Institut Riset dan Publikasi Indonesia (IRPI)
MALCOM: Indonesian Journal of Machine Learning and Computer Science
Journal Homepage: https://journal.irpi.or.id/index.php/malcom
Vol. 6 Iss. 2 April 2026, pp: 513-523
ISSN(P): 2797-2313 | ISSN(E): 2775-8575

Implementation and Comparison of YOLO v8, v9, and v11 in Occupational Safety Regulations: Detecting Safety Helmet Usage in Construction Environments

Implementasi Perbandingan *YOLO v8, v9,* dan *v11* dalam Penerapan Tata Tertib K3: Deteksi Penggunaan Helm Keselamatan di Lingkungan Konstruksi

Rafael Praseli[1*], Nenden Siti Fatonah[2], Diah Aryani[3], Hani Dewi Ariessanti[4]

[1,2,3,4]Program Studi Teknik Informatika, Fakultas Ilmu Komputer,
Universitas Esa Unggul, Indonesia

E-Mail: [1]rafael.praseli12@student.esaunggul.ac.id, [2]nenden.siti@esaunggul.ac.id,
[3]diah.aryani@esaunggul.ac.id, [4]hani.dewi@esaunggul.ac.id

Received Jan 30th 2026; Revised Feb 27th 2026; Accepted Mar 05th 2026; Available Online Apr 19th 2026
Corresponding Author: Rafael Praseli

Abstract

Consistent enforcement of occupational safety regulations is an important step toward creating a safe, well-organized construction environment. This study employs an experimental approach using the Hard Hats Computer Vision Project dataset, which consists of approximately 20,000 annotated images, and utilizes artificial intelligence with YOLOv8, YOLOv9, and YOLOv11 algorithms. The models are trained on construction worker image data and evaluated using three main metrics, namely precision, recall, and mean Average Precision (mAP), to measure detection accuracy and generalization capability. The results show that YOLOv8 achieves stable performance with a precision of 0.895, recall of 0.895, and mAP@50–95 of 0.597 at 60 epochs, while YOLOv11 achieves the highest accuracy with a precision of 0.903, recall of 0.897, and mAP@50–95 of 0.600. Meanwhile, YOLOv9 demonstrates lower efficiency compared to the other models. Therefore, YOLOv8 is more suitable for real-time implementation due to its stability and computational efficiency, while YOLOv11 offers greater potential for higher accuracy. This study contributes both academically to the development of AI-based systems and practically to improving occupational safety in construction environments.

Keywords: Artificial Intelligence, Computer Vision, Occupational Safety and Health, Safety Helmet Detection, YOLO.

Abstrak

Penegakan aturan keselamatan kerja secara konsisten merupakan langkah penting dalam menciptakan lingkungan konstruksi yang aman dan tertib. Penelitian ini dilakukan dengan pendekatan eksperimen menggunakan *dataset Hard Hats Computer Vision Project* yang berjumlah sekitar 20.000 gambar beranotasi, dengan memanfaatkan teknologi kecerdasan buatan melalui algoritma *YOLOv8, YOLOv9,* dan *YOLOv11*. Model dilatih menggunakan *dataset* pekerja konstruksi dan dievaluasi berdasarkan tiga metrik utama, yaitu *precision, recall,* serta *mean Average Precision* (*mAP*) untuk mengukur akurasi dan kemampuan generalisasi deteksi. Hasil penelitian menunjukkan bahwa *YOLOv8* menghasilkan performa yang stabil dengan nilai *precision* sebesar 0,895, *recall* sebesar 0,895, dan *mAP*@50–95 sebesar 0,597 pada 60 *epoch*, sedangkan *YOLOv11* mencapai akurasi tertinggi dengan *precision* sebesar 0,903, *recall* sebesar 0,897, dan *mAP*@50–95 sebesar 0,600. Sementara itu, *YOLOv9* menunjukkan efisiensi yang lebih rendah dibandingkan dengan kedua model lainnya. Dengan demikian, *YOLOv8* lebih sesuai untuk implementasi *real-time* karena stabil dan efisien, sedangkan *YOLOv11* memiliki potensi akurasi yang lebih tinggi. Penelitian ini memberikan kontribusi baik secara akademis dalam pengembangan sistem berbasis AI maupun secara praktis dalam meningkatkan keselamatan kerja di lingkungan konstruksi.

Kata Kunci: Visi Komputer, Deteksi Helm Keselamatan, Keselamatan dan Kesehatan Kerja, Kecerdasan Buatan, *YOLO.*

1. PENDAHULUAN

Industri konstruksi di Indonesia masih menghadapi tantangan besar dalam penerapan Keselamatan dan Kesehatan Kerja (K3). Data Kementerian Ketenagakerjaan menunjukkan bahwa pada tahun 2024 tercatat lebih dari 462.000 kasus kecelakaan kerja di berbagai sektor, di mana sektor konstruksi memberikan kontribusi signifikan terhadap angka kecelakaan nasional. Selain itu, laporan BPJS Ketenagakerjaan juga menunjukkan peningkatan klaim Jaminan Kecelakaan Kerja (JKK) dari tahun ke tahun, yang mengindikasikan masih tingginya risiko kecelakaan di lingkungan kerja. Kondisi ini menegaskan pentingnya penerapan K3 secara konsisten, termasuk kewajiban penggunaan alat pelindung diri (APD) berupa helm keselamatan sebagaimana diatur dalam Undang-Undang No. 1 Tahun 1970 [1].

Meskipun regulasi K3 telah ditetapkan, tingkat kepatuhan pekerja konstruksi terhadap penggunaan helm keselamatan di lapangan masih relatif rendah. Beberapa penelitian menyebutkan bahwa kelalaian pekerja, rendahnya kesadaran terhadap risiko, serta keterbatasan efektivitas pengawasan manual menjadi faktor utama terjadinya pelanggaran K3 [2]. Cedera kepala merupakan salah satu jenis cedera yang paling sering terjadi dan memiliki tingkat fatalitas tinggi dalam kecelakaan kerja, sehingga menimbulkan dampak serius baik secara fisik maupun sosial-ekonomi bagi pekerja [3]. Oleh karena itu, diperlukan upaya yang lebih efektif untuk meningkatkan pengawasan dan kepatuhan terhadap penggunaan APD di lingkungan konstruksi.

Perkembangan teknologi kecerdasan buatan *(Artificial Intelligence)* dan visi komputer *(Computer Vision)* membuka peluang baru dalam meningkatkan sistem pemantauan keselamatan kerja. Berbagai algoritma deteksi objek, seperti Faster R-CNN, SSD, dan *EfficientNet*, telah digunakan dalam penelitian sebelumnya, namun masih menghadapi keterbatasan dalam keseimbangan antara akurasi dan kecepatan deteksi untuk aplikasi *real-time* [4], [5], [6], [7], [8], [9], [10], [11].

