JURNAL FASILKOM
Volume 16 No. 1 | April 2026: 10-19

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Deteksi Bahasa Isyarat Menggunakan Arsitektur YOLOv8 Berbasis *Website*

Muhammad Faruqi Rabbani[1], Danang Arbian Sulistyo[2]
[1,2]Teknik Informatika, Fakultas Teknologi dan Desain, Institut Teknologi dan Bisnis Asia Malang
[1]faruqirabbani3@gmail.com, [2]danangarbian@gmail.com *

Abstract

Communication difficulties between the general public and people with hearing impairments due to limited access to real-time detection tools are the primary urgency of this research. This research aims to develop a cross-platform and easily accessible website-based sign language detection system, while implementing the YOLOv8 variant to remain accurate on devices with limited computing resources. The method used is Research and Development (R&D) with the AI Project Cycle framework, which includes data collection, preprocessing, modeling using the YOLOv8n variant, and implementation. The data used is sourced from the Roboflow platform, consisting of hand gesture images divided into 70% training data, 20% validation, and 10% testing. The results show that the YOLOv8n model provides high performance with a precision of 0.932, recall of 0.997, and mAP50 value of 0.995. Additionally, the model achieves an efficient inference speed averaging 2.1 ms. In conclusion, the implementation of YOLOv8 on a website-based successfully creates an accurate and responsive sign language detection system, making it suitable for assisting communication in real-world scenarios.

Keywords: artificial intelligence, computer vision, sign language detection, website, YOLOv8

Abstrak

Kesulitan komunikasi antara masyarakat umum dan penyandang tunarungu akibat keterbatasan akses alat bantu deteksi *real-time* menjadi urgensi utama penelitian ini. Penelitian ini bertujuan untuk mengembangkan sistem deteksi bahasa isyarat berbasis *website* yang lintas *platform* dan mudah diakses, serta mengimplementasikan varian YOLOv8 agar tetap akurat pada perangkat dengan sumber daya komputasi terbatas. Metode yang digunakan adalah *Research and Development* (R&D) dengan kerangka *AI Project Cycle* yang meliputi tahap pengumpulan data, prapemrosesan, pemodelan menggunakan varian YOLOv8n, hingga implementasi. Data yang digunakan berasal dari *platform Roboflow* berupa citra gestur tangan yang dibagi menjadi 70% data pelatihan, 20% validasi, dan 10% pengujian. Hasil penelitian menunjukkan bahwa model YOLOv8n memberikan performa yang tinggi dengan nilai *precision* sebesar 0,932, *recall* 0,997, serta nilai mAP50 mencapai 0,995. Selain itu, model mencapai kecepatan inferensi yang sangat efisien rata-rata 2,1 ms. Kesimpulannya, implementasi YOLOv8 pada *platform website* berhasil menciptakan sistem deteksi bahasa isyarat yang akurat dan responsif, sehingga layak digunakan untuk membantu komunikasi dalam skenario nyata.

Kata kunci: visi komputer, deteksi bahasa isyarat, kecerdasan buatan, situs web, YOLOv8

1. Pendahuluan

Bahasa isyarat merupakan sarana komunikasi utama bagi penyandang tunarungu dan tunawicara dalam berinteraksi serta menyampaikan informasi di berbagai konteks kehidupan. Namun, upaya mewujudkan komunikasi inklusif bagi penyandang tunarungu masih terhambat oleh keterbatasan teknologi deteksi bahasa isyarat yang seringkali membutuhkan perangkat keras spesifik dan biaya tinggi untuk digunakan oleh masyarakat umum. Kondisi ini mendorong perlunya pemanfaatan teknologi berbasis kecerdasan buatan untuk membantu mengenali dan menerjemahkan gestur bahasa isyarat secara otomatis. Perkembangan *computer vision* dan *deep learning* memungkinkan pengolahan citra dan video secara *real-time* dengan tingkat akurasi yang tinggi, sehingga sangat relevan untuk diterapkan pada sistem deteksi bahasa isyarat berbasis visual [1]. Pendekatan pemodelan sistem semacam ini juga telah terbukti efektif dalam mengatasi permasalahan manajemen operasional melalui integrasi perangkat keras dan pengolahan citra cerdas [2].

Berbagai penelitian sebelumnya menunjukkan bahwa algoritma deteksi objek berbasis *Convolutional Neural Network* (CNN), khususnya *You Only Look Once* (YOLO), memiliki performa yang baik dalam mengenali objek secara cepat dan akurat. Studi terbaru menegaskan bahwa arsitektur berbasis CNN mampu menangkap fitur esensial dari citra dengan akurasi yang sangat tinggi [3]. YOLOv8 sebagai versi terbaru menawarkan peningkatan signifikan melalui arsitektur yang lebih ringan, pendekatan *anchor-free*, serta efisiensi komputasi yang lebih baik dibandingkan versi sebelumnya. Sejumlah studi membuktikan bahwa YOLOv8 mampu menghasilkan nilai *precision*, *recall*, dan *mean Average Precision* (mAP) yang tinggi pada berbagai skenario deteksi objek, termasuk pada sistem berbasis citra dan video [4], [5]. Penelitian terkait deteksi bahasa isyarat Indonesia juga menunjukkan bahwa pendekatan berbasis YOLO efektif dalam

P-ISSN : 2089-3353
E-ISSN : 2808-9162

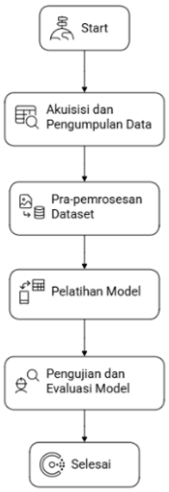
mengenali gestur tangan dengan tingkat akurasi yang kompetitif [1].

Meskipun demikian, sebagian besar implementasi YOLOv8 masih difokuskan pada aplikasi yang dijalankan pada perangkat dengan sumber daya komputasi relatif besar, seperti laptop atau perangkat bergerak tertentu, sehingga pemanfaatannya menjadi terbatas pada *platform* dan lingkungan tertentu. Padahal, kebutuhan akan sistem deteksi bahasa isyarat yang bersifat lintas *platform* dan mudah diakses semakin meningkat. *Platform* berbasis *website* menawarkan keunggulan dalam hal aksesibilitas karena dapat digunakan pada berbagai perangkat tanpa memerlukan instalasi khusus. Namun, secara teknis, aplikasi berbasis web seringkali memiliki performa inferensi yang lebih rendah dibandingkan aplikasi natif karena adanya lapisan abstraksi tambahan pada peramban dan keterbatasan akses langsung ke akselerasi perangkat keras [1]. Tantangan utama dalam penerapan YOLOv8 pada *platform website* ini mencakup keterbatasan sumber daya komputasi dan latensi, yang berpotensi menurunkan responsivitas sistem jika tidak dikelola melalui pemilihan varian model yang tepat.

Oleh karena itu, penelitian ini dilakukan untuk menjawab permasalahan mengenai bagaimana mengimplementasikan varian YOLOv8 agar kompatibel dengan komputasi terbatas pada *platform website* tanpa mengorbankan akurasi deteksi. Penelitian ini bertujuan mengembangkan sistem deteksi bahasa isyarat berbasis *website* yang dapat diakses dari berbagai perangkat, sekaligus menganalisis efisiensi penggunaan model YOLOv8 dalam mencapai keseimbangan antara kecepatan inferensi dan akurasi. Evaluasi performa model dilakukan untuk memastikan sistem memenuhi parameter keberhasilan berupa kecepatan inferensi di bawah 5 ms per citra dan nilai mAP50 di atas 0,90, sehingga sistem tetap responsif dan akurat saat digunakan dalam skenario nyata.

