

Sistem Deteksi Kecacatan Permukaan Rel Kereta Api Berbasis YOLOv8n dan *IP camera*

Arjuna Kurniawan Kususma P.
Engineering Department
Madiun State of Polytechnic
Madiun, Indonesia
arjunakurniawan2003@gmail.com

Finando Ilham Setya P.
Engineering Department
Madiun State of Polytechnic
Madiun, Indonesia
finando.pratama695@gmail.com

Kevin Bayu Pradana
Engineering Department
Madiun State of Polytechnic
Madiun, Indonesia
kevinbayupradana18@gmail.com

Muhammad Dandi Tri Wibowo
Engineering Department
Madiun State of Polytechnic
Madiun, Indonesia
tridand01@gmail.com

Rafli May Sandy
Engineering Department
Madiun State of Polytechnic
Madiun, Indonesia
raflimaysandy@gmail.com

Teuku Muhammad Rafli
Engineering Department
Madiun State of Polytechnic
Madiun, Indonesia
pakpohenjoy@gmail.com

Abstract— Pemantauan kondisi permukaan rel kereta api secara rutin sangat penting untuk menjamin keselamatan operasional dan mencegah kecelakaan. Penelitian ini bertujuan mengembangkan sistem deteksi otomatis untuk cacat permukaan rel seperti *Crack*, *Corrosion*, *Squat*, *Peeling*, dan *Scratch* menggunakan model YOLOv8n yang diintegrasikan dengan *IP camera*. *Dataset* diperoleh melalui dokumentasi lapangan dan sumber publik, kemudian dianotasi menggunakan *Roboflow* dalam format YOLO. Proses pelatihan model dilakukan di *Google Colab* dengan memanfaatkan bobot pralatih YOLOv8n dan teknik *Transfer learning*. *Dataset* dibagi menjadi 70% data latih, 15% validasi, dan 15% pengujian. Evaluasi model dilakukan menggunakan metrik *Precision*, *Recall*, *mAP*, serta kecepatan deteksi (*FPS*). Hasil pelatihan menunjukkan nilai *mAP@0.5* sebesar 92,37% dan *mAP@0.5:0.95* sebesar 61,98%, dengan *Precision* dan *Recall* masing-masing mencapai 94,35% dan 90,27%. Pengujian lapangan menggunakan *IP camera* menunjukkan bahwa sistem mampu mendeteksi sebagian besar cacat dengan akurasi tinggi, meskipun performa masih rendah pada kelas tertentu seperti *Scratch*. Penelitian ini menunjukkan bahwa integrasi YOLOv8n dengan *IP camera* memiliki potensi besar dalam mendukung sistem inspeksi rel otomatis secara *Real-time*.

Keywords— YOLOv8n, deteksi cacat rel, deep learning

I. PENDAHULUAN

Kereta api merupakan salah satu moda transportasi darat yang sangat vital dalam mendukung mobilitas masyarakat dan distribusi barang. Dalam operasionalnya, rel kereta menjadi komponen utama yang harus senantiasa dijaga kualitas dan keamanannya. Cacat atau kerusakan pada permukaan rel seperti retakan (*Crack*), korosi (*Corrosion*), *Squat*, *Peeling*, dan goresan (*Scratch*) dapat menyebabkan kegagalan struktural dan berisiko mengakibatkan kecelakaan [1]. Oleh karena itu, inspeksi dan pemeliharaan rel kereta secara rutin sangat penting untuk memastikan keselamatan dan kelancaran operasi kereta api.

Metode konvensional dalam inspeksi rel umumnya masih bergantung pada pemeriksaan visual oleh petugas lapangan. Namun, pendekatan ini memiliki berbagai keterbatasan seperti ketergantungan pada kondisi cuaca, subjektivitas pengamat, serta tidak efisien dalam menjangkau jaringan rel

yang luas [2]. Untuk mengatasi kendala tersebut, teknologi visi komputer (*Computer Vision*) dan kecerdasan buatan (*Artificial Intelligence*) mulai banyak diterapkan dalam sistem pemantauan infrastruktur perkeretaapian [3].

