Terjemahan artikel 10.22146/jnteti.v14i4.21931

Penerapan Metode *You Only Look Once* (YOLO) untuk Identifikasi Bahasa Isyarat

Reni Triyaningsih[1], Pradita Eko Prasetyo Utomo[2], Benedika Ferdian Hutabarat[2]

[1] Program Studi Sistem Informasi, Fakultas Sains dan Teknologi, Universitas Jambi, Muaro Jambi, Jambi 36361, Indonesia
[2] Program Studi Informatika, Fakultas Sains dan Teknologi, Universitas Jambi, Muaro Jambi, Jambi 36361, Indonesia

[Diserahkan: 2 Juli 2025, Direvisi: 5 Oktober 2025, Diterima: 30 Oktober 2025]
Penulis Korespondensi: Pradita Eko Prasetyo Utomo (email: pradita.eko@unja.ac.id)

INTISARI — Pemahaman yang terbatas terhadap bahasa isyarat telah memperlebar kesenjangan sosial bagi komunitas Tuli, sehingga menciptakan hambatan dalam komunikasi dan interaksi sosial. Untuk mengatasi permasalahan tersebut, diperlukan solusi berbasis teknologi guna memfasilitasi komunikasi yang inklusif. Metode deteksi berbasis *deep learning*, khususnya algoritma *You Only Look Once* (YOLO), telah memperoleh perhatian karena kecepatan dan akurasinya dalam deteksi objek secara waktu nyata. Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model pelatihan YOLO dalam mengidentifikasi Sistem Isyarat Bahasa Indonesia (SIBI). *Dataset* diperoleh dari narasumber di Sekolah Luar Biasa Negeri Prof. Dr. Sri Soedewi Masjchun Sofwan, S.H., Jambi, serta diperkaya dengan citra tambahan yang dikumpulkan dari subjek eksternal. Teknik augmentasi menggunakan Roboflow diterapkan untuk memperluas *dataset* dan beberapa skema pelatihan diimplementasikan. Kinerja model dievaluasi menggunakan *confusion matrix* dengan mempertimbangkan akurasi serta indikasi *overfitting*. Hasil penelitian menunjukkan bahwa kualitas dan kuantitas data pelatihan serta jumlah *epoch* berpengaruh signifikan terhadap akurasi model yang dihasilkan. Kinerja terbaik dicapai dengan 40 citra utama per kelas label, yang dikenai proses augmentasi menjadi 60 citra serta dilatih selama 24 *epoch*, sehingga menghasilkan akurasi *confusion matrix* sebesar 99,9%. Model yang diimplementasikan mampu mengenali isyarat SIBI secara waktu nyata menggunakan kamera web dengan proses yang cepat. Secara keseluruhan, model berbasis YOLO yang diusulkan berhasil mengidentifikasi bahasa isyarat secara waktu nyata dan menunjukkan potensi yang kuat dalam mengurangi hambatan komunikasi bagi komunitas Tuli. Meskipun demikian, penyempurnaan lebih lanjut serta perluasan *dataset* masih direkomendasikan untuk meningkatkan efektivitas dan mendukung penerapan yang lebih luas di dunia nyata.

KATA KUNCI — Bahasa Isyarat, Deteksi Waktu Nyata, *You Only Look Once*, Penyetelan Halus Model YOLO.

I. PENDAHULUAN

Bahasa isyarat merupakan bentuk komunikasi yang umum digunakan oleh komunitas Tuli dengan memanfaatkan gerakan tangan, ekspresi wajah, dan gerakan tubuh untuk membentuk simbol yang merepresentasikan huruf atau kata [1]. Sebagaimana bahasa pada umumnya, bahasa isyarat merupakan sistem yang berkembang secara alami, bersifat sistematis, dan diatur oleh kaidah linguistik [2]. Di Indonesia, penggunaan formal bahasa isyarat sebagai sarana komunikasi dan bahan ajar yang telah dibakukan oleh pemerintah adalah Sistem Isyarat Bahasa Indonesia (SIBI) [3]. SIBI merupakan bentuk komunikasi lisan yang diadaptasi ke dalam bahasa isyarat, dengan sebagian kosakata yang diambil dari American Sign Language (ASL) [4]. SIBI dikembangkan melalui kombinasi empat jenis isyarat, yaitu isyarat lokal, bentukan, ciptaan, dan serapan, yang kemudian terstandardisasi menjadi sistem isyarat nasional. Sistem ini diterbitkan oleh Kementerian Pendidikan dan Kebudayaan serta diterapkan di sekolah formal, seperti sekolah luar biasa [5]. Penggunaan SIBI dalam pendidikan bertujuan untuk membantu individu dengan disabilitas agar dapat berpartisipasi secara aktif, khususnya dalam kegiatan belajar-mengajar, sehingga dapat meningkatkan kualitas interaksi sosial. Keterbatasan pendengaran dan komunikasi tidak menghalangi penyandang tunarungu untuk melakukan aktivitas sehari-hari, termasuk melanjutkan pendidikan. Penyandang tunarungu tetap dapat berkomunikasi menggunakan bahasa isyarat. Sebaliknya, lingkungan nondisabilitas pada umumnya belajar tanpa memerlukan tenaga pengajar bahasa isyarat, sehingga tingkat keterbiasaan (*familiarity*) terhadap bahasa isyarat menjadi lebih rendah [6].