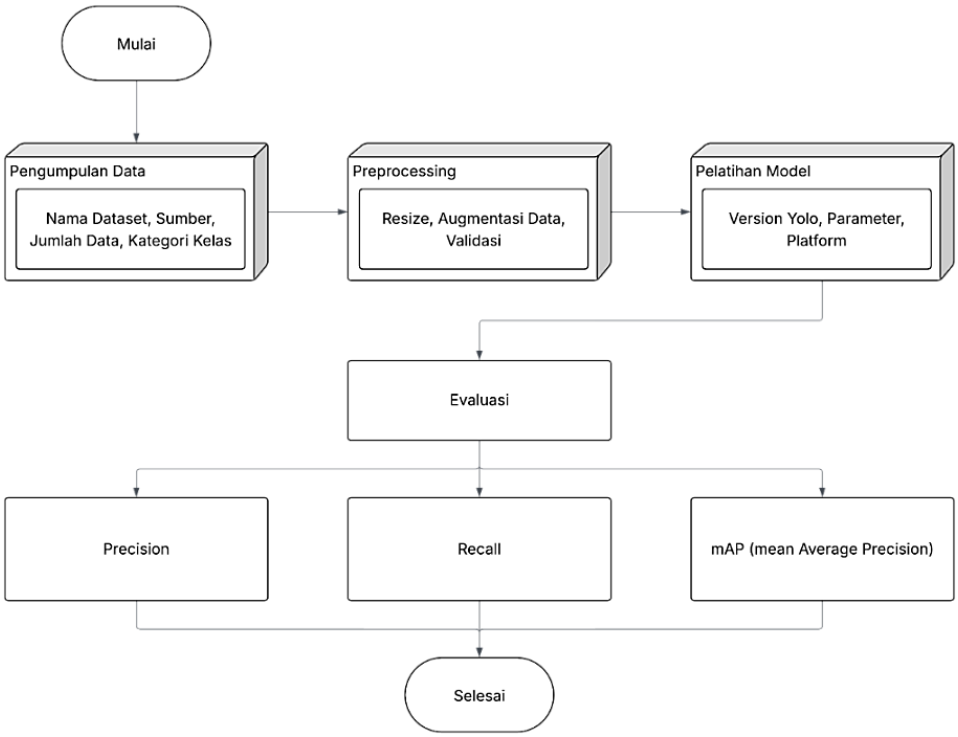
Penelitian terkait deteksi penggunaan helm keselamatan juga telah dilakukan menggunakan berbagai pendekatan berbasis deep learning. Beberapa studi menggunakan *You Only Look Once* v5 (*YOLO*v5) menunjukkan kemampuan deteksi yang baik dalam kondisi tertentu, namun masih menghadapi keterbatasan dalam mendeteksi objek berukuran kecil, kondisi pencahayaan rendah, serta objek yang tertutup sebagian. Sementara itu, metode berbasis SSD dan *Faster* R-CNN cenderung memiliki akurasi yang cukup tinggi, tetapi kurang optimal dalam aspek kecepatan sehingga kurang sesuai untuk implementasi *real-time* di lingkungan konstruksi yang dinamis. Selain itu, pendekatan berbasis EfficientDet menawarkan keseimbangan antara akurasi dan efisiensi, namun membutuhkan sumber daya komputasi yang relatif lebih besar.

Meskipun berbagai penelitian telah dilakukan, sebagian besar studi masih berfokus pada satu jenis model atau hanya membandingkan model generasi sebelumnya, sehingga belum memberikan gambaran komprehensif mengenai perbandingan performa antarversi terbaru algoritma *YOLO*, khususnya *YOLOv8*, *YOLOv9*, dan *YOLOv11*. Padahal, setiap versi memiliki karakteristik arsitektur yang berbeda yang berpotensi memengaruhi akurasi, stabilitas, dan efisiensi komputasi dalam deteksi objek secara *real-time*. Oleh karena itu, diperlukan penelitian komparatif yang mampu mengevaluasi ketiga model tersebut secara sistematis untuk mengidentifikasi model yang paling optimal sesuai kebutuhan implementasi di lapangan. Algoritma *YOLO* menjadi salah satu pendekatan yang paling relevan karena kemampuannya dalam melakukan deteksi objek secara *real-time* dengan tingkat akurasi yang kompetitif. Versi terbaru *YOLOv8*, *YOLOv9*, dan *YOLOv11* menunjukkan peningkatan signifikan dalam hal presisi, efisiensi komputasi, serta kemampuan mendeteksi objek berukuran kecil seperti helm keselamatan [12], [13].

Berdasarkan latar belakang tersebut, penelitian ini dilakukan untuk membandingkan performa algoritma *YOLOv8*, *YOLOv9*, dan *YOLOv11* dalam mendeteksi penggunaan helm keselamatan pada lingkungan konstruksi. Evaluasi dilakukan menggunakan *dataset Hard Hats Computer Vision Project* dengan metrik *precision*, *recall*, dan *mean Average Precision* (*mAP*) [14], [15], [16]. Hasil penelitian ini diharapkan dapat memberikan rekomendasi model terbaik yang paling akurat dan efisien, serta berkontribusi dalam meningkatkan efektivitas pemantauan K3 dan menurunkan angka pelanggaran keselamatan kerja di sektor konstruksi.

2. METODELOGI PENELITIAN

Penelitian ini menerapkan pendekatan eksperimen rekayasa perangkat lunak untuk membandingkan performa tiga versi algoritma *YOLO*, yaitu *YOLOv8* sebagai versi yang stabil dan banyak digunakan, *YOLOv9* yang memiliki peningkatan arsitektur untuk efisiensi dan akurasi deteksi, serta *YOLOv11* sebagai versi terbaru dengan peningkatan fitur, efisiensi komputasi, dan kemampuan deteksi objek kecil secara *real-time*. Seluruh model dilatih dan dievaluasi menggunakan *dataset* yang sama dengan parameter pelatihan yang disesuaikan guna memastikan keadilan evaluasi. *Dataset* dibagi ke dalam tiga subset, yaitu data latih, validasi, dan uji dengan proporsi umum 70:20:10 untuk menjaga keseimbangan proses pembelajaran dan pengujian model. Pengukuran performa dilakukan berdasarkan metrik *precision*, *recall*, serta *mAP*, termasuk *mAP*@50 dan *mAP*@50–95 untuk memberikan evaluasi yang lebih komprehensif terhadap akurasi deteksi pada berbagai tingkat *Intersection over Union* (*IoU*). Metodologi penelitian dapat dilihat pada Gambar 1.