Dalam beberapa tahun terakhir, pendekatan deep learning, khususnya model deteksi objek berbasis You Only Look Once (YOLO), telah menunjukkan performa yang sangat baik dalam berbagai tugas deteksi objek secara *Real-time* [4], [5]. Versi terbaru, yaitu YOLOv8 dari Ultralytics, membawa peningkatan signifikan dalam hal efisiensi, ukuran model, serta akurasi deteksi [6].

Integrasi YOLOv8 dengan sistem pengambilan gambar secara *Real-time* melalui *IP camera* menawarkan potensi besar dalam pengembangan sistem pemantauan rel otomatis yang mampu mendeteksi cacat dengan cepat dan akurat. Dengan mengadopsi teknik ini, sistem inspeksi rel dapat dilakukan secara kontinu dan otomatis, bahkan di area-area yang sulit dijangkau oleh manusia [7].

Namun, penerapan deep learning dalam domain ini memerlukan pendekatan yang sistematis, mulai dari pengumpulan dan anotasi data gambar, pelatihan model dengan *Dataset* representatif, hingga pengujian dan evaluasi sistem dalam kondisi nyata. Penelitian ini bertujuan untuk membangun dan mengevaluasi sistem deteksi cacat permukaan rel berbasis YOLOv8 yang diintegrasikan dengan *IP camera*, serta mengkaji performanya melalui berbagai metrik evaluasi [8].

II. METODOLOGI

A. Pengumpulan Data

Tahap awal dimulai dengan pengumpulan *Dataset* berupa gambar atau video permukaan rel kereta api. Data dapat diperoleh dari beberapa sumber:

1. Dokumentasi lapangan menggunakan *IP camera* atau kamera biasa.
2. *Dataset* publik dari penelitian terdahulu atau situs seperti Kaggle.

Gambar yang dikumpulkan mencerminkan berbagai kondisi permukaan rel, *Crack*, *Corrosion*, *Squat*, *Peeling*,

Scratch, dan *Normal*. Pembagian jumlah data antar kelas terlihat pada Tabel 2. 1.

Tabel 2. 1 Jumlah Data Antar Kelas

Nama Kelas	Jumlah Data
<i>Crack</i> ,	304
<i>Corrosion</i>	547
<i>Squat</i>	796
<i>Peeling</i>	1840
<i>Scratch</i>	113
<i>Normal</i>	220

B. Anotasi Data

Gambar yang dikumpulkan kemudian dianotasi untuk menandai lokasi dan jenis cacat pada permukaan rel. Proses anotasi dilakukan menggunakan *Roboflow*. Setiap objek kecacatan diberikan *Label* yang sesuai, yaitu *Crack*, *Corrosion*, *Squat*, *Peeling*, *Scratch*, dan *Normal*. Hasil anotasi disimpan dengan YOLOv8n dalam format *Labelling* .txt untuk menyimpan *Bounding box* (*Label*, *x_center*, *y_center*, *width*, *height*) untuk setiap gambar. .yaml digunakan untuk menyimpan struktur *Dataset*, path ke folder gambar dan *Label*, serta daftar nama kelas.

C. Pelatihan Model (*Training*)

Pada tahap ini, dilakukan pelatihan model deteksi objek menggunakan YOLOv8n dari *Ultralytics*. Model YOLOv8n dipilih karena memiliki ukuran yang ringan dan efisien, sehingga cocok untuk dijalankan di *Google Colab*.

Pelatihan dimulai dengan memanfaatkan model YOLOv8n *PreTrained weights* dari *Ultralytics*. Teknik *Transfer learning* ini bertujuan untuk mempercepat proses pelatihan dan meningkatkan akurasi, karena model telah memiliki pengetahuan awal dari pelatihan sebelumnya.

Pembagian *Dataset* dilakukan sebagai berikut, 70% untuk data pelatihan (*Train*), 15% untuk data validasi (*valid*), 15% untuk data pengujian (*test*).

Pelatihan model dilakukan di *Google Colab* menggunakan pustaka *ultralytics*. Parameter *imgsz=640* digunakan untuk menetapkan ukuran input gambar, *Batch=6* mengatur jumlah *Batch* per iterasi, dan *amp=Trn* mengaktifkan *Mixed precision Training* untuk efisiensi memori. Model disimpan setiap 10 *Epoch* melalui parameter *save_period=10*.