2. Metode Penelitian

Penelitian ini menggunakan metode *Research and Development* (R&D) untuk mengembangkan sistem deteksi bahasa isyarat berbasis *website* dengan arsitektur YOLOv8. Proses pengembangan mengadopsi kerangka AI *Project Cycle* yang meliputi identifikasi masalah, pengumpulan data, prapemrosesan data, pemodelan, evaluasi, hingga implementasi sistem guna memastikan hasil yang optimal [6].



Gambar 1. *Flowchart* Tahapan Penelitian

Secara sistematis, tahapan penelitian ini digambarkan melalui flowchart pada Gambar 1. Alur dimulai dari tahap Akuisisi dan Pengumpulan Data, di mana dilakukan pengumpulan dataset bahasa isyarat. Selanjutnya adalah pra-pemrosesan dataset yang mencakup anotasi data dan pembagian data (*data splitting*) untuk kebutuhan validasi.

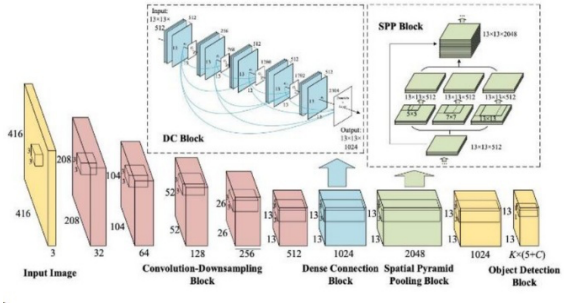
Tahap berikutnya adalah pelatihan model, di mana arsitektur YOLOv8 dilatih menggunakan *dataset* yang telah disiapkan. Setelah model terbentuk, dilakukan pengujian dan evaluasi model untuk mengukur tingkat akurasi dan kinerja deteksi sebelum akhirnya diimplementasikan ke dalam *platform* berbasis *website* [1]. Pendekatan ini memastikan setiap tahapan dapat dianalisis dan dievaluasi secara terukur.

Dataset penelitian yang berupa citra gestur bahasa isyarat Indonesia dikumpulkan dari berbagai sumber dengan memperhatikan variasi pencahayaan serta latar belakang yang berbeda. Variasi kondisi lingkungan seperti pencahayaan yang kurang ideal dan latar belakang yang memiliki banyak *noise* merupakan tantangan krusial yang harus diantisipasi agar model tidak hanya bekerja baik pada kondisi laboratorium tetapi juga pada kondisi liar (*wild condition*) [7]. Proses pengambilan data menggunakan kamera digital untuk merekam detail gestur yang digunakan dalam proses analisis [8]. Setelah data terkumpul, tahap prapemrosesan segera dilakukan dengan menyesuaikan ukuran gambar menjadi 640×640 piksel, melakukan anotasi *bounding box*, serta memberikan pelabelan pada setiap kelas gestur. Tahapan *preprocessing* seperti normalisasi dan *resizing* ini penting dilakukan agar model dapat belajar lebih efisien dan menghindari masalah seperti *vanishing gradient*. Penggunaan *platform Roboflow* dalam proses anotasi ini sangat krusial untuk menghasilkan format data yang kompatibel dengan arsitektur YOLOv8 [9].

Untuk menjaga objektivitas dalam pembelajaran dan evaluasi model, *dataset* dengan total keseluruhan 10.261 citra tersebut dibagi menjadi tiga bagian utama.

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Pembagian data latih, validasi, dan uji merupakan standar dalam *deep learning* untuk menghindari *overfitting* serta mendapatkan evaluasi yang realistis. Pada penelitian ini, rasio pembagian data yang digunakan adalah 70% untuk data pelatihan (7.201 citra), 20% untuk data validasi (2.052 citra), dan 10% untuk data pengujian (1.008 citra) [10]. Proporsi pembagian data 70:20:10 ini terbukti efektif dalam mencerminkan kondisi dunia nyata dan memberikan keseimbangan antara proses pembelajaran dan validasi performa model pada berbagai kondisi kualitas gambar [11]. Guna memperkaya *dataset*, teknik augmentasi diterapkan untuk menghasilkan 2 luaran per data latih (*outputs per training example*). Parameter augmentasi yang digunakan dalam penelitian ini meliputi beberapa teknik transformasi citra untuk meningkatkan variasi data. Teknik tersebut terdiri dari *Rotation* (rotasi), *Shear* (pergeseran sudut), penyesuaian *Brightness* (kecerahan), *Flip Horizontal* (pencerminan horizontal), dan *Blur* (pengaburan). Penggunaan parameter ini bertujuan agar model memiliki ketahanan terhadap variasi posisi, pencahayaan, dan kualitas ketajaman gambar pada *dataset*. Teknik ini diterapkan secara sistematis untuk meningkatkan keragaman data [8]. Seluruh rangkaian persiapan ini dirancang sedemikian rupa agar model memiliki performa yang optimal saat mengenali gestur dalam lingkungan yang dinamis.



Gambar 2. Arsitektur YOLOv8

Berdasarkan gambar arsitektur yang dilampirkan, proses deteksi diawali dengan transformasi citra input melalui tahapan *Convolution-Downsampling* yang secara bertahap mereduksi dimensi spasial. Komponen kunci yang memperkuat model ini adalah penggunaan blok C2f (*Cross-stage Partial Bottleneck with 2 Convolutions*) yang memungkinkan integrasi fitur secara efisien melalui pemisahan aliran gradien, serta blok SPPF (*Spatial Pyramid Pooling - Fast*). Sebagaimana terlihat pada detail blok SPPF di gambar, model menerapkan berbagai ukuran *pooling* secara paralel namun tetap efisien untuk mengekstraksi informasi kontekstual yang lebih kaya, sebelum akhirnya diproses oleh *Object Detection Block (Head)* untuk menghasilkan prediksi akhir.

Arsitektur YOLOv8 dipilih sebagai model deteksi objek utama karena keunggulannya dalam menyeimbangkan kecepatan pemrosesan dan akurasi tinggi melalui tiga komponen terintegrasi, yaitu *backbone*, *neck*, dan *head* [12]. Pada tahap awal, komponen *backbone* yang berbasis *CSPDarknet* bertugas mengekstraksi fitur-fitur esensial dari citra input menggunakan lapisan konvolusi dan modul C2f yang dioptimalkan untuk aliran gradien yang lebih baik. Fitur-fitur tersebut kemudian diteruskan ke bagian *neck* yang mengimplementasikan struktur *Feature Pyramid Network* (FPN) dan *Path Aggregation Network* (PAN) guna menggabungkan informasi fitur dari berbagai skala, sehingga model mampu mengenali objek kecil maupun besar dengan presisi yang konsisten [1].