Gambar 1. Metodologi Penelitian

2.1. Pengumpulan Data

Tahap awal dalam penelitian ini dimulai dengan pengumpulan data yang berfungsi sebagai dasar dalam membangun sistem deteksi penggunaan helm. *Dataset* yang digunakan adalah *Hard Hats Computer Vision Project* yang diperoleh dari *Roboflow Universe*. *Dataset* ini terdiri atas kurang lebih 20.000 gambar dengan dua kategori kelas, yaitu helm dan tanpa helm. Data telah dilengkapi dengan anotasi berformat kompatibel dengan *YOLOv8*, v9, v11 (XML, JSON, dan TXT) serta memiliki resolusi yang bervariasi dengan mayoritas berstandar HD. *Dataset* ini telah disusun dengan pembagian data latih, validasi, dan pengujian secara proporsional sehingga mendukung proses eksperimen yang sistematis, terukur, dan tidak bias.

2.2. *Preprocessing* Data

Tahap berikutnya adalah preprocessing untuk memastikan kualitas data siap digunakan dalam pelatihan model. Gambar pada *dataset* terlebih dahulu dilakukan resize ke ukuran standar input *YOLO*, yaitu 640×640 piksel, dan dinormalisasi agar distribusi pixel seragam [17], [18]. Selanjutnya dilakukan augmentasi data berupa flip horizontal, rotasi, scaling, serta penyesuaian tingkat kecerahan dan kontras guna meningkatkan variasi data sehingga model lebih robust terhadap kondisi nyata di lapangan. Setelah itu, anotasi pada gambar diperiksa kembali melalui tahap validasi untuk memastikan *bounding box* dan label kelas sesuai dengan objek yang ditandai [19].

2.3. Pelatihan Model

Pada tahap pelatihan model, dilakukan eksperimen dengan melatih tiga versi *YOLO*, yaitu *YOLOv8*, *YOLOv9*, dan *YOLOv11*, menggunakan *dataset* yang sama. Tujuan dari pelatihan multi-versi ini adalah untuk membandingkan performa masing-masing arsitektur dalam mendeteksi pekerja yang menggunakan atau tidak menggunakan helm. Parameter pelatihan ditetapkan sebanyak 25, 30, 50, dan 60 *epoch* untuk pengujian akhir dengan batch size menyesuaikan dengan kapasitas GPU (umumnya 16–32), serta learning rate awal sebesar 0.01 dengan *scheduler* adaptif untuk menjaga stabilitas konvergensi. *Optimizer* yang digunakan adalah *Stochastic Gradient Descent* (*SGD*) dengan momentum standar. Seluruh pelatihan dilakukan pada platform Google Colab dengan GPU Tesla T4 (jika tersedia) untuk mempercepat komputasi dan memastikan efisiensi proses training.

2.4. Evaluasi

Tahap terakhir adalah evaluasi model. Evaluasi dilakukan dengan menggunakan tiga metrik utama, yaitu *precision*, *recall*, dan *mAP*.

Precision mengukur ketepatan model dalam mendeteksi objek helm atau tanpa helm, dihitung menggunakan Persamaan 1.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

di mana *TP* adalah *True Positive* dan *FP* adalah *False Positive* [8], [20].

Sementara itu, *recall* menilai sejauh mana model dapat menangkap seluruh objek yang ada dalam gambar, dengan Persamaan 2.

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

di mana *FN* adalah *False Negative* [8], [12], [20].

Adapun *mAP* digunakan untuk memberikan skor rata-rata akurasi deteksi pada berbagai nilai ambang batas *IoU* [6], [21], umumnya dinyatakan dalam Persamaan 3.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} \quad AP_i \quad (3)$$

dengan AP_imerupakan *Average Precision* pada kelas ke-i dan *N* Jumlah kelas yang dievaluasi.

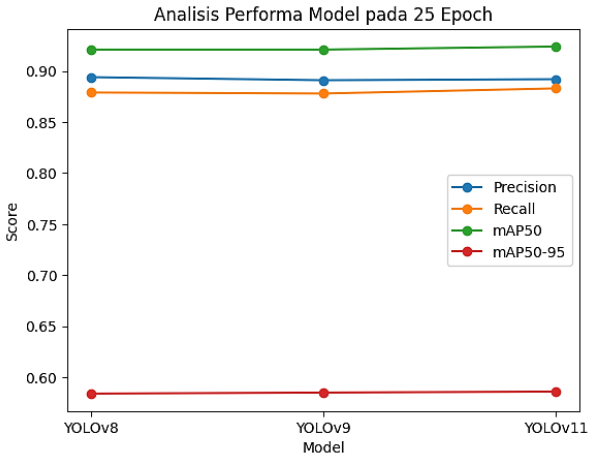
Penggunaan ketiga metrik ini memberikan gambaran komprehensif mengenai performa setiap model *YOLO* yang diuji, sehingga dapat ditentukan versi model yang paling optimal dalam penerapan deteksi keselamatan kerja, khususnya penggunaan helm di lingkungan konstruksi [15], [22], [23], [24], [25].

3. HASIL DAN ANALISIS

Analisis data dalam penelitian ini didasarkan pada hasil pengujian model *YOLOv8*, *YOLOv9*, dan *YOLOv11* dengan variasi jumlah *epoch* 25, 30, dan 50 sebagai tahap pengujian utama, serta 60 *epoch* sebagai tahap pengujian lanjutan. Pengujian ini bertujuan untuk mengevaluasi pengaruh jumlah *epoch* terhadap performa dan stabilitas model deteksi objek. Parameter evaluasi yang digunakan meliputi *precision* untuk mengukur ketepatan prediksi, *recall* untuk menilai kemampuan model dalam mendeteksi seluruh objek yang relevan, serta *mAP* yang direpresentasikan oleh *mAP*@50 dan *mAP*@50–95 sebagai indikator utama kualitas deteksi dan kemampuan generalisasi model. Selain aspek akurasi, waktu pelatihan juga dianalisis sebagai indikator efisiensi komputasi. Dengan demikian, analisis yang dilakukan tidak hanya menitikberatkan pada performa deteksi, tetapi juga mempertimbangkan keseimbangan antara akurasi dan efisiensi pelatihan model.

3.1. Performa Model *YOLO* Pada 25 *Epoch*

Pada 25 *epoch*, ketiga model *YOLO* telah menunjukkan kemampuan deteksi objek yang cukup baik meskipun masih berada pada tahap awal proses pelatihan. *YOLOv8* menghasilkan nilai *precision* sebesar 0,894, yang menunjukkan bahwa sebagian besar prediksi *bounding box* yang dihasilkan model sudah tepat dan kesalahan deteksi (*false positive*) masih relatif rendah. Namun, nilai *recall* sebesar 0,879 menunjukkan bahwa masih ada beberapa objek yang belum terdeteksi oleh model. Ini menunjukkan bahwa meskipun *YOLOv8* cukup selektif dan akurat dalam memprediksi objek, cakupan deteksinya masih dapat ditingkatkan.



