAFTAR PUSTAKA

Y.Wang, Bochkovskiy and H. Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection,"Computer Vision and Pattern Recognition, 2020.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A, "You only look once: Unified, real time object detection," IEEE Conference on Computer Vision and Pattern Recognition, 2016.

M.Hussain, "Yolov5, yolov8and yolov10: The go-todetectors for real-time vision," arXiv:2407.02988, 2024.

Mingxing Tan, Ruoming Pang, Quoc V. Le, "Efficientdet: Scalable and efficient object detection," IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10781 10790), 2020.

U. Sirisha, S. Phani Praveen, Parvathaneni Naga Srinivasu, Paolo Barsocchi, Akash Kumar Bhoi, "Statistical analysis of design aspects of various yolo-based deep learning models for object detection," International Journal of Computational

Intelligence Systems, 2023.

Zijian He, Kang Wang, Tian Fang, Lei Su, Rui Chen, Xihong Fei, "Comprehensive performance evaluation of yolov11,

yolov10, yolov9, yolov8 and yolov5 on object detection of power equipment," arXiv preprint arXiv:2411.18871, 2024.

Peiyuan jiang, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma, "A Review of Yolo Algorithm Developments" , Procedia Computer Science Volume 199, 2022, Pages 1066-1073

G. Jocher, A. Stoken, J. Borovec, N. J. Chaurasia, and L. Changyu, "ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support," Oct. 2021, doi: 10.5281/zenodo.5563715.

Redmon, J., & Farhadi, A., " YOLOv3: An incremental improvement". arXiv preprint arXiv:1804.02767, 2018.

Yunjie Tian, Qixiang Ye, David Doermann, " YOLOv12: Attention-Centric Real-Time Object Detectors", arXiv:2502.12524, 2025.

Hussain, M., Bird, J. J., & Faria, D. R, "A study on CNN transfer learning for image classification", UK Workshop on Computational Intelligence, 2018.