Pemahaman yang terbatas terhadap bahasa isyarat di kalangan individu nondisabilitas makin memperlebar kesenjangan sosial dengan komunitas Tuli. Kondisi ini menciptakan hambatan bagi penyandang tunarungu dalam interaksi sosial, aspek emosional, dan komunikasi [7]. Komunikasi dapat dikatakan berhasil apabila pesan yang disampaikan dan dimaksudkan oleh seseorang dapat dipahami oleh lawan bicara [8]. Namun, penyandang tunarungu cenderung mengalami kesulitan dalam berkomunikasi dengan komunitas nondisabilitas. Akibatnya, penyandang tunarungu sering dipandang berbeda oleh lingkungan nondisabilitas, yang dapat menimbulkan perasaan rendah diri dan putus asa.

Untuk mewujudkan langkah cepat dalam mengatasi kesenjangan komunikasi antara penyandang tunarungu dan individu nondisabilitas, diperlukan pengembangan suatu sistem yang tidak hanya dapat digunakan oleh tenaga pengajar, tetapi juga oleh masyarakat umum untuk membantu berkomunikasi dengan komunitas Tuli. Berbeda dengan bahasa lisan, penyampaian bahasa isyarat dilakukan melalui gerakan tubuh sebagai sarana utama komunikasi [9]. Hal ini menunjukkan bahwa bahasa isyarat dapat divisualisasikan dalam bentuk citra, sehingga gerakan tangan maupun ekspresi wajah dapat dideteksi secara otomatis melalui teknologi. Pengembangan sistem pengenalan bahasa isyarat bertujuan untuk mempermudah masyarakat dalam memahami bahasa isyarat [10].

Salah satu metode yang paling umum diterapkan untuk deteksi objek adalah metode *convolutional neural network* (CNN), yang merupakan metode jaringan saraf populer dan banyak digunakan dalam deteksi objek [11]. Namun demikian, telah dikembangkan metode deteksi objek lain yang mampu mendeteksi objek secara waktu nyata dengan tingkat akurasi dan kecepatan yang lebih tinggi, yaitu metode *You Only Look Once* (YOLO) [12]. Metode ini pertama kali dikembangkan oleh Joseph Redmon, yang mengusulkan pendekatan deteksi objek berbasis *grid* (kisi) dengan menerapkan satu jaringan saraf konvolusional [13]. YOLO dirancang sebagai algoritma model terpadu yang secara langsung mendeteksi dan mengenali objek secara keseluruhan dalam satu proses [14]. YOLO bekerja dengan membagi citra masukan menjadi *grid* berukuran $S \times S$. Jika suatu objek berada dalam sebuah sel *grid,* sel tersebut bertugas untuk mendeteksi objek tersebut [15]. Tujuan utama deteksi objek menggunakan YOLO adalah untuk menemukan dan mengidentifikasi objek dalam suatu citra berdasarkan label yang telah ditentukan sebelumnya, dengan menetapkan kelas objek serta menunjukkan posisi objek melalui pembuatan *bounding box* di sekelilingnya [16]. Metode YOLO juga mampu mendeteksi objek secara waktu nyata, sehingga sesuai untuk sistem yang memerlukan respons cepat [17].