Selanjutnya, bagian *head* pada YOLOv8 menerapkan pendekatan *anchor-free detection* yang secara signifikan menyederhanakan proses prediksi *bounding box* dan klasifikasi dibandingkan generasi sebelumnya. Dalam implementasi ini, varian YOLOv8 digunakan karena sifatnya yang sangat ringan dan efisien dengan total parameter hanya sekitar 3,2 juta, namun tetap kompetitif dalam hasil deteksi. Untuk mengoptimalkan performa model selama fase pelatihan, digunakan mekanisme optimasi otomatis (*auto-optimizer*) yang secara cerdas menentukan *optimizer* terbaik antara AdamW atau SGD berdasarkan arsitektur model. Dalam penelitian ini, penggunaan AdamW lebih diutamakan karena kemampuannya dalam melakukan peluruhan bobot (*weight decay*) secara terpisah dari pembaruan gradien, yang terbukti lebih efektif dalam mempercepat konvergensi dan meningkatkan generalisasi model pada arsitektur *deep learning* [12]. Proses pelatihan dijalankan dengan *learning rate* 0,01 dan *batch size* 32, serta didukung oleh mekanisme *early stopping* dengan nilai *patience* sebesar 10. Parameter *patience* ini memastikan pelatihan dihentikan secara otomatis jika tidak terdapat peningkatan performa pada data validasi selama 10 epoch berturut-turut guna mencegah terjadinya *overfitting* [10]. Eksplorasi *hyperparameter* seperti epoch, *batch size*, dan *learning rate* sangat penting dilakukan untuk mencapai konvergensi model yang optimal dan akurasi deteksi yang maksimal, sebagaimana diterapkan dalam pengembangan sistem deteksi berbasis YOLOv8 lainnya [13].

Perancangan sistem deteksi bahasa isyarat ini diwujudkan dalam bentuk aplikasi berbasis website guna memastikan aksesibilitas yang luas bagi pengguna. Arsitektur *backend* dibangun menggunakan bahasa pemrograman Python dengan *framework Flask* yang dijalankan melalui lingkungan *Google Colab*. Untuk menunjang kebutuhan komputasi, sistem dijalankan menggunakan GPU NVIDIA Tesla T4 dengan memori video (VRAM) sebesar 16 GB. Selain itu, sistem dikonfigurasi agar memiliki fleksibilitas tinggi dengan menyediakan mekanisme *fallback* otomatis; sistem akan mendeteksi ketersediaan akselerasi perangkat keras dan secara otomatis beralih menggunakan CPU Intel(R) Xeon(R) @ 2.20GHz jika GPU tidak terdeteksi. Spesifikasi ini menjadi batasan daya komputasi yang mendasari tercapainya kecepatan inferensi model sebesar 2,1 ms per citra selama proses deteksi *real-time*. Sementara itu, sisi *frontend*

P-ISSN : 2089-3353
E-ISSN : 2808-9162

dikembangkan menggunakan kombinasi HTML, Talwind CSS, dan *JavaScript* guna menciptakan antarmuka yang responsif, intuitif, dan mudah digunakan [14]. Melalui antarmuka tersebut, pengguna memiliki fleksibilitas untuk mengunggah gambar statis maupun menggunakan akses kamera secara langsung untuk mendeteksi gestur bahasa isyarat, di mana hasil deteksinya akan ditampilkan secara visual dalam bentuk *bounding box*, label kelas, serta nilai kepercayaan (*confidence score*) pada halaman *website* [15].

Untuk mendukung mekanisme kerja tersebut, *backend* sistem menyediakan API (*Application Programming Interface*) berbasis Flask yang bertugas menerima input data visual, memprosesnya menggunakan algoritma YOLOv8, dan mengembalikan hasil deteksi dalam format JSON. Karena dijalankan di lingkungan *Google Colab*, pemanfaatan pustaka *PyTorch* menjadi krusial untuk mendukung proses inferensi yang adaptif, di mana sistem secara otomatis mendeteksi dan memanfaatkan akselerasi GPU apabila tersedia atau beralih ke CPU untuk menjaga stabilitas operasional [6]. Selain itu, integrasi layanan *tunneling* Ngrok digunakan untuk memastikan aksesibilitas API melalui domain statis. Pada tahap akhir, sistem menjalani rangkaian pengujian fungsional dan performa, sementara kualitas model dievaluasi secara mendalam menggunakan metrik *accuracy, precision, recall, dan F1-score* yang diekstraksi dari *confusion matrix* untuk menjamin akurasi deteksi yang optimal [14].

Pentingnya evaluasi menyeluruh terhadap kinerja model juga ditekankan dalam studi komparasi metode deteksi objek, di mana analisis terhadap kondisi pencahayaan yang beragam dan gangguan visual (*noise*) menjadi indikator utama keandalan sistem. Seperti halnya pada perbandingan performa antara arsitektur deteksi objek lainnya, kemampuan model untuk mempertahankan akurasi tinggi di bawah kondisi pencahayaan yang kurang ideal atau gangguan lingkungan adalah faktor penentu keberhasilan implementasi di dunia nyata [11]. Selain itu, perbandingan metode deteksi juga menunjukkan bahwa metode berbasis CNN dan variannya cenderung lebih unggul dalam menangani tantangan variasi pose dan oklusi dibandingkan metode deteksi konvensional [7]. Oleh karena itu, pengujian sistem ini dirancang untuk mencakup berbagai skenario guna memvalidasi ketangguhan arsitektur yang diusulkan.

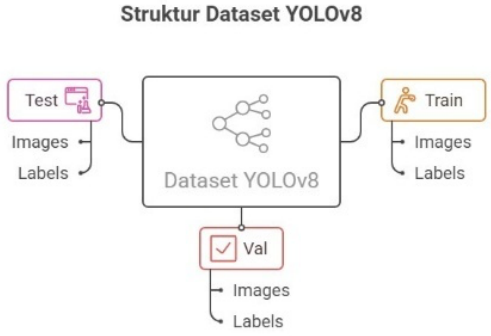
3. Hasil dan Pembahasan

Pada bagian hasil dan pembahasan ini, dijelaskan secara menyeluruh bagaimana proses penelitian dilakukan mulai dari pengolahan *dataset*, pemanfaatan arsitektur YOLOv8, hingga analisis performa model setelah melalui proses pelatihan. Seluruh tahapan yang disampaikan bertujuan untuk menunjukkan alur kerja model dalam mendeteksi bahasa isyarat, mulai dari penyiapan data yang tepat, konfigurasi model, proses training, hingga interpretasi hasil evaluasi. Dengan penyampaian ini, diharapkan pembaca dapat memahami keterkaitan setiap proses serta bagaimana tahapan-tahapan tersebut berpengaruh terhadap kualitas model yang dihasilkan.

3.1. *Dataset*

Penelitian ini menggunakan *dataset* yang bersumber dari dua proyek berbeda pada *platform Roboflow* guna memperoleh variasi data kata dan abjad yang komprehensif. *Dataset* pertama diperoleh melalui proyek berjudul "Deteksi Bahasa Isyarat Indonesia" yang dapat diakses melalui tautan https://universe.roboflow.com/tugasakhirvidia/deteksi-bahasa-isyarat-indonesia. Berdasarkan hasil peninjauan visual secara mendalam, ditemukan bahwa *dataset* tersebut memiliki ketidaksesuaian pada representasi abjadnya, di mana konfigurasi jari yang digunakan cenderung mengikuti standar *American Sign Language* (ASL) dibandingkan Bahasa Isyarat Indonesia (BISINDO). Mengingat adanya perbedaan fundamental pada representasi visual tersebut, penelitian ini hanya mengekstraksi sampel data berupa kategori kata dari sumber tersebut, yang meliputi label "Ayah", "Halo", "Kakak", "Minum", "Rumah", "Sama-sama", "Sehat", "Teman", "Terima Kasih", dan "Tidur".