Gambar 2. Analisis Performa Model *YOLO* pada 25 *Epoch*

Pada Gambar 2 nilai *mAP*@50 sebesar 0,921 pada *YOLOv8* menunjukkan bahwa model telah mampu melakukan deteksi dengan baik pada ambang *IoU* yang relatif longgar, sementara nilai *mAP*@50–95 sebesar 0,584 menunjukkan bahwa nilai performa model mengalami menurun ketika diuji. Perbedaan yang cukup

besar antara *mAP*@50 dan *mAP*@50–95 ini mengindikasikan bahwa kualitas lokalisasi *bounding box* masih perlu ditingkatkan.

Tabel 1. Data Performa Model *YOLO* pada 25 *Epoch*

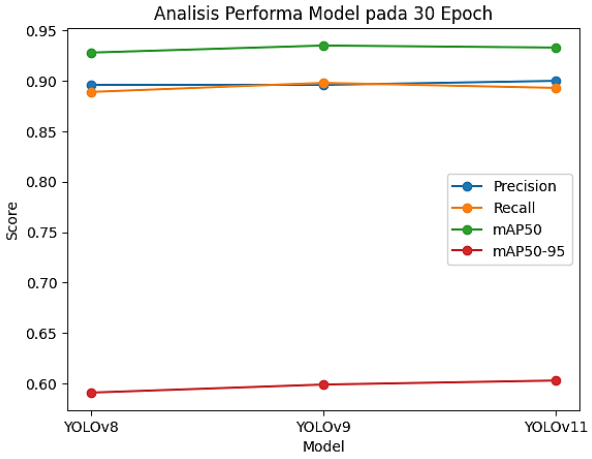
Model *YOLO*	*Precision*	*Recall*	*mAP*50	*MAP*50-95	Waktu
YOLO v8	0.894	0.879	0.921	0.584	1.735 h
YOLO v9	0.891	0.878	0.921	0.585	2.289 h
YOLO v11	0.892	0.883	0.924	81,28	1.875 h

Berdasarkan Tabel 1 *YOLOv9* pada 25 *epoch* memperoleh nilai tren *precision* sebesar 0,891 dan *recall* sebesar 0,878, dengan *mAP*@50 sebesar 0,921 dan *mAP*@50–95 sebesar 0,585. Nilai *precision* dan *recall* yang hampir sama dengan *YOLOv8* menunjukkan bahwa kedua model memiliki kemampuan deteksi awal yang sebanding. Namun, dengan arsitektur yang lebih kompleks, *YOLOv9* belum mampu menunjukkan peningkatan performa yang signifikan pada *epoch* rendah. Hal ini mengindikasikan bahwa keunggulan arsitektur *YOLOv9* baru akan terlihat setelah proses pelatihan yang lebih panjang.

Sementara itu, *YOLOv11* menunjukkan performa paling baik pada 25 *epoch* dengan nilai *precision* 0,892, *recall* 0,883, *mAP*@50 0,924, dan *mAP*@50–95 0,586. Nilai *recall* yang lebih tinggi menunjukkan bahwa *YOLOv11* mampu mendeteksi lebih banyak objek dibandingkan dengan dua model lainnya. Selain itu, nilai *mAP* yang lebih tinggi menunjukkan bahwa *YOLOv11* memiliki kualitas lokalisasi *bounding box* yang lebih baik dan lebih konsisten pada berbagai ambang *IoU*. Dengan demikian, pada *epoch* rendah, *YOLOv11* menunjukkan kemampuan generalisasi yang lebih baik meskipun selisih performanya masih relatif kecil.

3.2. Performa Model *YOLO* Pada 30 *Epoch*

Pada 30 *epoch*, ketiga model menunjukkan peningkatan performa yang lebih jelas dibandingkan dengan *epoch* sebelumnya, menandakan bahwa proses pembelajaran telah berjalan lebih optimal. Peningkatan ini terlihat pada seluruh parameter evaluasi, yaitu *precision*, *recall*, dan *mAP*, yang secara kolektif merepresentasikan ketepatan, cakupan, dan kualitas deteksi objek. Hal ini menunjukkan bahwa jumlah *epoch* yang lebih besar memungkinkan model untuk mempelajari pola visual secara lebih mendalam dan stabil.



Gambar 3. Analisis Performa Model *YOLO* pada 30 *Epoch*

Pada Gambar 3 *YOLOv8* mengalami peningkatan nilai *precision* menjadi 0,896 dan *recall* menjadi 0,889, yang menunjukkan kalau model tidak hanya semakin akurat dalam memprediksi objek, tetapi juga mampu mendeteksi lebih banyak objek yang relevan. Nilai *mAP*@50 0,928 dan *mAP*@50–95 0,591 memperlihatkan adanya peningkatan kualitas deteksi pada berbagai tingkat *IoU*. Meskipun peningkatan performanya tidak terlalu drastis, hasil ini menunjukkan bahwa *YOLOv8* memiliki karakteristik pembelajaran yang stabil dan bertahap.

Tabel 2. Data Performa Model *YOLO* pada 30 *Epoch*

Model *YOLO*	*Precision*	*Recall*	*mAP*50	*MAP*50-95	Waktu
YOLO v8	0.896	0.889	0.928	0.591	2.533 h
YOLO v9	0.896	0.898	0.935	0.599	3.556 h
YOLO v11	0.9	0.893	0.933	0.603	2.766 h

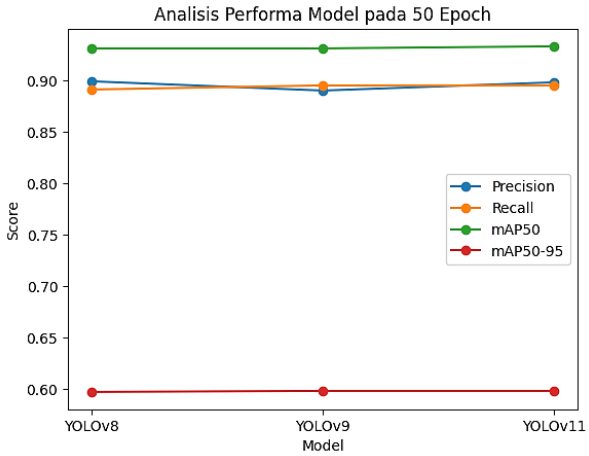
Berdasarkan Tabel 2, dari sisi kualitas deteksi, *YOLOv9* menunjukkan peningkatan yang paling menonjol pada 30 *epoch*. Model ini menghasilkan nilai *precision* sebesar 0,896 dan *recall* sebesar 0,898, yang merupakan nilai *recall* tertinggi di antara ketiga model. Nilai *mAP*@50 mencapai 0,935 mengindikasikan bahwa *YOLOv9* sangat efektif dalam mendeteksi objek ketika toleransi tumpang tindih relatif longgar. Namun, nilai *mAP*@50–95 sebesar 0,599 menunjukkan bahwa performa model menurun, yang mengindikasikan keterbatasan pada presisi lokalisasi *bounding box*.