Selama pelatihan, dilakukan pemantauan terhadap metrik seperti *loss*, *Precision*, *Recall*, dan *mAP* untuk menghindari *overfitting* dan *underfitting*.

D. Integrasi dengan *IP camera*

Model YOLOv8n yang telah dilatih diintegrasikan dengan input video dari *IP camera* untuk mendeteksi cacat secara *Real-time* atau *Near real-time*. Proses ini melibatkan:

1. Akses stream video dari *IP camera* menggunakan *OpenCV* (*cv2.VideoCapture(URL)*).
2. Proses frame-by-frame untuk mendeteksi objek menggunakan model YOLOv8n.

Menampilkan hasil deteksi (*Bounding box* dan *Label*) secara langsung atau menyimpannya untuk analisis lanjutan.

E. Evaluasi Sistem

Evaluasi dilakukan untuk mengukur kinerja model terhadap data uji. Beberapa metrik yang digunakan antara lain:

1. *Precision* dan *Recall*: Mengukur akurasi dan kelengkapan deteksi.
2. *mAP* (mean Average *Precision*): Menilai performa model secara keseluruhan dalam deteksi multi-kelas.
3. Kecepatan Deteksi (*FPS*): Untuk memastikan sistem dapat berjalan secara *Real-time*.
4. *Confusion matrix*: Untuk melihat kesalahan klasifikasi antara kelas-kelas cacat.

Evaluasi juga dapat mencakup uji coba langsung di lapangan untuk memastikan sistem dapat bekerja secara andal dalam kondisi nyata.