Untuk melengkapi kebutuhan data abjad yang sesuai dengan standar lokal, dilakukan integrasi *dataset* kedua yang bersumber dari proyek "BISIN-Anotasi" dengan tautan https://universe.roboflow.com/bisindo-4zqi6/bisin-anotasi. *Dataset* ini digunakan secara khusus untuk merepresentasikan abjad A–Z dalam format BISINDO. Dalam tahap pengolahan awal, diterapkan teknik *preprocessing* melalui augmentasi data guna meningkatkan *volume* citra sekaligus memastikan keseimbangan distribusi kelas pada model. Metode augmentasi yang diimplementasikan dalam penelitian ini meliputi *rotation*, *shear*, *brightness adjustment*, *horizontal flip*, dan *blur*. Melalui serangkaian prosedur tersebut, variasi dataset ditingkatkan dan jumlah data pada setiap kelas berhasil diseragamkan. Hal ini dilakukan guna menjaga stabilitas performa model saat proses pelatihan serta menghindari bias pada kelas tertentu (*overfitting*) akibat kurangnya keberagaman fitur visual.

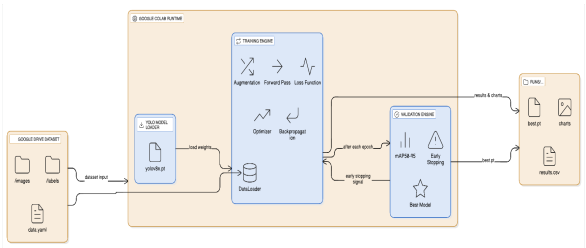


Gambar 3. Struktur *Dataset* YOLOv8

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Keseluruhan *dataset* yang telah terkumpul kemudian dikonsolidasikan ke dalam format arsitektur YOLOv8 dengan total 36 kelas objek. Data tersebut dibagi menggunakan teknik *random splitting* dengan rasio 70% sebagai data *training* (pelatihan), 20% sebagai data *validation* (validasi), dan 10% sebagai data *test* (pengujian). Secara spesifik, setiap kelas memiliki distribusi data yang konsisten, yaitu sebanyak 200 sampel untuk tahap *train*, 57 sampel untuk valid, dan 28 sampel untuk *test*. Setiap sampel citra telah dilengkapi dengan anotasi dalam format .txt yang berisi label kelas dan koordinat *bounding box* yang telah dinormalisasi. Struktur data ini kemudian didefinisikan ke dalam berkas konfigurasi data.yaml untuk memfasilitasi proses ekstraksi fitur dan deteksi objek pada tahap pelatihan model YOLOv8.

3.2. Arsitektur YOLOv8



Gambar 4. Arsitektur YOLOv8

Arsitektur YOLOv8 terdiri dari tiga komponen utama, yaitu *backbone, neck,* dan *detection head*. Pada penelitian ini digunakan varian YOLOv8n yang memiliki ukuran ringan namun tetap mampu menghasilkan performa deteksi yang baik. Bagian *backbone* berfungsi mengekstraksi fitur dari citra masukan melalui kombinasi *convolutional layers* serta modul C2f yang dirancang untuk meningkatkan efisiensi dan kapasitas representasi fitur. Fitur-fitur yang dihasilkan kemudian diproses oleh *neck*, yang mengimplementasikan struktur FPN/PAN untuk menggabungkan informasi dari berbagai tingkat kedalaman sehingga model mampu mendeteksi objek dengan ukuran beragam. Pada tahap akhir, *detection head* YOLOv8 menggunakan pendekatan *anchor-free* yang memungkinkan prediksi *bounding box* dan kelas secara lebih sederhana, cepat, dan adaptif dibandingkan generasi sebelumnya. Keseluruhan arsitektur ini menghasilkan model yang efisien, mudah dilatih, dan memiliki kemampuan generalisasi yang baik, sehingga cocok digunakan pada tugas deteksi objek dengan sumber daya komputasi terbatas.

Tabel 1. Parameter Arsitektur YOLOv8

Parameter	Nilai
Model	YOLOv8
Pre-trained Weights	yolov8n.pt
Input Size	640 × 640
Batch Size	32
Epochs	50
Early Stopping	10 *epochs*
Optimizer	*Default* (SGD/AdamW)
Learning Rate	*Default* (0.01)
Backbone	CSPDarknet + C2f
Neck	PAN
Head	*Decoupled Head*
Parameters	~3.2M
Loss Function	CIoU + DFL
Device	GPU (CUDA:0)

Berdasarkan Tabel 1 menggambarkan struktur arsitektur YOLOv8 yang terdiri dari tiga komponen utama, yaitu *backbone, neck,* dan *head*. Pada bagian *backbone*, YOLOv8 menggunakan kombinasi Conv dan blok C2f untuk meningkatkan ekstraksi fitur secara efisien, serta modul SPPF yang berfungsi menangkap konteks global melalui mekanisme *pooling* multi-skala. Bagian *neck* mengintegrasikan FPN dan PAN untuk menggabungkan fitur dari berbagai level kedalaman, sehingga model mampu mendeteksi objek dengan ukuran kecil hingga besar secara optimal. Pada bagian head, YOLOv8 mengadopsi desain *decoupled head* dengan pemisahan tugas klasifikasi dan regresi, serta menerapkan pendekatan *anchor-free* untuk menyederhanakan proses prediksi dan meningkatkan fleksibilitas deteksi. Proses akhir prediksi disempurnakan dengan *Non-Maximum Suppression* (NMS) untuk menghilangkan prediksi yang tumpang tindih. Struktur ini menjadikan YOLOv8 lebih ringan, akurat, dan efisien dibandingkan generasi sebelumnya.

3.3. *Pseudocode* dan Prosedur Sistem

Prosedur penelitian ini dibagi menjadi dua tahapan utama: proses pelatihan model secara matematis dan proses inferensi sistem melalui integrasi backend.

a. Alur Pelatihan Model (*Training Phase*)

Penelitian ini menggunakan arsitektur YOLOv8 nano (YOLOv8n) sebagai model deteksi objek untuk mengenali gestur bahasa isyarat. Proses *training* diawali dengan memuat model *pre-trained* YOLOv8n, kemudian dilakukan *fine-tuning* pada *dataset custom*. Model diinisialisasi dengan parameter awal $\theta 0$ dan dikonfigurasi dengan *hyperparameter* sebagai berikut: jumlah *epoch* maksimal $E_{max} = 50$, ukuran *batch* $B = 32$, dan ukuran gambar input $I = 640 \times 640$ piksel. Seluruh proses *training* dilakukan pada GPU untuk mempercepat komputasi.

Proses *training* dilakukan secara iteratif untuk setiap *epoch* $e = 1$ hingga E_{max}. Pada setiap *epoch*, dilakukan dua tahap utama yaitu *training step* dan *validation step*. Pada *training step*, untuk setiap *batch* b_i dalam *dataset training*, model menghitung total loss yang merupakan kombinasi dari tiga komponen *loss function*: (1)

$$L_{train}(e) = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} [L_{box}(y_i, \hat{y}_i) + L_{cls}(c_i, \hat{c}_i) + L_{obj}(o_i, \hat{o}_i)] \quad (1)$$

dimana L_{box} merupakan *bounding box regression loss* menggunakan metode *Complete Intersection over Union* (CIoU), L_{cls} adalah *classification loss* untuk memprediksi kelas objek, dan L_{obj} adalah *objectness loss* yang mengukur kepercayaan keberadaan objek. Notasi y_i dan $\hat{y}_i$ masing-masing merepresentasikan *ground truth* dan *predicted bounding box*, sedangkan c_i

P-ISSN : 2089-3353
E-ISSN : 2808-9162

dan $\hat{c}_i$ merepresentasikan *ground truth* dan *predicted class*. Parameter model kemudian diperbarui menggunakan *gradient descent* dengan persamaan:(2)

$$\theta_{\mathrm{e}} = \theta_{\mathrm{e}-1} - \eta \nabla\theta \, L_{train}(\mathrm{e}) \quad (2)$$

dengan η adalah *learning rate* yang mengontrol besarnya *update* parameter pada setiap iterasi.