Sementara itu, *YOLOv11* menunjukkan performa yang paling seimbang pada 30 *epoch* dengan nilai *precision* tertinggi 0,900, *recall* 0,893, *mAP*@50 0,933, dan *mAP*@50–95 0,603, yang merupakan nilai tertinggi untuk *mAP*@50–95 pada tahap ini. Nilai tersebut menunjukkan bahwa *YOLOv11* mampu mempertahankan kualitas deteksi yang baik dari sisi ketepatan prediksi maupun konsistensi lokalisasi objek. Keseimbangan antara *precision* dan *recall* yang dicapai *YOLOv11* menandakan bahwa model ini tidak terlalu selektif maupun terlalu permisif dalam mendeteksi objek.

Secara keseluruhan, hasil pada 30 *epoch* menunjukkan bahwa peningkatan *epoch* memberikan dampak positif terhadap performa ketiga model. Namun, jika ditinjau dari kombinasi *precision*, *recall*, dan *mAP*, *YOLOv11* menunjukkan performa paling optimal, sedangkan *YOLOv9* unggul pada sensitivitas deteksi dan *YOLOv8* unggul dalam stabilitas pembelajaran.

3.3. Performa Model *YOLO* pada 50 *Epoch*

Pada 50 *epoch*, peningkatan performa model mulai menunjukkan kecenderungan melambat, yang mengindikasikan bahwa proses pelatihan telah mendekati kondisi konvergen. Pada tahap ini, penambahan *epoch* tidak lagi memberikan peningkatan performa yang signifikan seperti pada fase sebelumnya. Hal ini penting untuk dianalisis karena berkaitan dengan efisiensi pelatihan dan potensi terjadinya overfitting.



Gambar 4. Analisis Performa Model *YOLO* pada 50 *Epoch*

Pada Gambar 4 *YOLOv8* mencatat nilai *precision* 0,899 dan *recall* 0,891, yang menunjukkan jika model telah mencapai kestabilan dalam menghasilkan prediksi yang akurat dan konsisten. Nilai *mAP*@50 0,931 dan *mAP*@50–95 0,597 menunjukkan kalau peningkatan kualitas deteksi masih terjadi, namun dengan selisih yang relatif kecil dibandingkan 30 *epoch*. Hal ini menunjukkan bahwa *YOLOv8* cenderung mencapai performa optimalnya secara bertahap tanpa fluktuasi yang signifikan.

Tabel 3. Data Performa Model *YOLO* pada 50 *Epoch*

Model *YOLO*	*Precision*	*Recall*	*mAP*50	*MAP*50-95	Waktu
YOLO v8	0.899	0.891	0.931	0.597	3.691 h
YOLO v9	0.89	0.895	0.931	0.598	4.580 h
YOLO v11	0.898	0.895	0.933	0.598	3.679 h

Berdasarkan Tabel 3 pada *YOLOv9*, nilai *precision* 0,890 menunjukkan sedikit penurunan dibandingkan 30 *epoch*, sementara nilai *recall* meningkat menjadi 0,895. Kondisi ini mengindikasikan bahwa *YOLOv9* menjadi lebih agresif dalam mendeteksi objek, namun dengan konsekuensi meningkatnya kesalahan prediksi. Nilai *mAP*@50 0,931 dan *mAP*@50–95 0,598 menunjukkan bahwa meskipun performa deteksi masih meningkat, kualitas lokalisasi *bounding box* tidak mengalami peningkatan yang signifikan.

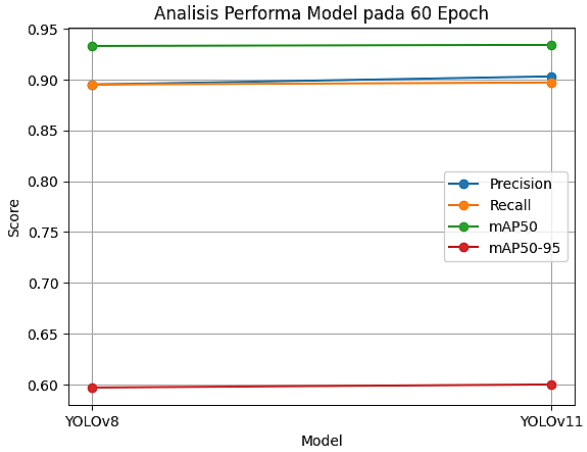
Sementara itu, *YOLOv11* menunjukkan performa yang paling stabil pada 50 *epoch* dengan nilai *precision* 0,898, *recall* 0,895, *mAP*@50 0,933, dan *mAP*@50–95 0,598. Nilai-nilai ini relatif mendekati hasil pada 30 *epoch*, yang mengindikasikan bahwa *YOLOv11* telah mencapai titik optimal dan tidak memperoleh

manfaat signifikan dari penambahan *epoch* lebih lanjut. Stabilitas ini menunjukkan bahwa *YOLOv11* memiliki kemampuan generalisasi yang baik dan tidak mudah mengalami degradasi performa.

Secara keseluruhan, hasil pada 50 *epoch* menunjukkan bahwa penambahan *epoch* setelah titik tertentu tidak selalu memberikan peningkatan performa yang sebanding dengan biaya komputasi yang dikeluarkan. *YOLOv11* tetap menunjukkan keseimbangan terbaik antara *precision*, *recall*, dan *mAP*, sedangkan *YOLOv8* menunjukkan kestabilan performa, dan *YOLOv9* menunjukkan kecenderungan trade-off antara *recall* dan *precision*. Temuan ini menegaskan pentingnya pemilihan jumlah *epoch* yang tepat untuk memperoleh performa optimal tanpa pemborosan sumber daya komputasi.

3.4. Analisis Pengujian Lanjutan pada 60 *Epoch*

Pengujian lanjutan pada 60 *epoch* dilakukan untuk mengevaluasi stabilitas performa *YOLOv8* dan *YOLOv11*, yang pada tahap sebelumnya menunjukkan performa paling kompetitif berdasarkan parameter *precision*, *recall*, dan *mAP*. Tujuan pengujian ini untuk menilai pengaruh penambahan *epoch* terhadap peningkatan akurasi deteksi serta efisiensi pelatihan model.



