

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kesehatan dan Keselamatan Kerja (K3) bertujuan menciptakan lingkungan kerja yang aman dan sehat guna menekan risiko kecelakaan serta penyakit akibat kerja [1]. Area proyek memiliki tingkat risiko tinggi, seperti jatuhnya material, penggunaan alat berat, dan interaksi pekerja dengan kendaraan proyek [2]. Oleh karena itu penerapan K3 dan penggunaan Alat Pelindung Diri (APD), khususnya helm menjadi sangat krusial.

Dalam pengembangan sistem deteksi objek, berbagai algoritma seperti Faster R-CNN, SSD, RetinaNet, EfficientDet, dan keluarga YOLO telah dikembangkan dengan pendekatan *two-stage detector* dan *one-stage detector*. Pendekatan *two-stage detector* umumnya menawarkan akurasi yang tinggi, sedangkan *one-stage detector* lebih efisien dalam proses inferensi karena menggabungkan lokalisasi dan klasifikasi dalam satu tahap sehingga sesuai untuk aplikasi deteksi objek yang membutuhkan keseimbangan antara kecepatan dan akurasi [3, 4, 5]. Meskipun demikian, deteksi objek berukuran kecil atau objek yang berada jauh dari kamera masih menjadi tantangan, terutama pada lingkungan konstruksi yang padat, dinamis, memiliki variasi pencahayaan, serta kondisi saling menutupi antar objek pekerja [4]. Oleh karena itu, diperlukan model deteksi yang efisien, akurat, dan stabil untuk mendukung pengawasan keselamatan kerja.

Dalam konteks tersebut, keluarga YOLO menjadi salah satu pendekatan yang banyak digunakan karena mampu menghasilkan keseimbangan antara akurasi deteksi, kecepatan inferensi, dan efisiensi komputasi [6]. Karakteristik tersebut menjadikan YOLO sesuai untuk pengembangan sistem deteksi helm keselamatan pada video di lingkungan konstruksi.

Meskipun YOLO mampu melakukan deteksi objek secara cepat, pendekatan berbasis *frame* tunggal masih memiliki keterbatasan dalam mempertahankan identitas objek pada kondisi pergerakan cepat, saling menutupi antar objek, dan perubahan posisi yang dinamis [6]. Untuk mengatasinya, dikembangkan berbagai algoritma *Multiple Object Tracking* (MOT). SORT menggunakan deteksi objek dan teknik pelacakan, namun kurang stabil pada kondisi *occlusion* [7]. DeepSORT meningkatkan konsistensi identitas melalui modul *Re-Identification* (ReID), tetapi memerlukan biaya komputasi yang lebih tinggi [8]. Sementara itu, ByteTrack mampu memberikan akurasi dan konsistensi pelacakan yang lebih baik dibandingkan SORT dan DeepSORT, serta lebih efisien secara komputasi dibandingkan

BoTSORT [7, 9]. Dengan kombinasi akurasi, stabilitas identitas, dan efisiensi komputasi tersebut, ByteTrack dinilai sesuai untuk sistem deteksi helm keselamatan berbasis YOLO karena mampu mempertahankan konsistensi identitas objek dengan efisiensi komputasi yang baik pada kondisi lingkungan konstruksi yang dinamis [9].

Seiring perkembangannya, setiap versi YOLO mengalami peningkatan dalam kecepatan, efisiensi, dan kemampuan mendeteksi objek berukuran kecil seperti helm keselamatan. YOLOv1 hingga YOLOv3 menjadi fondasi awal pengembangan algoritma deteksi objek satu tahap, namun masih memiliki keterbatasan dalam mendeteksi objek berukuran kecil. YOLOv4 dan YOLOv5 meningkatkan stabilitas dan akurasi deteksi, sedangkan YOLOv6 dan YOLOv7 menghadirkan optimasi yang mampu meningkatkan performa pada objek kecil hingga menengah, meskipun membutuhkan sumber daya komputasi yang lebih besar. Di antara berbagai versi tersebut, YOLOv8 menawarkan fleksibilitas melalui berbagai ukuran model yang memungkinkan penyesuaian antara kecepatan inferensi, penggunaan memori, dan akurasi deteksi. Selain itu, penerapan pendekatan *anchor-free* meningkatkan akurasi lokalisasi, khususnya pada objek berukuran kecil hingga menengah, tanpa mengorbankan performa inferensi secara signifikan. Sementara itu, YOLOv9, YOLOv10, dan YOLOv11 menunjukkan peningkatan performa secara umum, tetapi belum memberikan keunggulan yang signifikan dalam mendeteksi objek kecil seperti helm keselamatan dibandingkan YOLOv8 [10, 11, 12].