Setelah *training step* selesai, dilakukan *validation step* untuk mengevaluasi performa model pada *validation set*. *Loss* validasi dihitung dengan formula yang serupa dengan *loss training*:(3)

$$L_{val(e)} = \frac{1}{N_{val}} \sum_{j=1}^{N_{val}} [L_{box}(y_j, \hat{y}_j) + L_{cls}(c_j, \hat{c}_j) + L_{obj}(o_j, \hat{o}_j)] \quad (3)$$

selain loss, metrik evaluasi utama yang digunakan adalah *mean Average Precision* (mAP). Metrik mAP_{50} dihitung pada *threshold* IoU 0.5:(4)

$$mAP_{50(e)} = \frac{1}{C} \sum_{k=1}^{C} AP_k^{IoU=0.5} \quad (4)$$

sedangkan $mAP_{\{50:95\}}$ dihitung sebagai rata-rata mAP pada rentang *threshold* IoU dari 0.5 hingga 0.95 dengan interval 0.05:(5)

$$mAP_{50}{:}95\ (e) = \frac{1}{C} \sum_{k=1}^{C} \frac{1}{10} \sum_{t=0.5}^{0.95} AP_k^{IoU=t} \quad (5)$$

dimana C adalah jumlah total kelas dalam dataset dan AP_k merupakan *Average Precision* untuk kelas ke-k. Metrik $mAP_{\{50:95\}}$ digunakan sebagai kriteria utama untuk model selection.

Untuk mengoptimalkan performa dan efisiensi *training*, diterapkan strategi *early stopping* dengan *patience* $\mathrm{p} = 10$ epoch. Sistem menyimpan model terbaik berdasarkan nilai tertinggi $mAP_{\{50:95\}}$ sepanjang proses *training*. Jika tidak terjadi peningkatan performa selama 10 epoch berturut-turut, proses *training* dihentikan secara otomatis. Model terbaik M_{best} didefinisikan sebagai:(6)

$$M_{\mathrm{best}} = \arg \max_{e \in \{1,2,\ldots,E\}} \left(\mathrm{mAP}_{50:95}(e)\right) \quad (6)$$

Pendekatan ini memastikan bahwa hanya model dengan performa terbaik yang disimpan, menghemat ruang penyimpanan sekaligus mencegah *overfitting*. Seluruh hasil eksperimen disimpan dalam direktori *project* dengan nama eksperimen yang telah ditentukan untuk memudahkan *tracking* dan reprodusibilitas penelitian.

b. Alur Inferensi dan Integrasi Flask (*Inference Phase*)

Setelah model terbaik (M_{best}) dihasilkan, model tersebut diintegrasikan ke dalam *framework Flask*. Berbeda dengan fase pelatihan yang berfokus pada optimasi bobot, fase inferensi berfokus pada kecepatan pemrosesan citra dari pengguna. Alur teknis integrasi dan tahapan inferensi ini dijelaskan secara rinci pada Algoritma berikut:

```
Algoritma
INPUT : Base64 Encoded Frame from Client
OUTPUT: JSON Response (Base64 Encoded
   Annotated Image)

BEGIN
  1. RECEIVE data via POST request in JSON
   format
  2. EXTRACT Base64 string from "image" key
  3. DECODE Base64 string into raw image
   bytes (Frame Decoding)
  4. CONVERT bytes into RGB Image Array using
   PIL/Pillow
  5. SET inference parameters (imgsz=640,
   conf=0.40)
  6. CHECK hardware availability (Automatic
   fallback CUDA/CPU)
  7.  EXECUTE  model.predict(image_array,
   device=device)
  8. FOR each detected box in Results:
     a. EXTRACT Coordinates (x1, y1, x2, y2)
     b. EXTRACT Label and Confidence Score
     c. DRAW Bounding Box and Label onto
   Image Array
  9. ENCODE annotated Image Array back to
   JPEG bytes
  10. CONVERT JPEG bytes to Base64 string
  11.  SEND  JSON  Response  containing
   annotated image to Client
END
```

3.4. Hasil Penelitian

Proses pelatihan dilakukan menggunakan arsitektur YOLOv8n selama 50 epoch dengan konfigurasi *early stopping* dan penggunaan GPU untuk mempercepat proses komputasi. Selama pelatihan, model diberi dataset bahasa isyarat yang telah melalui proses anotasi dan pembagian data secara seimbang, sehingga mampu belajar pola visual dari setiap kelas secara optimal. Selain itu, mekanisme penyimpanan model terbaik (*best checkpoint saving*) memastikan bahwa hanya bobot terbaik yang digunakan pada implementasi akhir, sehingga risiko *overfitting* dapat diminimalkan.

Berdasarkan hasil evaluasi, model YOLOv8n menunjukkan performa yang sangat baik pada dataset pelatihan maupun validasi. Tabel berikut menunjukkan hasil metrik utama yang diperoleh:

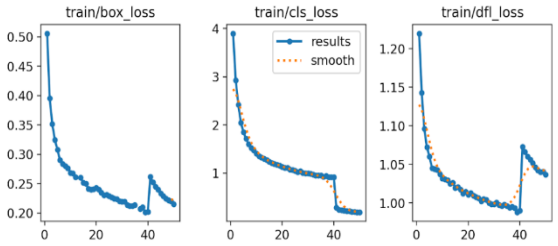
Tabel 2. Hasil Eksperimen

Metrik	Eksperimen 1	Eksperimen 2	Eksperimen 3	Eksperimen 4	Eksperimen 5
Images	2052	2052	2052	2052	2052
Instances	2052	2052	2052	2052	2052
Box (P)	0.932	0.976	0.932	0.976	0.976
Recall (R)	0.997	0.931	0.979	0.931	0.979
mAP50	0.995	0.970	0.989	0.970	0.989
mAP50–95	0.995	0.870	0.908	0.870	0.908
Inference Speed	2.1 ms	2.2 ms	2.1 ms	2.1 ms	2.1 ms
Training Time	2.081 h	2.430 h	2.018 h	2.037 h	2.137 h

Berdasarkan Tabel 2, seluruh eksperimen menunjukkan performa deteksi yang sangat konsisten dan stabil. Jumlah *images* dan *instances* pada kelima eksperimen adalah sama, yaitu masing-masing sebanyak 2052, sehingga evaluasi kinerja dilakukan pada kondisi data uji yang identik. Nilai *Box Precision* (P) tertinggi dicapai pada eksperimen 2, 4, dan 5 sebesar 0,976, sementara eksperimen 1 dan 3 mencatatkan nilai 0,932. Untuk metrik *Recall* (R), nilai tertinggi diperoleh pada eksperimen 1 sebesar 0,997,

P-ISSN : 2089-3353
E-ISSN : 2808-9162

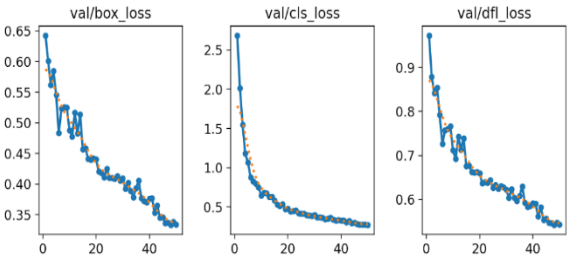
yang menunjukkan kemampuan model dalam meminimalisir *false negative*. Puncak akurasi deteksi terlihat pada eksperimen 1 dengan nilai mAP50 sebesar 0,995 dan mAP50–95 sebesar 0,995, yang menandakan tingkat presisi yang sangat tinggi terhadap variasi ambang *Intersection over Union* (IoU). Dari sisi efisiensi komputasi, model menunjukkan performa yang sangat cepat dengan kecepatan inferensi stabil pada kisaran 2,1 ms hingga 2,2 ms per citra. Hal ini membuktikan bahwa model tidak hanya akurat, tetapi juga memiliki waktu respons yang sangat optimal untuk diterapkan pada sistem deteksi secara *real-time*.