Gambar 5. Analisis Performa Model *YOLO* pada 60 *Epoch*

Pada Gambar 5 dengan 60 *epoch*, *YOLOv8* memperoleh nilai *precision* 0,895 dan *recall* 0,895, dengan *mAP*@50 0,933 dan *mAP*@50–95 0,597. Nilai tersebut menunjukkan kalau *YOLOv8* mampu mempertahankan performa deteksi yang stabil dan konsisten pada jumlah *epoch* yang lebih tinggi. Meskipun peningkatan nilai *mAP*@50–95 tidak signifikan dibandingkan pengujian sebelumnya, performa yang dicapai mengindikasikan bahwa *YOLOv8* telah mencapai kondisi konvergen dengan tingkat akurasi yang tetap tinggi. Waktu pelatihan yang dibutuhkan sebesar 4,563 jam, menunjukkan efisiensi komputasi yang masih kompetitif.

Tabel 4. Data Performa Model *YOLO* pada 60 *Epoch*

Model *YOLO*	*Precision*	*Recall*	*mAP*50	*MAP*50-95	Waktu
YOLO v8	0.895	0.895	0.933	0.597	4.563 h
YOLO v11	0.903	0.897	0.934	0.6	4.522 h

Sementara itu pada Tabel 4, *YOLOv11* menunjukkan peningkatan nilai *precision* 0,903 dan *recall* 0,897, dengan *mAP*@50 0,934 dan *mAP*@50–95 0,600. Peningkatan pada nilai *mAP*@50–95 menunjukkan adanya perbaikan kemampuan generalisasi model pada berbagai ambang *IoU*. Waktu pelatihan *YOLOv11* pada tahap ini adalah 4,522 jam, yang relatif sebanding dengan *YOLOv8*.

Berdasarkan hasil pengujian tersebut, dapat disimpulkan bahwa kedua model menunjukkan performa yang sangat kompetitif pada 60 *epoch*. *YOLOv8* unggul dalam hal stabilitas dan konsistensi performa, dengan nilai *mAP* yang tinggi dan efisiensi pelatihan yang tetap terjaga. Di sisi lain, *YOLOv11* menunjukkan peningkatan bertahap pada aspek generalisasi, meskipun selisih performanya relatif kecil. Dengan demikian, pemilihan model terbaik pada tahap ini dapat disesuaikan dengan kebutuhan aplikasi, di mana *YOLOv8* lebih sesuai untuk skenario yang mengutamakan stabilitas dan efisiensi, sedangkan *YOLOv11* relevan untuk aplikasi yang memerlukan peningkatan generalisasi secara bertahap.

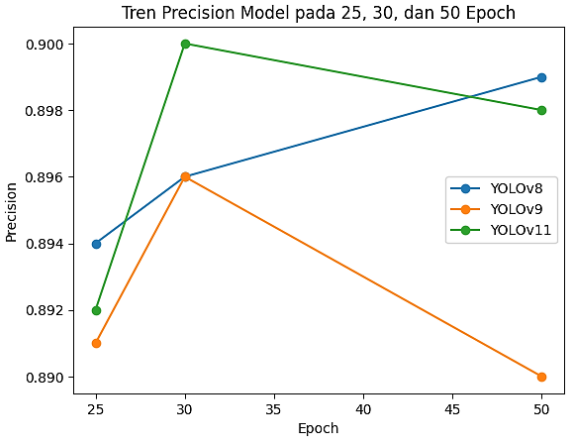
3.5. Diskusi secara umum tentang *YOLO*

Untuk memberikan gambaran komprehensif mengenai perbedaan performa antar model, ringkasan hasil terbaik dari masing-masing model disajikan pada Tabel 5.

Tabel 5. Ringkasan Performa Terbaik Model

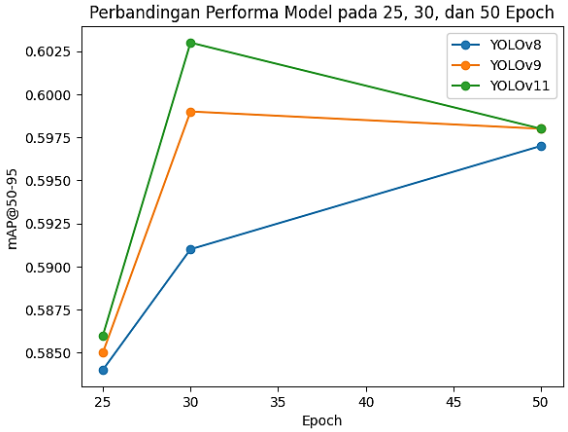
Model *YOLO*	*Epoch* Terbaik	*Precision*	*Recall*	*mAP*50	*MAP*50-95	Waktu
YOLO v8	50–60	0.89–0.895	0.895	0.933	0.597	4.563 h
YOLO v9	50	0.89	0.895	0.931	0.598	4.580 h
YOLO v11	60	0.903	0.897	0.934	0.6	4.522 h

Berdasarkan hasil tersebut, perbedaan performa antar model dapat dijelaskan melalui karakteristik arsitektur dan pola pembelajaran masing-masing model. *YOLOv8* menunjukkan performa yang relatif stabil sejak *epoch* menengah hingga akhir. Hal ini mengindikasikan bahwa arsitektur *YOLOv8* memiliki keseimbangan yang baik antara kompleksitas model dan kemampuan generalisasi, sehingga mampu mencapai konvergensi lebih cepat tanpa memerlukan iterasi pelatihan yang terlalu panjang. Stabilitas ini menjadikan *YOLOv8* lebih efisien dalam hal waktu pelatihan dan penggunaan sumber daya komputasi, serta cocok untuk implementasi pada sistem *real-time* dengan keterbatasan hardware. Grafik tren *precision* dapat dilihat pada Gambar 6.



Gambar 6. Grafik Tren *Precision*

Sebaliknya, *YOLOv9* menunjukkan peningkatan performa yang lebih lambat dan kurang signifikan dibandingkan model lainnya. Hal ini dapat disebabkan oleh kompleksitas arsitektur yang lebih tinggi sehingga membutuhkan penyesuaian hiperparameter yang lebih spesifik agar dapat mencapai performa optimal. Dalam penelitian ini, konfigurasi pelatihan yang diseragamkan untuk semua model kemungkinan belum sepenuhnya optimal bagi *YOLOv9*, sehingga berdampak pada efisiensi dan performa yang dihasilkan. Kondisi ini menunjukkan bahwa peningkatan arsitektur tidak selalu berbanding lurus dengan peningkatan performa jika tidak diimbangi dengan tuning yang tepat. Grafik tren *mAP*@50–95 dapat dilihat pada Gambar 7.