Penelitian studi kasus sebelumnya yang menerapkan metode YOLO telah dilakukan. Salah satu penelitian mengenai pengenalan emosi pada citra wajah menggunakan metode YOLOv8 telah dilakukan [18]. Hasil penelitian menunjukkan bahwa penerapan model berhasil mendeteksi emosi melalui citra wajah secara waktu nyata dengan memproses 400 citra *dataset* yang terdiri atas emosi senang, sedih, marah, dan terkejut, dengan tingkat validasi berupa nilai *mean average precision* (mAP) mencapai 90%.

Selanjutnya, penelitian lain dilakukan untuk mendeteksi kekerasan terhadap anak dan perundungan berbasis YOLOv8 [19]. Penelitian tersebut menggunakan pendekatan *dataset* yang memuat bentuk tindakan kekerasan dan nonkekerasan dengan ukuran citra 640 piksel. Hasilnya, model mampu berjalan dan mendeteksi tindakan kekerasan dengan nilai akurasi sebesar 85%, presisi sebesar 81,8%, dan *recall* sebesar 90%.

Penelitian lain juga telah mengembangkan sistem untuk mendeteksi bahasa isyarat SIBI [20]. Penelitian tersebut menggunakan *dataset* yang berfokus pada penerjemahan isyarat alfabet dari 24 huruf yang dikelompokkan menjadi 4 kelompok, dengan masing-masing kelompok terdiri atas 20 citra *dataset*. Hasil penelitian menunjukkan bahwa penerapan metode YOLO mampu mendeteksi isyarat dengan nilai evaluasi pada kelompok 1 menghasilkan skor F1 sebesar 90,90%, kelompok 2 sebesar 97,1%, kelompok 3 sebesar 90,90%, dan kelompok 4 sebesar 83,8%.

Beberapa penelitian tersebut telah menerapkan algoritma dari model YOLO untuk mendeteksi objek dan menghasilkan nilai evaluasi yang baik. Meskipun penelitian deteksi objek dengan penerapan metode YOLO telah banyak dilakukan, belum terdapat penelitian yang secara khusus mengkaji identifikasi bahasa isyarat yang diterapkan secara waktu nyata dengan cakupan *dataset* dari tiga kategori, yaitu alfabet, angka, dan kata dasar SIBI, khususnya dengan sampel data yang diperoleh secara langsung dari narasumber di Sekolah Luar Biasa Negeri Prof. Dr. Sri Soedewi Masjchun Sofwan, S.H., Jambi. Penelitian ini mengusulkan pengembangan sistem pengenalan bahasa isyarat berbasis *computer vision* dengan menggunakan model yang dilatih menggunakan metode YOLO, yang mampu mengenali gerakan tangan secara langsung melalui kamera tanpa memerlukan perangkat bantu tambahan. Model yang dikembangkan dirancang khusus untuk beroperasi secara waktu nyata dan menghasilkan keluaran dalam bentuk teks dan audio, sehingga memungkinkan komunikasi dua arah antara penyandang tunarungu dan individu nondisabilitas, dengan tujuan agar model yang dikembangkan dapat menjadi alat bantu komunikasi yang efektif serta meningkatkan interaksi sosial antara kedua kelompok tersebut.

Dalam penelitian ini, model dilatih menggunakan data mentah yang diperoleh secara langsung, bukan *dataset* yang telah disiapkan secara khusus untuk kategori bahasa isyarat lokal. Pendekatan ini memungkinkan pengembangan sistem yang lebih realistis dan praktis, karena data yang digunakan merepresentasikan variasi nyata dalam gerakan tangan serta kondisi pencahayaan yang umum dijumpai dalam lingkungan sehari-hari.

II. METODOLOGI

Bagian ini membahas hal-hal yang berkaitan dengan alur kerja penelitian, meliputi *dataset*, praproses data, pelatihan model, evaluasi model, implementasi, serta pengujian sistem.

A. DATASET

Dalam penelitian ini, *dataset* yang digunakan untuk proses pelatihan model adalah SIBI. Data citra isyarat diperoleh secara langsung dari narasumber di Sekolah Luar Biasa Negeri Prof. Dr. Sri Soedewi Masjchun Sofwan, S.H., Jambi. Tahap awal sebelum pengambilan gambar adalah proses wawancara dengan ahli bahasa isyarat untuk menentukan kelas label isyarat yang akan dimodelkan. Hasil wawancara menunjukkan bahwa cakupan label *dataset* terdiri atas tiga kategori, yaitu alfabet, angka, dan kata dasar SIBI. Data alfabet terdiri atas huruf A–Z, dengan pengecualian huruf J yang direpresentasikan dalam bentuk gerakan. Data angka terdiri atas satu hingga sembilan, sedangkan kata dasar meliputi "bodoh," "cinta," "jahat," "kamu," "kasih," "maaf," "makan," "masuk," "minum," "nama," "rumah," "saya," "terima," "tidur," dan "tolong."