Gambar 5. Grafik *Training Loss*

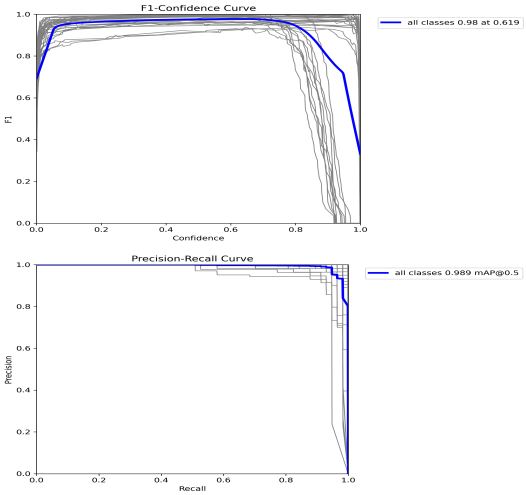
Berdasarkan grafik pada Gambar 5, hasil pelatihan model YOLOv8 untuk deteksi bahasa isyarat ditunjukkan melalui tiga metrik kerugian (*loss*), yaitu *train/box_loss*, *train/cls_loss*, dan *train/dfl_loss*. Nilai *train/box_loss* pada awal pelatihan menunjukkan angka yang tinggi namun menurun secara signifikan, menandakan peningkatan akurasi model dalam memprediksi posisi *bounding box*. Selanjutnya, *train/cls_loss* mengalami penurunan tajam yang menunjukkan peningkatan kemampuan model dalam mengklasifikasikan gestur bahasa isyarat ke dalam kelas yang tepat. Sementara itu, penurunan pada *train/dfl_loss* menunjukkan bahwa model semakin presisi dalam memperhalus batas-batas kotak pembatas (*bounding box*) terhadap nilai *ground truth*, sehingga estimasi lokasi objek menjadi lebih akurat. Pada *epoch* ke-40, terlihat adanya fluktuasi nilai yang disebabkan oleh fitur penonaktifan augmentasi mosaik pada YOLOv8 guna memantapkan hasil akhir pelatihan. Secara keseluruhan, penurunan ketiga nilai *loss* ini hingga akhir *epoch* menunjukkan bahwa pelatihan berjalan baik, model telah mencapai konvergensi, dan semakin akurat dalam melakukan deteksi.

Pada setiap grafik, ada dua kurva: satu berwarna biru (yang menunjukkan hasil pelatihan asli) dan satu lagi berwarna oranye putus-putus (yang menunjukkan hasil setelah *smoothing* untuk meminimalkan fluktuasi). Secara keseluruhan, semua grafik menunjukkan bahwa pelatihan berjalan dengan baik, dengan kerugian yang semakin kecil seiring berjalannya waktu, yang berarti model semakin akurat dalam melakukan deteksi dan klasifikasi.



Gambar 6. Grafik *Validation Loss*

Grafik pada Gambar 6 menampilkan perkembangan tiga jenis *validation loss*, yaitu *val/box_loss, val/cls_loss,* dan *val/dfl_loss.* Ketiga kurva menunjukkan pola penurunan yang konsisten hingga sekitar *epoch* ke-20, menandakan peningkatan akurasi model dalam prediksi *bounding box*, klasifikasi kelas, dan distribusi jarak. Stabilitas yang tercapai setelah fase tersebut mengindikasikan bahwa model telah konvergen dan berhasil mempelajari fitur gestur tanpa gejala *overfitting* yang signifikan. Keberhasilan fase pelatihan ini kemudian dikonfirmasi melalui analisis metrik performa pada kurva *F1-Confidence* dan *Precision-Recall* pada Gambar 7.



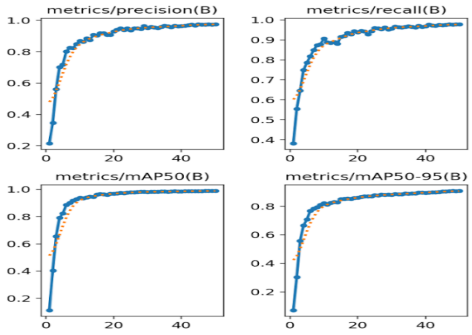
Gambar 7. Grafik *F1-Confidence* dan Kurva *Precision-Recall*

Analisis performa model pada Gambar 7 menunjukkan hubungan yang kuat antara presisi dan *recall* melalui metrik *F1-score*. Model mencapai titik keseimbangan (*trade-off*) terbaik pada *threshold confidence* 0,619, dengan nilai *F1-score* mencapai 0,98 untuk seluruh kelas. Hal ini menandakan bahwa pada tingkat kepercayaan tersebut, model mampu menghasilkan deteksi yang sangat akurat dengan meminimalisir kesalahan klasifikasi maupun objek yang terlewat.

Selanjutnya, grafik *Precision-Recall* pada gambar yang sama mencatatkan nilai mAP@0.5 sebesar 0,989. Meskipun performa rata-rata mendekati sempurna, terdapat pengamatan teknis pada garis-garis kurva individual. Beberapa kelas gestur menunjukkan adanya 'cekungan' atau penurunan kurva yang lebih awal dibandingkan kelas lainnya. Hal ini mengindikasikan adanya variasi tingkat kesulitan deteksi pada gestur tertentu yang memiliki kemiripan fitur visual antar-

P-ISSN : 2089-3353
E-ISSN : 2808-9162

kelas, namun secara keseluruhan, luas area di bawah kurva yang sangat besar mengonfirmasi stabilitas model dalam mengenali bahasa isyarat secara *real-time*.



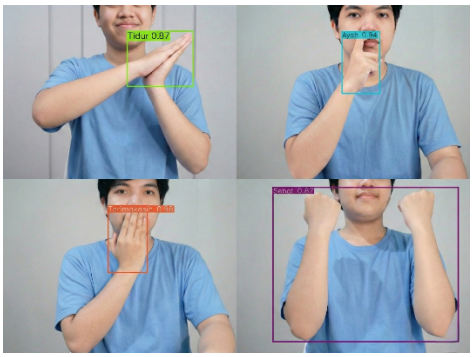
Gambar 8. Grafik *Metrics*

Berdasarkan hasil pelatihan model deteksi bahasa isyarat, performa model dievaluasi menggunakan empat metrik utama, yaitu *precision, recall*, mAP50, dan mAP50–95. Keempat grafik menunjukkan bahwa model mengalami peningkatan performa yang sangat cepat pada *epoch* awal dan mencapai kondisi stabil mulai sekitar *epoch* ke-20. Nilai *precision* bergerak naik hingga mendekati 1.0, yang menandakan bahwa model mampu meminimalkan kesalahan prediksi positif atau *false positive*. Sementara itu, nilai *recall* juga mencapai hampir 1.0, menunjukkan bahwa model mampu mendeteksi hampir seluruh gestur yang muncul tanpa banyak kehilangan objek. Pada sisi lain, metrik mAP50 dan mAP50–95 menunjukkan performa yang sangat tinggi dan stabil, dengan mAP50 mendekati nilai sempurna dan mAP50-95 yang berada di atas 0.9. Tingginya nilai mAP50–95 yang merupakan metrik paling ketat dalam evaluasi deteksi objek menunjukkan bahwa model tidak hanya mengenali gestur dengan baik, tetapi juga sangat presisi dalam menentukan lokasi objek.