Gambar 7. Grafik Tren *mAP*@50–95

Di sisi lain, *YOLOv11* menunjukkan peningkatan performa yang lebih signifikan pada *epoch* tinggi, terutama pada *epoch* 60. Hal ini mengindikasikan bahwa model dengan arsitektur yang lebih kompleks

membutuhkan proses pembelajaran yang lebih panjang untuk dapat mengekstraksi fitur secara optimal. Peningkatan ini juga menunjukkan bahwa *YOLOv11* memiliki kapasitas representasi yang lebih besar, sehingga mampu mencapai akurasi yang lebih tinggi dibandingkan model lainnya. Namun, konsekuensi dari hal tersebut adalah kebutuhan komputasi yang lebih besar serta waktu pelatihan yang lebih lama.

Dengan demikian, hasil penelitian ini menunjukkan bahwa terdapat trade-off antara akurasi dan efisiensi komputasi. *YOLOv8* lebih unggul dalam hal stabilitas dan efisiensi, sehingga lebih sesuai untuk implementasi *real-time*, sedangkan *YOLOv11* lebih unggul dalam hal akurasi dan cocok digunakan pada skenario yang membutuhkan tingkat presisi tinggi. Sementara itu, *YOLOv9* dalam penelitian ini belum menunjukkan keunggulan yang signifikan, sehingga memerlukan eksplorasi lebih lanjut terkait konfigurasi pelatihan untuk memaksimalkan potensinya.

3.6. Analisis Pengaruh *Epoch* terhadap Performa

Peningkatan jumlah *epoch* memberikan dampak positif terhadap performa model, terutama pada tahap awal pelatihan. Pada *epoch* 25 hingga 50, terjadi peningkatan signifikan pada nilai *precision*, *recall*, dan *mAP*. Namun, setelah mencapai *epoch* 60, peningkatan performa cenderung tidak signifikan, terutama pada *YOLOv8*, yang menunjukkan indikasi saturasi performa.

Hal ini menunjukkan bahwa penambahan *epoch* yang berlebihan tidak selalu memberikan peningkatan akurasi yang sebanding, melainkan hanya menambah beban komputasi. Oleh karena itu, pemilihan jumlah *epoch* yang optimal menjadi faktor penting dalam efisiensi pelatihan model.

3.7. Analisis Kesalahan (*Error Analysis*)

Analisis kesalahan menunjukkan bahwa meskipun model menghasilkan performa yang tinggi secara kuantitatif, masih terdapat keterbatasan signifikan pada kondisi visual tertentu yang mempengaruhi keandalan deteksi. Penurunan performa paling terlihat pada kondisi pencahayaan rendah, di mana kontras objek helm terhadap latar belakang menjadi tidak optimal sehingga fitur visual sulit diekstraksi secara konsisten oleh model. Selain itu, kasus *occlusion* menjadi tantangan utama, terutama ketika helm tertutup sebagian oleh objek lain seperti tangan pekerja, alat, atau material konstruksi, yang menyebabkan model gagal mengenali bentuk utuh objek. Variasi sudut pandang dan skala objek juga berkontribusi terhadap kesalahan deteksi, khususnya pada posisi ekstrem atau jarak yang terlalu jauh dari kamera.

Di sisi lain, masih ditemukan *false positive* akibat kemiripan warna dan bentuk antara helm dengan objek lain di lingkungan kerja, serta *false negative* yang menunjukkan bahwa model belum sepenuhnya mampu menangkap seluruh objek target. Temuan ini mengindikasikan adanya potensi bias pada *dataset*, terutama terkait distribusi kondisi pencahayaan, sudut pengambilan gambar, dan tingkat keberagaman objek. Oleh karena itu, peningkatan kualitas dan variasi *dataset* melalui teknik augmentasi yang lebih adaptif, seperti simulasi kondisi pencahayaan ekstrem dan occlusi sintetis, serta pengujian pada *dataset* yang lebih representatif terhadap kondisi lapangan, menjadi langkah penting untuk meningkatkan generalisasi model. Selain itu, untuk implementasi pada sistem monitoring *real-time*, perlu dilakukan evaluasi tambahan terhadap performa model dalam kondisi dinamis, termasuk pengukuran *frame rate* (*FPS*) dan latensi deteksi, guna memastikan bahwa sistem tidak hanya akurat tetapi juga responsif dalam mendukung penerapan K3 di lingkungan konstruksi.

4. KESIMPULAN

Penelitian ini dilakukan untuk membandingkan performa algoritma *YOLOv8*, *YOLOv9*, dan *YOLOv11* dalam mendeteksi penggunaan helm keselamatan kerja menggunakan dataset *Hard Hats Computer Vision Project*. Evaluasi dilakukan dengan variasi jumlah *epoch* pelatihan, yaitu 25, 30, dan 50 *epoch* sebagai tahap pengujian utama, serta 60 *epoch* sebagai tahap pengujian lanjutan untuk mengevaluasi karakteristik performa model secara lebih mendalam.

Hasil pengujian menunjukkan bahwa ketiga model mampu mencapai tingkat akurasi yang tinggi dalam mendeteksi objek helm keselamatan. Namun, pada pengujian awal hingga menengah, *YOLOv8* dan *YOLOv11* menunjukkan performa yang lebih stabil dan konsisten dibandingkan dengan *YOLOv9*, baik dari sisi nilai *mAP* maupun efisiensi waktu pelatihan. *YOLOv9* memang menunjukkan peningkatan performa pada kondisi tertentu, tetapi peningkatan tersebut tidak sebanding dengan waktu pelatihan yang dibutuhkan, sehingga model ini dinilai kurang efisien untuk dilanjutkan ke tahap pengujian lanjutan.

Pengujian lanjutan pada 60 *epoch* dilakukan untuk mengevaluasi lebih jauh performa *YOLOv8* dan *YOLOv11* setelah keduanya menunjukkan hasil yang kompetitif pada tahap sebelumnya. Hasil pengujian lanjutan menunjukkan bahwa *YOLOv8* cenderung mengalami saturasi performa, di mana peningkatan jumlah *epoch* tidak lagi memberikan peningkatan signifikan terhadap nilai akurasi. Sebaliknya, *YOLOv11* masih menunjukkan potensi peningkatan performa pada beberapa metrik evaluasi, khususnya *mAP*50–95, yang merepresentasikan kemampuan generalisasi model.

Meskipun *YOLOv11* memiliki potensi peningkatan akurasi pada *epoch* yang lebih tinggi, selisih performa yang dihasilkan tidak terlalu signifikan jika dibandingkan dengan *YOLOv8*. Di sisi lain, *YOLOv8*

menunjukkan keunggulan dalam hal efisiensi komputasi, kestabilan performa, serta waktu inferensi yang lebih cepat. Dengan mempertimbangkan keseimbangan antara akurasi, *precision*, dan efisiensi, *YOLOv8* direkomendasikan sebagai model yang paling optimal untuk implementasi sistem deteksi helm keselamatan, khususnya pada aplikasi *real-time* dengan keterbatasan sumber daya perangkat keras, sementara *YOLOv11* dapat dipertimbangkan sebagai alternatif pada skenario yang menitikberatkan pada pencapaian akurasi maksimum.