Proses pengambilan data citra dari narasumber dilakukan menggunakan kamera ponsel dengan rasio aspek 1:1 dan resolusi 12 MP. Data citra isyarat yang diperoleh dari narasumber di sekolah luar biasa digunakan sebagai data referensi untuk setiap kelas label yang telah ditentukan. Selanjutnya, untuk memperkaya konten *dataset*, data citra juga dikumpulkan dari subjek selain narasumber utama melalui pengambilan gambar secara langsung menggunakan kamera ponsel dan laptop. Kamera laptop yang digunakan memiliki rasio aspek 16:9 dengan resolusi 0,9 MP. Selama proses pengambilan gambar, kamera ditempatkan pada jarak sekitar 50–70 cm dari subjek, dengan ketinggian sejajar dengan dada subjek. Kondisi pencahayaan dijaga menggunakan cahaya alami siang hari atau pencahayaan dalam ruangan yang merata untuk meminimalkan bayangan dan memastikan visibilitas gerakan tangan yang jelas. Dari proses ini, diperoleh 70 citra utama untuk setiap kelas label yang telah ditentukan.

B. PRAPEMROSESAN DATA

Prapemrosesan data merujuk pada proses mengonversi data mentah menjadi format yang lebih mudah dipahami oleh mesin. Prapemrosesan data dilakukan pada platform Roboflow yang menyediakan antarmuka berbasis web dengan tahapan pelabelan citra, pembagian *dataset*, serta augmentasi data [21].

Pelabelan citra dilakukan dengan cara menganotasi dan mengelompokkan data sesuai dengan nama kelas setiap objek dalam citra untuk menyimpan informasi yang berkaitan dengan citra tersebut. Proses pelabelan citra dilakukan secara manual dengan memberikan *bounding box* pada objek citra satu per satu guna memastikan akurasi anotasi.

Data yang telah dianotasi dan diberi label kelas kemudian dibagi menjadi tiga subset, yaitu subset pelatihan, validasi, dan pengujian: sebesar 70% digunakan sebagai data utama untuk melatih model, 20% dialokasikan sebagai subset validasi untuk memantau kinerja model selama proses pelatihan, dan 10% digunakan sebagai subset pengujian untuk menguji kinerja model yang telah dilatih.

Tahap akhir prapemrosesan data adalah penerapan augmentasi untuk memperluas variasi data tanpa menambah jumlah data utama. Metode augmentasi yang digunakan meliputi *crop*, yaitu pemotongan sebagian area citra, sehingga posisi atau ukurannya berubah; *grayscale*, yaitu penghilangan informasi warna dari citra agar model lebih berfokus pada bentuk dan pola; *blur*, yaitu penerapan efek kabur untuk menyimulasikan kondisi kamera tidak fokus; *noise*, yaitu penambahan gangguan visual acak pada citra, seperti titik-titik kecil atau distorsi ringan; serta *brightness*, yaitu penyesuaian tingkat kecerahan citra untuk menyimulasikan kondisi pencahayaan yang berbeda. Proses augmentasi data tidak mencakup transformasi *flip* citra, karena terdapat kesamaan isyarat pada dua kelas label yang berbeda dan dibedakan berdasarkan posisi tangan saat pengambilan gambar.

C. PELATIHAN MODEL

Proses pelatihan model dilakukan menggunakan model YOLOv11, yang secara resmi dirilis pada 30 September 2024 dan digunakan sebagai kerangka utama deteksi. YOLOv11 dipilih karena merupakan pengembangan terbaru dalam keluarga YOLO yang menawarkan peningkatan signifikan dalam kinerja deteksi objek secara waktu nyata dibandingkan dengan versi sebelumnya, seperti YOLOv8, YOLOv9, dan YOLOv10. Di antara varian YOLOv11 yang tersedia (*nano, small, medium, large,* dan *x-large*), model YOLOv11n (*nano*) dipilih karena arsitekturnya yang ringan, sehingga memungkinkan kecepatan inferensi yang lebih tinggi serta implementasi waktu nyata pada sumber daya *graphic processing unit* (GPU) yang terbatas, dengan tetap mempertahankan akurasi yang kompetitif.