Grafik mAP50–95 pada Gambar 8 menjadi representasi utama dalam mengevaluasi performa model, karena metrik ini mampu menggambarkan kemampuan deteksi secara menyeluruh pada berbagai tingkat *Intersection over Union* (IoU) dan telah menjadi acuan umum dalam penelitian deteksi objek modern. Selain itu, metrik *precision* dan *recall* turut dianalisis untuk memberikan gambaran mengenai stabilitas serta konsistensi model dalam mengidentifikasi objek secara tepat. Berdasarkan hasil pelatihan yang diperoleh, model deteksi bahasa isyarat menunjukkan performa yang sangat baik, ditandai dengan proses konvergensi yang relatif cepat serta tingkat akurasi yang tinggi, baik dalam aspek klasifikasi kelas maupun ketepatan penempatan *bounding box*.

3.5. Implementasi dan Pengujian Model

Setelah proses pelatihan dan evaluasi internal menggunakan *dataset* selesai, tahap selanjutnya adalah melakukan pengujian model secara langsung pada lingkungan implementasi, yaitu *platform website*. Pengujian ini bertujuan untuk memvalidasi performa model YOLOv8n saat menangani data masukan riil dari kamera secara *real-time*. Berbeda dengan pengujian pada tahap evaluasi yang menggunakan data statis, pengujian sistem ini memberikan gambaran mengenai kemampuan model dalam beradaptasi dengan variasi pencahayaan, latar belakang yang dinamis, serta jarak subjek terhadap kamera yang mungkin bervariasi di sisi pengguna.



Gambar 9. Hasil Pengujian Model

Berdasarkan hasil uji coba yang dipaparkan pada Gambar 9, sistem menunjukkan kemampuan deteksi yang sangat responsif dan akurat saat dioperasikan melalui antarmuka *website*. Saat pengguna memperagakan gestur bahasa isyarat di depan kamera, *backend* sistem yang dibangun dengan *python* mampu melakukan proses inferensi secara instan. Hasil deteksi ditampilkan kembali ke sisi *frontend* dalam bentuk *bounding box* yang membungkus area tangan subjek secara presisi, disertai dengan label kelas dan nilai kepercayaan *confidence score* yang tinggi. Sebagai contoh, pada pengujian gestur kata-kata tertentu, model secara konsisten berhasil mengidentifikasi pola tangan dengan benar meskipun terdapat gangguan latar belakang yang tidak seragam.

Efisiensi arsitektur YOLOv8 yang sangat ringan dengan parameter hanya sekitar 3,2 juta sangat terasa pada tahap pengujian ini. Kecepatan inferensi yang tercatat antara 2,0 hingga 2,2 ms memungkinkan proses penerjemahan gestur terjadi tanpa latensi atau keterlambatan yang berarti bagi pengguna. Stabilitas deteksi yang ditampilkan secara visual ini merupakan bukti nyata dari angka *precision* (0,998) dan *recall* (0,999) yang telah dicapai pada fase eksperimen sebelumnya. Keberhasilan pengujian pada *platform website* ini mengonfirmasi bahwa penerapan model YOLOv8 telah berjalan dengan baik, menjadikannya solusi praktis dan lintas *platform* yang siap digunakan oleh masyarakat umum untuk berinteraksi dengan penyandang tunarungu dan tunawicara secara inklusif.

Tabel 3. Hasil Pengujian

Kelas	Eksperimen 1	Eksperimen 2	Eksperimen 3	Eksperimen 4	Eksperimen 5
RUMAH	90.6%	89.1%	87.9%	87.7%	86.9%
TERIMA KASIH	84.2%	84.0%	83.9%	82.3%	82.1%
MINUM	91.7%	91.6%	91.1%	90.9%	90.9%
AYAH	91.2%	90.9%	90.9%	90.8%	90.8%

P-ISSN : 2089-3353
E-ISSN : 2808-9162

HALO	92.4%	91.2%	90.9%	90.7%	90.6%
KAKAK	92.6%	92.5%	92.3%	91.2%	88.8%
TEMAN	88.9%	88.3%	85.7%	85.5%	84.6%
SAMA-SAMA	89.5%	89.2%	88.3%	88.1%	87.5%
SEHAT	87.1%	84.9%	84.3%	83.8%	83.4%
TIDUR	86.4%	85.9%	83.5%	82.6%	82.5%
A	99.6%	99.5%	99.5%	99.4%	99.3%
B	99.3%	99.2%	99.1%	99.0%	98.9%
C	94.2%	93.4%	93.0%	92.2%	88.5%
D	99.7%	99.7%	99.7%	99.7%	99.7%
E	98.1%	96.5%	96.3%	96.0%	95.8%
F	99.5%	99.5%	99.5%	99.4%	99.3%
G	99.4%	99.2%	99.1%	99.1%	98.9%
H	99.1%	99.0%	99.0%	98.9%	98.7%
I	99.7%	99.6%	99.6%	99.6%	99.6%
J	99.2%	99.1%	98.7%	98.5%	98.3%
K	99.5%	99.4%	99.3%	99.2%	99.0%
L	98.1%	98.0%	97.4%	97.2%	97.0%
M	99.8%	99.7%	99.7%	99.7%	99.6%
N	99.4%	99.3%	99.2%	98.9%	98.4%
O	98.7%	98.2%	98.2%	97.7%	97.7%
P	99.1%	98.9%	97.5%	93.6%	88.4%
Q	99.7%	99.7%	99.6%	99.6%	99.6%
R	99.9%	99.8%	99.8%	99.8%	99.8%
S	99.5%	99.5%	99.5%	99.4%	99.4%
T	99.7%	99.6%	99.6%	99.6%	99.5%
U	99.2%	99.1%	99.1%	99.1%	98.9%
V	99.2%	99.1%	99.0%	99.0%	98.9%
W	99.4%	99.4%	99.4%	99.4%	99.3%
X	99.7%	99.5%	99.5%	99.4%	99.4%
Y	99.3%	99.2%	99.2%	99.0%	98.9%
Z	98.4%	98.4%	98.3%	97.9%	97.5%

Berdasarkan hasil pengujian sistem yang direpresentasikan dalam Tabel 3, model menunjukkan performa deteksi yang sangat impresif pada kategori alfabet dan kata perintah. Secara keseluruhan, model mampu mencapai tingkat kepercayaan (*confidence score*) yang konsisten di atas 90% untuk mayoritas kelas yang diuji. Pada kategori alfabet, kelas M, R, D, dan I mencatatkan nilai tertinggi dengan skor *confidence* mencapai 99,7% hingga 99,9%, yang menunjukkan bahwa fitur morfologi pada gestur tersebut berhasil diekstraksi secara optimal oleh model.