Sebagai implikasi penelitian, hasil ini menunjukkan bahwa pemilihan model deteksi objek perlu mempertimbangkan trade-off antara akurasi dan efisiensi komputasi sesuai kebutuhan implementasi. Untuk pengembangan selanjutnya, disarankan dilakukan pengujian pada skenario *real-time* guna mengevaluasi performa model dalam kondisi dinamis, termasuk pengukuran *frame rate* (FPS) dan latensi sistem. Selain itu, penggunaan *dataset* lokal yang lebih merepresentasikan kondisi lingkungan konstruksi di Indonesia juga perlu dipertimbangkan untuk mengurangi potensi bias data dan meningkatkan kemampuan generalisasi model dalam implementasi nyata.

REFERENSI

[1] Kementerian Ketenagakerjaan Republik Indonesia, "Undang-Undang Nomor 1 Tahun 1970," Jan. 1970.

[2] M. Masinaei, H. Asady, and S. J. Shahtaheri, "Risk Factors of Work-Related Head and Neck Injuries: A National Survey," 2022. [Online]. Available: https://creativecommons.org/licenses/by-nc/4.0/

[3] R. Meta, F. Desyana, B. Adhayati, and F. A. Sari, "International Journal Of Occupational Medicine And Public Health Factors Associated With Case Fatality In Work Accidents Examined At The Serang And Cilegon Regional Hospital," 2022. [Online]. Available: https://doi.org/

[4] A. N. Azhari and W. Wahyono, "Automatic Detection of Helmets on Motorcyclists Using Faster - RCNN," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 16, no. 4, p. 357, Oct. 2022, doi: 10.22146/ijccs.68245.

[5] X. Li, T. Hao, F. Li, L. Zhao, and Z. Wang, "Faster R-CNN-LSTM Construction Site Unsafe Behavior Recognition Model," *Applied Sciences (Switzerland)*, vol. 13, no. 19, Oct. 2023, doi: 10.3390/app131910700.

[6] A. Purnama, J. Indra, S. Arum Puspita Lestari, and S. Faisal, "Deteksi Pelanggaran Penggunaan Helm Dengan Metode Ssd Dan Arsitektur Mobilenetv2," 2025.

[7] B. Dai, Y. Nie, W. Cui, R. Liu, and Z. Zheng, "Real-time safety helmet detection system based on improved SSD," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Oct. 2020, pp. 95–99. doi: 10.1145/3421766.3421774.

[8] F. Nursulistio, A. Kurniawardhani, and D. Hatta Fudholi, "Deteksi Objek Masker Menggunakan EfficientDet-Lite3," 2022.

[9] X. Fan, F. Wang, S. Pang, J. Wang, and W. Wang, "Safety helmet wearing detection based on EfficientDet algorithm," 2022. doi: 10.1117/12.2641445.

[10] D. Tarale and P. Chauhan, "Helmet detection system using Mask R-CNN," 2023. [Online]. Available: www.ijres.org

[11] J. Y. Lee, W. S. Choi, and S. H. Choi, "Verification and performance comparison of CNN-based algorithms for two-step helmet-wearing detection," *Expert Syst. Appl.*, vol. 225, Sep. 2023, doi: 10.1016/j.eswa.2023.120096.

[12] K. Patel, V. Patel, V. Prajapati, D. Chauhan, A. Haji, and S. Degadwala, "Safety Helmet Detection Using *YOLO* V8," in *Proceedings - 2023 3rd International Conference on Pervasive Computing and Social Networking, ICPCSN 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 22–26. doi: 10.1109/ICPCSN58827.2023.00012.

[13] H. M. Ahmad and A. Rahimi, "SH17: A dataset for human safety and personal protective equipment detection in manufacturing industry," *Journal of Safety Science and Resilience*, vol. 6, no. 2, pp. 175–185, Jun. 2025, doi: 10.1016/j.jnlssr.2024.09.002.

[14] K. S. Jayanthan and S. Domnic, "An attentive convolutional transformer-based network for road safety," *Journal of Supercomputing*, vol. 79, no. 14, pp. 16351–16377, Sep. 2023, doi: 10.1007/s11227-023-05293-1.

[15] A. Hayat and F. Morgado-Dias, "Deep Learning-Based Automatic Safety Helmet Detection System for Construction Safety," *Applied Sciences (Switzerland)*, vol. 12, no. 16, Aug. 2022, doi: 10.3390/app12168268.

[16] Y. Liu, B. Jiang, H. He, Z. Chen, and Z. Xu, "Helmet wearing detection algorithm based on improved *YOLO*v5," *Sci. Rep.*, vol. 14, no. 1, Dec. 2024, doi: 10.1038/s41598-024-58800-6.

[17] Y. J. Qian and B. Wang, "A new method for safety helmet detection based on convolutional neural network," *PLoS One*, vol. 18, no. 10 October, Oct. 2023, doi: 10.1371/journal.pone.0292970.

[18] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "*YOLO*v4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.10934

[19] L. Baoju *et al.*, "Safety helmet detection methods in heavy machinery factory," *Sci. Rep.*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-02824-z.
[20] R. Kaur, J. Singh, and S. Sharma, "Enhanced Helmet Detection in Surveillance Systems with *YOLO*v6 for Accident Prevention and Safety Compliance," *J. Sci. Ind. Res. (India).*, vol. 84, no. 5, pp. 601–613, May 2025, doi: 10.56042/jsir.v84i5.15782.
[21] J. Redmon and A. Farhadi, "*YOLO*v3: An Incremental Improvement," Apr. 2018, [Online]. Available: http://arxiv.org/abs/1804.02767
[22] H. Peng and Z. Zhang, "Helmet Wearing Recognition of Construction Workers Using Convolutional Neural Network," *Wirel. Commun. Mob. Comput.*, vol. 2022, 2022, doi: 10.1155/2022/4739897.
[23] Kisaezehra, M. U. Farooq, M. A. Bhutto, and A. K. Kazi, "Real-Time Safety Helmet Detection Using *YOLO*v5 at Construction Sites," *Intelligent Automation and Soft Computing*, vol. 36, no. 1, pp. 911–927, 2023, doi: 10.32604/iasc.2023.031359.
[24] K. Han and X. Zeng, "Deep Learning-Based Workers Safety Helmet Wearing Detection on Construction Sites Using Multi-Scale Features," *IEEE Access*, vol. 10, pp. 718–729, 2022, doi: 10.1109/ACCESS.2021.3138407.
[25] M. E. Otgonbold *et al.*, "SHEL5K: An Extended Dataset and Benchmarking for Safety Helmet Detection," *Sensors*, vol. 22, no. 6, Mar. 2022, doi: 10.3390/s22062315.