III. HASIL DAN PEMBAHASAN

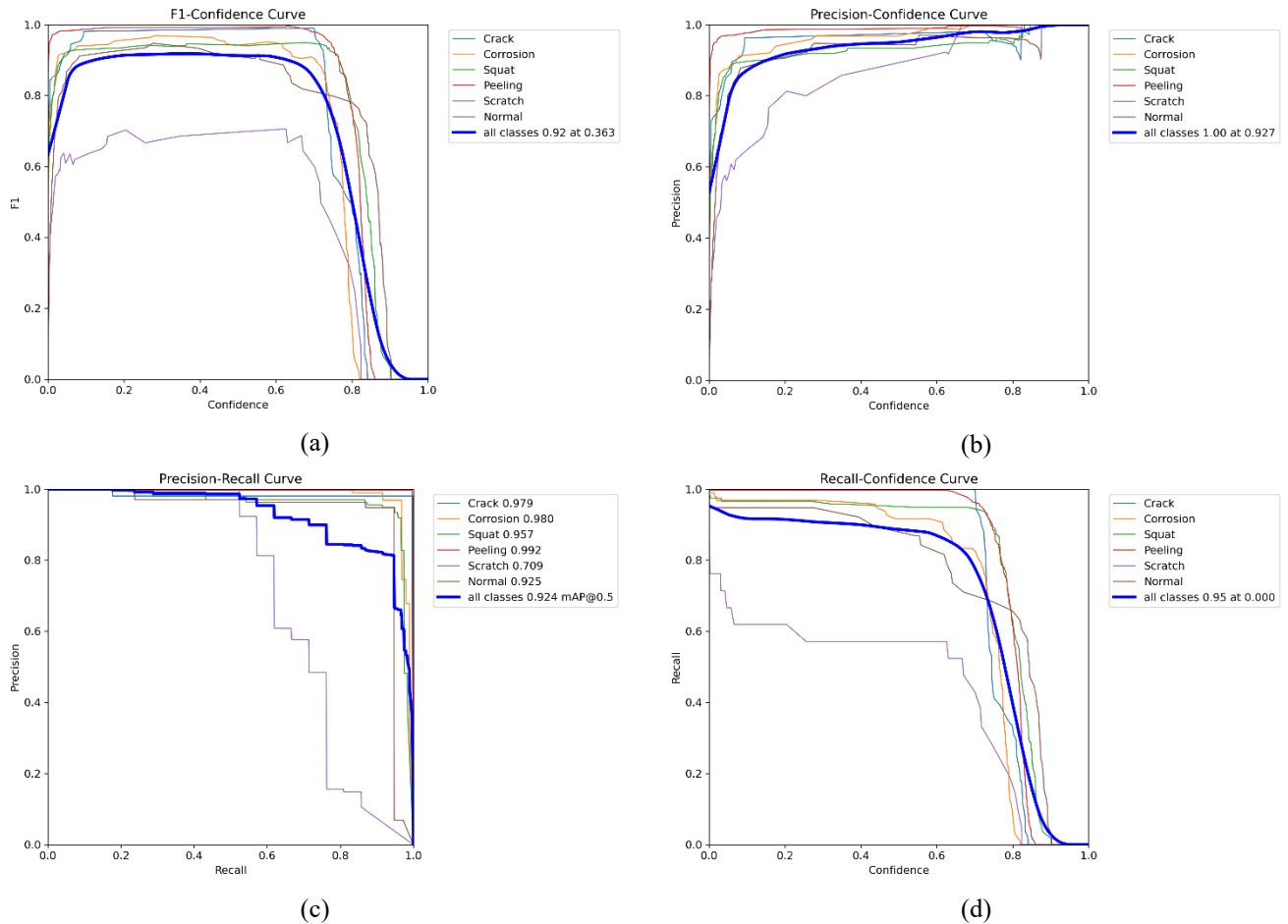
A. Hasil Pelatihan Model

Model YOLOv8n dilatih menggunakan *Dataset* yang telah dianotasi dengan berbagai jenis kecacatan permukaan rel. Proses pelatihan dilakukan selama 100 *Epochs* dengan resolusi input 640×640 piksel. *Dataset* dibagi menjadi 70% data *Train*, 15% data validasi dan 15% data test. Hasil ringkasan pelatihan model terlihat pada Tabel 3. 1.

Tabel 3. 1 Pelatihan Model

Parameter	Result
<i>Loss</i> deteksi (box loss)	0.5 (menurun stabil)
<i>Loss</i> klasifikasi	0.5
<i>Precision</i>	94,35%
<i>Recall</i>	90,27%
<i>mAP@0.5</i>	92,37%
<i>mAP@0.5:0.95</i>	61,98%

Gambar 3.1 merupakan kurva evaluasi performa model klasifikasi berdasarkan metrik confidence, yang digunakan untuk menilai seberapa baik model mendeteksi berbagai jenis kerusakan atau kondisi pada data yang dianalisis. Setiap grafik berisi kurva untuk masing-masing kelas (seperti *Crack*, *Corrosion*, *Squat*, *Peeling*, *Scratch*, dan *Normal*), serta agregat seluruh kelas yang ditunjukkan oleh garis tebal berwarna biru.



Gambar 3. 1 Grafik F1-Confidence (a), Precision-Confidence(b), Precision-Recall(c), Recall(d)

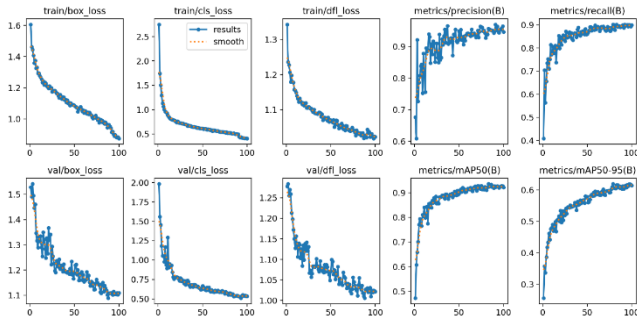
F1-confidence curve, Grafik ini menunjukkan hubungan antara skor F1 dan nilai confidence dari prediksi model. Skor F1 merupakan rata-rata harmonis antara *Precision* dan *Recall*, sehingga menggambarkan keseimbangan keduanya. Dari grafik terlihat bahwa skor F1 agregat terbaik (0.92) tercapai pada threshold confidence sebesar 0.363. Ini menunjukkan bahwa untuk menghasilkan performa terbaik secara keseluruhan, ambang kepercayaan (*Confidence threshold*) ideal adalah sekitar 0.36. Beberapa kelas seperti *Peeling* dan *Crack* menunjukkan performa F1 yang sangat tinggi hampir di seluruh rentang confidence, sedangkan kelas seperti *Scratch* menunjukkan performa yang lebih rendah dan tidak stabil.