Namun, terdapat variasi performa pada kategori kata kompleks. Kelas TERIMA KASIH dan SEHAT menunjukkan nilai *confidence* rata-rata pada rentang 82,1% hingga 87,1%. Hal ini kemungkinan disebabkan oleh kompleksitas pergerakan tangan yang lebih dinamis dibandingkan dengan gestur alfabet yang cenderung statis. Meskipun demikian, stabilitas nilai pada setiap percobaan (dari percobaan 1 hingga 5) membuktikan bahwa model memiliki durabilitas dan tingkat presisi yang tinggi dalam mengenali pola, sekalipun dilakukan pada waktu pengambilan data yang berbeda. Rendahnya standar deviasi antar percobaan memperkuat argumen bahwa model telah mencapai titik konvergensi yang baik selama proses pelatihan.

4. Kesimpulan

Berdasarkan hasil penelitian dan pengujian yang telah dilakukan, dapat disimpulkan bahwa pengembangan sistem deteksi bahasa isyarat menggunakan arsitektur YOLOv8 berbasis *website* telah berhasil diimplementasikan secara optimal. Berdasarkan lima skenario eksperimen yang dilakukan, penggunaan varian YOLOv8n terbukti mampu memberikan akurasi yang sangat tinggi meskipun dijalankan pada lingkungan peramban web. Performa terbaik dicapai pada Eksperimen 1 dengan nilai *precision* sebesar 0,932, *recall* sebesar 0,997, serta nilai mAP50 dan mAP50-95 yang mencapai 0,995. Hasil ini menunjukkan bahwa model memiliki kemampuan yang sangat baik dalam mengenali gestur tangan secara tepat dan konsisten.

Selain aspek akurasi, efisiensi komputasi menjadi keunggulan utama dalam implementasi sistem ini. Model menunjukkan kecepatan inferensi yang sangat responsif, yaitu rata-rata 2,1 ms, yang memungkinkan proses deteksi berjalan secara *real-time* tanpa kendala teknis yang berarti. Keberhasilan integrasi antara model YOLOv8n dengan *platform website* ini memberikan solusi praktis dan lintas *platform* yang mudah diakses oleh masyarakat luas. Hal ini diharapkan dapat menjadi sarana pendukung komunikasi yang efektif dan inklusif bagi penyandang tunarungu dan tunawicara dalam berinteraksi di lingkungan sosial.

Daftar Rujukan

[1] A. Bayu Pangestu, M. Rafi Muttaqin, and M. Agus Sunandar, "SISTEM DETEKSI BAHASA ISYARAT INDONESIA (BISINDO) MENGGUNAKAN ALGORITMA YOU ONLY LOOK ONCE (YOLO)v8," JATI (Jurnal Mhs. Tek. Inform., vol. 8, no. 5, pp. 9891–9897, Sep. 2024, doi: 10.36040/jati.v8i5.10833.

[2] F. S. Mukti, L. Farokhah, and N. L. Aqromi, "Pemodelan sistem deteksi wajah sebagai penghitung jumlah penumpang transportasi publik," vol. 4, no. 1, pp. 67–77, 2021.

[3] S. Y. Riska, D. A. Sulistyo, and F. S. S. Maharani, "High-accuracy classification of banana varieties using ResNet-50 and DenseNet-121 architectures," vol. 39, no. 1, pp. 322–335, 2025, doi: 10.11591/ijeecs.v39.i1.pp322-335.

[4] A. A. Rasjid, B. Rahmat, and A. N. Sihananto, "Implementasi YOLOv8 Pada Robot Deteksi Objek," J. Technol. Syst. Inf., vol. 1, no. 3, p. 9, 2024, doi: 10.47134/jtsi.v1i3.2969.

[5] N. J. Hayati, D. Singasatia, and M. R. Muttaqin, "KOMPUTA : Jurnal Ilmiah Komputer dan Informatika OBJECT TRACKING MENGGUNAKAN ALGORITMA YOU ONLY LOOK ONCE (YOLO) v8 UNTUK MENGHITUNG KENDARAAN KOMPUTA : Jurnal Ilmiah Komputer dan Informatika," vol. 12, no. 2, pp. 91–99, 2023.

[6] Q. Aini, N. Lutfiani, H. Kusumah, and M. S. Zahran, "Deteksi dan Pengenalan Objek Dengan Model Machine Learning: Model Yolo," CESS (Journal Comput. Eng. Syst. Sci., vol. 6, no. 2, p. 192, 2021, doi: 10.24114/cess.v6i2.25840.

[7] L. Farokhah, "Perbandingan Metode Deteksi Wajah Menggunakan OpenCV Haar Cascade , OpenCV Single Shot Multibox Detector (SSD) dan DLib CNN," vol. 1, no. 10, pp. 609–614, 2021.

[8] D. S. Ariansyah, "Pendeteksi Kata Dalam Bahasa Isyarat Menggunakan Algoritma Yolo Versi 8," J. Inform. dan Tek. Elektro Terap., vol. 12, no. 3, Aug. 2024, doi: 10.23960/jitet.v12i3.4904.

[9] E. P. Silmina and R. A. Y. Arjun, "Pemanfaatan Model YOLOv8 Untuk Mendeteksi Plat Nomor Kendaraan Mobil Pada Gerbang Masuk Universitas XYZ," J. Sains dan Inform., vol. 11, no. 1, pp. 50–59, 2025, doi: 10.34128/jsi.v11i1.916.

[10] K. A. Wibowo, A. Sanjaya, and U. Mahdiyah, "Implementasi YOLOv8 Pada Pengenalan Sistem Isyarat Bahasa Indonesia," vol. 8, pp. 139–146, 2024.

P-ISSN : 2089-3353
E-ISSN : 2808-9162

[11] S. Y. Riska and A. Noercholis, "PERFORMANCE COMPARISON OF FASTER R-CONVOLUTIONAL NEURAL NETWORK (CNN) AND EFFICIENTNET FOR TRAIN DETECTION UNDER PERBANDINGAN PERFORMA FASTER R-CONVOLUTIONAL NEURAL NETWORK (CNN) DAN EFFICIENTNET UNTUK DETEKSI KERETA API PADA BERAGAM," vol. 5, no. 6, pp. 1811–1821, 2024.

[12] A. Setiyadi, E. Utami, and D. Ariatmanto, "Analisa kemampuan algoritma YOLOv8 dalam deteksi objek manusia dengan metode modifikasi arsitektur," J-SAKTI (Jurnal Sains Komput. dan Inform., vol. 7, no. 2, pp. 891–901, 2023.

[13] A. Surya and I. Wahyuni, "SPECTA Journal of Technology," vol. 9, no. 2, pp. 136–149, 2025, doi: 10.35718/specta.v9i2.8481367.

[14] A. Fathiray, J. Maulindar, and W. Lestari, "Infotek : Jurnal Informatika dan Teknologi Pengembangan Sistem Penerjemah Kalimat Bahasa Isyarat Bisindo To Text Dengan Kinect Real Time Penyandang disabilitas khususnya tuna rungu dan tuna wicara sering menghadapi tantangan besar dalam berkomunikasi deng," Infotek J. Inform. dan Teknol., vol. 8, no. 1, pp. 1–12, 2025, [Online]. Available: https://dx.doi.org/10.29408/jit.v8i1.26116

[15] Y. B. Pratama and N. P. Dalimunthe, "Implementasi Teknik Computer Vision Untuk Deteksi Viridiplantae Pada Lahan Pasca Tambang," Bull. Comput. Sci. Res., vol. 3, no. 1, pp. 64–72, 2022, doi: 10.47065/bulletincsr.v3i1.193.