Google Colab digunakan sebagai perangkat lunak untuk menjalankan proses pelatihan model. Pelatihan model memanfaatkan *runtime* GPU Tesla T4 untuk mempercepat proses eksekusi. Tabel I menunjukkan kombinasi hiperparameter yang digunakan untuk mendukung keberhasilan proses pelatihan. *Dataset* dilatih dengan menerapkan prosedur pelatihan bertahap dengan jumlah maksimum *epoch* sebanyak 100. Pendekatan ini bertujuan untuk memperoleh nilai *epoch* terbaik yang menghasilkan model dengan kinerja optimal. Apabila proses pelatihan menghasilkan model dengan nilai evaluasi dalam kategori optimal, proses pelatihan dihentikan meskipun jumlah *epoch* belum mencapai nilai maksimum. Strategi ini diterapkan untuk menghindari risiko *overfitting*, yaitu kondisi ketika model terlalu spesifik terhadap data pelatihan, sehingga mengakibatkan model kurang mampu menangani data baru karena cenderung menghafal data pelatihan, alih-alih mempelajari pola dari data tersebut.

TABEL I
KOMBINASI HIPERPARAMETER

No.	Parameter	Nilai
1.	*Epoch*	Maksimum 100
2.	Ukuran *batch*	16
3.	IoU	0,6
4.	Laju belajar	0,01

D. EVALUASI MODEL

Evaluasi terhadap model yang telah dilatih dilakukan dengan mengamati kurva pelatihan model serta nilai *confusion matrix*. *Confusion matrix* memvisualisasikan distribusi kesalahan prediksi model dalam satu tampilan. Berdasarkan nilai dasar hasil prediksi *confusion matrix*, dapat diperoleh nilai matriks evaluasi yang meliputi presisi, *recall*, akurasi, skor F1, serta mAP. Hasil perhitungan nilai matriks evaluasi tersebut digunakan untuk mengukur kinerja keseluruhan model [22]. Perhitungan nilai evaluasi bergantung pada nilai dasar hasil prediksi *confusion matrix*, yang merupakan komponen utama untuk memperoleh nilai perhitungan matriks evaluasi lainnya, yaitu *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Dalam kategori perhitungan presisi, nilai ditentukan berdasarkan jumlah elemen yang benar-benar positif dibandingkan dengan total elemen yang diprediksi positif.

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

Matriks evaluasi lainnya adalah *recall*, yang mengukur kemampuan model dalam mengidentifikasi seluruh elemen positif dalam suatu *dataset* [23]. Nilai *recall* dihitung dengan mengukur rasio antara elemen TP terhadap jumlah total elemen yang seharusnya diprediksi sebagai positif.

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

Elemen evaluasi lain yang menjadi poin penting dalam menentukan optimasi model adalah akurasi, yang mengukur seberapa baik algoritma bekerja dengan menganggap setiap elemen data memiliki bobot yang sama dan berkontribusi secara setara terhadap nilai akurasi.

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN}. \quad (3)$$

Dari perhitungan ketiga matriks evaluasi tersebut, nilai skor F1 juga dapat dihitung, yang merupakan bagian dari ukuran parametrik F yang paling sering digunakan. Skor F1 dihitung dengan menentukan nilai rata-rata perbandingan antara presisi dan *recall* [24].

$$Skor\ F1 = 2 \times \frac{Presisi \times Recall}{Presisi + Recall}. \quad (4)$$

Nilai matriks evaluasi terakhir yang dipertimbangkan dalam penelitian ini sebagai penentu optimasi model adalah mAP, yang merupakan rata-rata *average precision* (AP). Nilai AP diperoleh dari perhitungan antara presisi dan *recall*.

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k. \quad (5)$$

Dengan mengombinasikan seluruh nilai matriks evaluasi yang telah dihitung, diperoleh gambaran menyeluruh mengenai kinerja model yang telah dilatih. Model dengan hasil evaluasi terbaik kemudian digunakan untuk implementasi dalam deteksi bahasa isyarat secara waktu nyata.