Precision-Confidence Curve, Grafik ini menggambarkan bagaimana *Precision* berubah terhadap confidence. *Precision* mengukur seberapa akurat prediksi positif model. Pada grafik ini, *Precision* agregat mencapai nilai maksimum 1.00 pada confidence sebesar 0.927. Ini berarti bahwa jika hanya prediksi dengan confidence di atas 0.927 yang diterima, maka semua prediksi benar (*Precision* 100%), namun kemungkinan besar *Recall* akan menurun. Kelas *Peeling* dan *Crack* memiliki *Precision*

tinggi bahkan pada confidence yang lebih rendah, sedangkan *Scratch* tetap menunjukkan performa yang kurang baik.

Precision-Recall Curve, Grafik ini merupakan visualisasi klasik dari trade-off antara *Precision* dan *Recall* untuk setiap kelas. Nilai *mAP@0.5* (mean Average *Precision* pada threshold IoU 0.5) untuk semua kelas adalah 0.924, yang menunjukkan performa deteksi sangat baik secara keseluruhan. Kelas seperti *Peeling* (0.992), *Corrosion* (0.980), dan *Crack* (0.979) menunjukkan performa sangat tinggi, sedangkan *Scratch* kembali menjadi kelas dengan performa paling rendah (0.709), menandakan model kesulitan mengenali kategori ini secara konsisten.

Recall-Confidence Curve, Grafik ini menunjukkan bagaimana *Recall* (tingkat deteksi positif yang benar) berubah terhadap nilai confidence. *Recall* agregat tertinggi (0.95) tercapai pada threshold confidence 0.0, artinya model mendeteksi hampir semua objek pada confidence rendah, namun tentu saja ini bisa mengorbankan *Precision*. Kelas *Peeling*, *Crack*, dan *Corrosion* menunjukkan *Recall* tinggi yang stabil, sedangkan *Scratch* mengalami penurunan signifikan bahkan di confidence rendah.



Gambar 3. 2 Grafik *Trainig Loss & mAP*

Selama proses pelatihan model, terjadi peningkatan yang konsisten pada berbagai metrik dan fungsi kehilangan. Pada *Train/box_loss*, nilai kehilangan menurun secara stabil dari sekitar 1.6 pada *Epoch* 0 menjadi 0.9 pada *Epoch* 100. Ini menunjukkan bahwa model semakin tepat dalam memprediksi posisi *Bounding box*. *Train/cls_loss* mengalami penurunan drastis dari 2.5 menjadi 0.5, yang merefleksikan peningkatan kemampuan model dalam membedakan kelas objek. Sementara itu, *Train/df_l_Loss* turun dari sekitar 1.3 menjadi 1.05, mengindikasikan bahwa distribusi fokus model menjadi semakin presisi.

Pada data validasi, *val/box_loss* menurun dari 1.5 menjadi 1.1, meskipun terdapat fluktuasi kecil di beberapa *Epoch*, menandakan konsistensi generalisasi model pada prediksi *Bounding box*. *Val/cls_loss* turun tajam dari 2.0 menjadi 0.6, dan *val/df_l_Loss* dari 1.25 menjadi 1.02, memperlihatkan pola penurunan yang sejalan dengan data pelatihan.

Dari sisi metrik evaluasi, *Metrics/Precision(B)* meningkat dari kisaran 0.6–0.7 menjadi hampir 0.95, yang berarti proporsi prediksi benar dari seluruh prediksi

meningkat secara signifikan. *Metrics/Recall(B)* naik dari 0.4 menjadi hampir 0.9, menunjukkan peningkatan dalam kemampuan model untuk menangkap objek yang relevan. *Metrics/mAP50(B)* melonjak dari sekitar 0.5 menjadi mendekati 0.95, yang menunjukkan akurasi tinggi dalam mendeteksi objek dengan ambang IOU 50%. Terakhir, *Metrics/mAP50-95(B)* bertambah dari 0.3 menjadi 0.62, yang merefleksikan perbaikan performa pada berbagai tingkat IOU yang lebih ketat.