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah dalam penelitian ini adalah bagaimana mendeteksi penggunaan helm keselamatan pada video dengan mengidentifikasi objek *Person*, *Hardhat*, dan *No-hardhat* serta mempertahankan konsistensi identitas pekerja pada kondisi pergerakan dinamis dan *occlusion*.

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah mengembangkan sistem deteksi otomatis penggunaan helm keselamatan berbasis *deep learning* dengan menggunakan algoritma YOLOv8s untuk mendeteksi *person*, *hardhat*, dan *no hardhat*, serta mengimplementasikan algoritma ByteTrack untuk melakukan pelacakan pekerja pada citra video sehingga setiap individu yang terdeteksi dapat diikuti pergerakannya secara konsisten. Hasil penelitian ini berupa aplikasi desktop yang mampu melakukan deteksi dan pelacakan pekerja serta penggunaan helm keselamatan, mencatat pekerja yang terdeteksi tidak

menggunakan helm berdasarkan waktu deteksi, serta menyimpan *screenshot* secara otomatis sebagai dokumentasi pelanggaran.

1.4 Manfaat

Penelitian ini diharapkan dapat memberikan kontribusi ilmiah dalam pengembangan kajian *deep learning* melalui penerapan algoritma YOLOv8 dan ByteTrack untuk deteksi serta pelacakan objek. Selain itu, penelitian ini dapat menjadi referensi, bahan perbandingan, dan dasar pengembangan metode lanjutan dalam sistem deteksi dan pelacakan objek, khususnya untuk objek berukuran kecil seperti helm keselamatan.

1.5 Ruang Lingkup

Ruang lingkup penelitian ini dibatasi pada beberapa aspek berikut:

1. *Dataset* yang digunakan berasal dari *platform Kaggle* dengan jumlah data sebanyak 2.605 citra untuk data latih (*training*), 114 citra untuk data *validasi* (*validation*), dan 82 citra untuk data uji (*testing*).
2. Seluruh citra yang digunakan untuk *training*, *validation* dan *testing* memiliki ukuran resolusi 640 x 640 piksel dengan format *file* PNG.
3. Objek yang dideteksi dalam penelitian ini berfokus pada deteksi penggunaan helm keselamatan (*Hardhat* dan *No-hardhat*) serta keberadaan *Person*.
4. Metode deteksi objek yang digunakan adalah algoritma YOLOv8 dengan varian *small*.
5. Proses pelacakan objek dilakukan menggunakan algoritma ByteTrack untuk memberikan identitas unik pada setiap pekerja yang terdeteksi.
6. Lingkungan pengembangan sistem dilakukan menggunakan platform Google Colab dengan dukungan GPU NVIDIA T4.
7. Bahasa pemrograman yang digunakan dalam pengembangan sistem adalah Python versi 3.
8. Evaluasi kinerja model dilakukan menggunakan metrik *Precision*, *Recall*, *F1-score*, *mAP50*, dan *mAP50-95*, *MOTA*, *MOTP*, *ID Switches*, *MT*, *ML*.
9. Proses pelatihan model dilakukan dengan menguji tiga konfigurasi hyperparameter yang diadaptasi dari penelitian Wang et al., Han et al., serta Song dan Li. Konfigurasi Wang et al. menggunakan ukuran citra 640×640 piksel, *batch size* 8, *epoch* 200, *optimizer* SGD (*Stochastic Gradient Descent*), *initial learning rate* 0,01, *weight decay* 0,0005, *momentum* 0,937, *warmup epochs* 3, serta teknik augmentasi data berupa *Mosaic* 1.0, *Close Mosaic* 10, *MixUp* 0.1, *Hue Adjustment* (*hsv_h*) 0.015, *Saturation*

Adjustment (*hsv_s*) 0.7, Brightness Adjustment (*hsv_v*) 0.4, Flip Left-Right (*fliplr*) 0.5, dan Scale 0.5 [13]. Konfigurasi Han et al. menggunakan ukuran citra 640×640 piksel, *batch size* 48, *epoch* 100, *optimizer* SGD, *initial learning rate* 0,03, *final learning rate* 0,01, dan *weight decay* 0,0005 tanpa konfigurasi augmentasi yang dijelaskan secara rinci [14]. Sementara itu, konfigurasi Song dan Li menggunakan ukuran citra 640×640 piksel, *batch size* 16, *epoch* 300, *optimizer* SGD, *initial learning rate* 0,01, dan *final learning rate* 0,01 tanpa menerapkan teknik augmentasi data [15]. Ketiga konfigurasi tersebut diuji menggunakan dataset penelitian, kemudian konfigurasi Song dan Li dipilih sebagai konfigurasi pelatihan akhir berdasarkan hasil evaluasi yang memberikan performa terbaik.