E. IMPLEMENTASI DAN PENGUJIAN SISTEM

Implementasi model ke dalam sistem waktu nyata bertujuan untuk memberikan akses kepada pengguna agar dapat berinteraksi secara langsung dengan model. Model hasil pelatihan yang telah melalui proses evaluasi dan memiliki kinerja terbaik diimplementasikan secara waktu nyata dengan memanfaatkan *computer vision*. Penerapan *computer vision* memungkinkan komputer untuk "melihat" dan menangkap objek dalam lingkungan visual [25]. Pemanfaatan *computer vision* dalam bidang *deep learning* dapat dibagi menjadi klasifikasi, segmentasi, deteksi, dan generasi, yang memberikan manfaat untuk berbagai keperluan, seperti keamanan, penyediaan informasi, dan pemantauan [26]. Salah satu tugas yang paling umum digunakan dalam *computer vision* adalah deteksi objek, yaitu kemampuan komputer untuk mengidentifikasi objek visual yang termasuk dalam kelas tertentu, guna menentukan keberadaan objek dari kategori tertentu, seperti manusia, kendaraan, atau hewan [27]. Dalam penelitian ini, model diimplementasikan menggunakan pustaka sumber terbuka yang umum digunakan pada sistem berbasis *computer vision*, yaitu OpenCV. Pustaka ini menyediakan berbagai fungsi, seperti pelacakan gerakan, deteksi objek, pengenalan wajah, dan segmentasi. Dengan menggunakan OpenCV, citra atau video waktu nyata dapat diproses sesuai dengan kebutuhan sistem [28]. Model yang telah dilatih diintegrasikan ke dalam kode Python dan pustaka OpenCV digunakan untuk mengakses kamera web sebagai alat penangkap citra secara langsung. Hasil identifikasi model ditampilkan pada layar perangkat dalam bentuk teks. Teks yang teridentifikasi kemudian dikonversi menjadi audio menggunakan pustaka Google text-to-speech (gTTS). Pustaka gTTS merupakan alat yang mengubah teks menjadi format MP3, sehingga dapat disimpan sebagai berkas audio [29]. Audio yang dihasilkan selanjutnya diputar menggunakan pustaka Pygame, khususnya modul pygame.mixer, yang digunakan untuk memutar berkas audio [30].

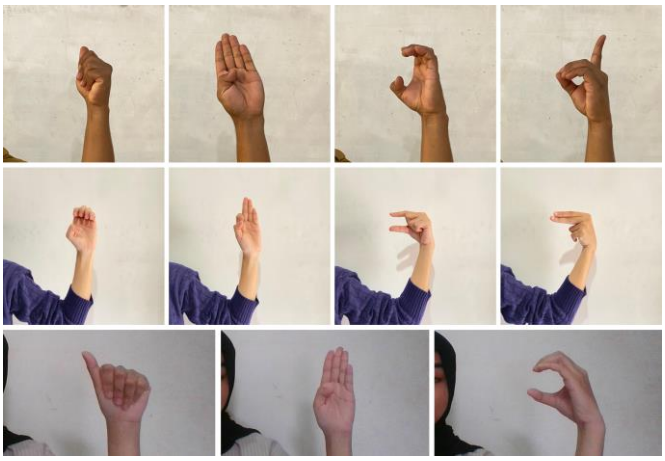
Sistem yang telah berhasil dijalankan kemudian diuji pada tahap akhir untuk memastikan bahwa sistem dapat digunakan dalam lingkungan nyata. Dalam pengujian tersebut, terdapat empat aspek utama yang dievaluasi, yaitu keresponsifan (*responsiveness*), akurasi, ketegaran (*robustness*), dan sistem suara. Keresponsifan diukur dengan menghitung nilai *frame per second* (FPS) untuk setiap kelas label saat sistem berjalan menggunakan fungsi waktu yang disediakan oleh pustaka OpenCV, guna memastikan model dapat mengidentifikasi objek tanpa jeda yang signifikan. Akurasi digunakan untuk menilai tingkat kesesuaian hasil deteksi model dengan isyarat yang dilakukan oleh pengguna. Ketegaran mengevaluasi kemampuan sistem dalam mengenali objek isyarat pada berbagai kondisi pencahayaan, latar belakang, dan distorsi lainnya. Sementara itu, sistem suara memastikan bahwa teks yang diinterpretasikan menjadi audio dapat berjalan secara sinkron dan berfungsi dengan baik.

III. HASIL DAN PEMBAHASAN

Bagian ini menjelaskan tahapan-tahapan yang telah dilakukan dalam penelitian untuk memperoleh hasil akhir penelitian.