B. Hasil Pengujian

Pengujian bertujuan untuk mengevaluasi performa model dalam kondisi nyata dan memastikan bahwa sistem dapat berjalan secara optimal.

Pengujian dilakukan menggunakan dua jenis data. Data lapangan, yang diambil secara langsung di sekitar rel kereta api Gedung D Politeknik Negeri Madiun menggunakan kamera IP (*IP camera*) secara *Real-time*. Data eksternal, berupa gambar rel kereta dari berbagai sumber di internet.









Pengujian ini mencakup berbagai kondisi rel untuk menguji kemampuan deteksi model terhadap jenis cacat yang berbeda. Adapun kondisi yang diuji meliputi:

1. Kondisi rel normal (tanpa cacat)
2. Kondisi Rel dengan *Crack*
3. Kondisi Rel dengan *Corrosion*
4. Kondisi Rel dengan *Squat*
5. Kondisi Rel dengan *Peeling*
6. Kondisi Rel dengan *Scratch*

Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi sebagian besar jenis kecacatan dengan tingkat akurasi yang cukup tinggi. Namun, pada beberapa kasus tertentu seperti retakan halus dengan kontras rendah, model mengalami kesulitan dalam melakukan deteksi secara akurat terlihat pada Tabel 2. 3.

Tabel 2. 2 Hasil Deteksi Menggunakan *IP camera*

Kode	Ground Truth	Hasil Deteksi	Jenis Cacat	Confidence Score	Status Deteksi
0			Crack	0.8	Terdeteksi
1			Corrosion	0.8	Terdeteksi

Kode	Ground Truth	Hasil Deteksi	Jenis Cacat	Confidence Score	Status Deteksi
2			<i>Squat</i>	0.9	Terdeteksi
3			<i>Peeling</i>	0.8	Terdeteksi
4			<i>Scratch</i>	0.8	Terdeteksi
5			Normal	0.9	Terdeteksi

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Penelitian ini berhasil mengembangkan sistem deteksi otomatis cacat permukaan rel kereta api berbasis model YOLOv8n yang terintegrasi dengan input video dari *IP camera*. Proses pengumpulan data mencakup berbagai jenis cacat seperti *Crack*, *Corrosion*, *Squat*, *Peeling*, *Scratch*, serta kondisi normal, yang dianotasi menggunakan *Roboflow* dan dilatih dengan pendekatan *Transfer learning*. Hasil pelatihan menunjukkan performa model yang sangat baik dengan nilai *Precision* sebesar 94,35%, *Recall* 90,27%, *mAP@0.5* sebesar 92,37%, dan *mAP@0.5:0.95* sebesar 61,98%. Evaluasi melalui pengujian langsung menggunakan *IP camera* menunjukkan bahwa sistem mampu mendeteksi berbagai jenis cacat secara *Real-time* dengan tingkat akurasi tinggi, meskipun performa deteksi terhadap kelas *Scratch* masih perlu ditingkatkan. Dengan demikian, sistem yang dikembangkan berpotensi besar untuk digunakan dalam inspeksi rel kereta api secara otomatis dan efisien.

B. Saran

Untuk pengembangan sistem lebih lanjut, beberapa saran yang dapat diberikan antara lain:

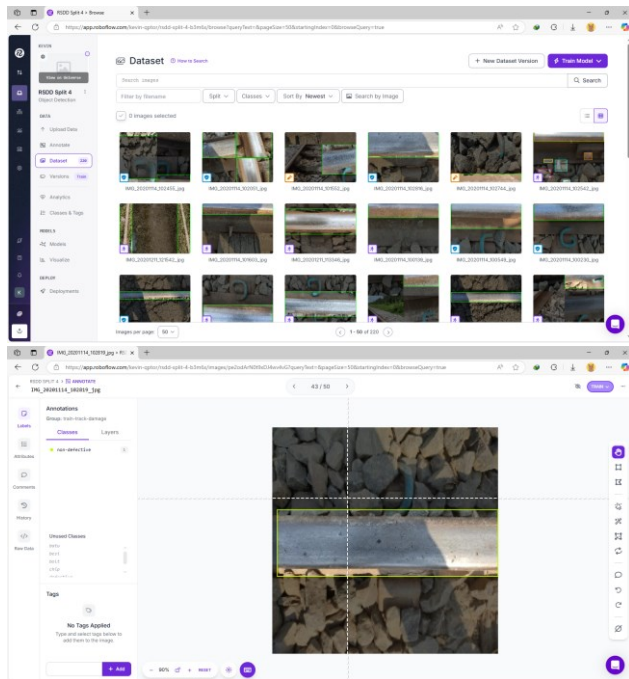
1. Peningkatan Kualitas dan Jumlah Data: Penambahan jumlah data, khususnya untuk kelas minoritas seperti *Scratch*, dapat membantu meningkatkan performa model terhadap kelas tersebut dan mengurangi ketidakseimbangan data.
2. Optimasi Model untuk *Edge deployment*: Mengingat YOLOv8n cukup ringan, pengembangan lebih lanjut dapat diarahkan pada implementasi di perangkat edge seperti *Jetson Nano* atau *Raspberry Pi* agar sistem lebih portabel.
3. Evaluasi Lapangan Lebih Luas: Pengujian model perlu diperluas ke berbagai jenis rel dan kondisi lingkungan nyata untuk memastikan generalisasi model yang lebih kuat dan adaptif.

DAFTAR PUSTAKA

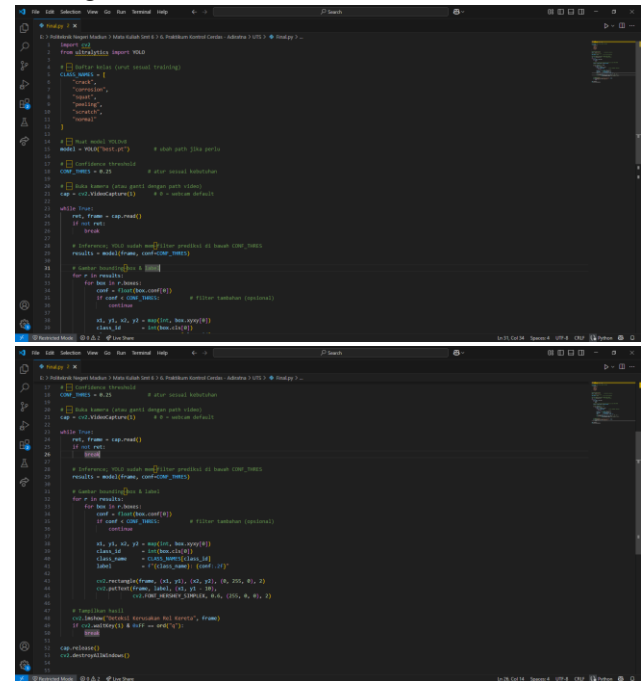
- [1] L. Zhang et al., "Rail defect detection using *Computer Vision* and deep learning: A review," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–19, 2022.
- [2] D. S. Kim et al., "Vision-based railway track inspection using deep convolutional neural networks," *IEEE Access*, vol. 8, pp. 73933–73950, 2020.
- [3] C. Liu et al., "Automatic rail surface defect detection using image processing and deep learning," *IEEE Trans. Ind. Electron.*, vol. 68, no. 4, pp. 3341–3350, 2021.
- [4] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv:1804.02767*, 2018.
- [5] G. Jocher et al., "YOLOv5 by Ultralytics," *GitHub Repository*, 2020. [Online]. Available: <https://github.com/ultralytics/yolov5>
- [6] Ultralytics, "YOLOv8: Next-Generation *Real-time* Object Detection," 2023. [Online]. Available: <https://docs.ultralytics.com>
- [7] A. L. Vargas et al., "*Real-time* object detection in railway using edge-AI and *IP cameras*," *IEEE Sensors Journal*, vol. 22, no. 13, pp. 13009–13017, 2022.
- [8] Y. Wang et al., "Railway surface inspection using lightweight convolutional neural networks," *IEEE Access*, vol. 9, pp. 134152–134165, 2021.

LAMPIRAN

A. Anotasi Gambar Untuk Membuat *Dataset*



B. Program Deteksi Kerusakan Rel



C. Dokumentasi Kegiatan