A. AKUISISI DATASET

Akuisisi *dataset* dilakukan melalui dua prosedur dalam pengambilan data citra. Pada prosedur pertama, sampel data yang menjadi acuan isyarat untuk setiap kelas label diperoleh secara langsung dari narasumber di Sekolah Luar Biasa Negeri Prof. Dr. Sri Soedewi Masjchun Sofwan, S.H., Jambi. Proses pengumpulan data dilakukan menggunakan kamera ponsel, sehingga menghasilkan 49 citra data utama yang sesuai dengan jumlah kelas label yang telah ditentukan. Selanjutnya, prosedur kedua dilakukan untuk memperluas jumlah data menjadi 70 citra data utama untuk setiap kelas label dengan menambahkan citra yang diambil dari subjek di luar narasumber, dengan tetap mempertahankan kesesuaian isyarat berdasarkan citra acuan. Citra pada prosedur kedua diambil menggunakan kamera ponsel dan kamera laptop untuk menghasilkan *dataset* yang lebih beragam dan representatif. *Dataset* yang digunakan dalam penelitian ini tidak dapat dipublikasikan secara terbuka karena pertimbangan privasi dan ukurannya yang besar. Namun demikian, Gambar 1 menyajikan hasil akuisisi data citra sebagai gambaran representatif dari *dataset*. Data yang diperoleh disimpan dalam satu folder utama, dengan setiap kelas label ditempatkan pada subfolder terpisah sesuai dengan kelas label masing-masing.



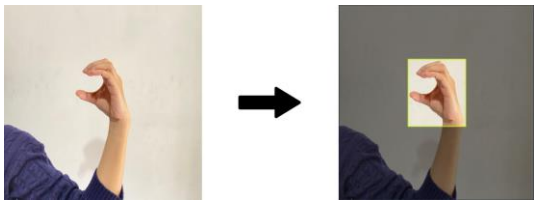
Gambar 1. Hasil akuisisi data citra.

B. PRAPEMROSESAN DATASET

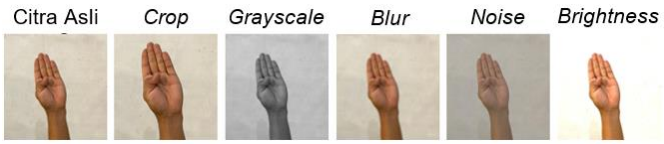
Data yang telah dikumpulkan dipersiapkan untuk pelatihan model dengan melakukan anotasi pada setiap kelas label di seluruh citra menggunakan Roboflow. Setiap citra dianotasi secara individual sesuai dengan kelas masing-masing dan hasilnya divalidasi secara langsung oleh penulis bersama narasumber dari Sekolah Luar Biasa Negeri Prof. Dr. Sri Soedewi Masjchun Sofwan, S.H., Jambi untuk memastikan akurasi pelabelan. Untuk kategori alfabet dan angka, pelabelan difokuskan pada area tangan yang membentuk isyarat bahasa isyarat. Sementara itu, untuk kelas kata dasar, pelabelan mencakup bagian tubuh yang terlibat dalam isyarat, seperti wajah, dada, dan lengan, bergantung pada karakteristik masing-masing isyarat. Gambar 2 menampilkan proses pelabelan citra yang dilakukan pada platform Roboflow.

Setelah tahap pelabelan selesai, *dataset* yang telah dianotasi dibagi menjadi tiga subset, yaitu pelatihan, validasi, dan pengujian. Jumlah keseluruhan citra data utama untuk setiap kelas label adalah 70, sehingga keseluruhan citra data utama yang diperoleh dalam *dataset* untuk 49 kelas label adalah 3.430 citra. Dari jumlah tersebut, sebesar 70% (2.399 citra) dialokasikan untuk subset pelatihan, 20% (687 citra) untuk subset validasi, dan 10% (344 citra) untuk subset pengujian.

Tahap akhir prapemrosesan melibatkan augmentasi citra untuk meningkatkan *dataset* pada subset pelatihan. Proses augmentasi tetap dilakukan pada platform Roboflow, yang juga dapat mengubah ukuran citra menjadi 640 × 640 piksel,



Gambar 2. Anotasi data citra.



Gambar 3. Augmentasi *dataset*.

sehingga seluruh citra memiliki ukuran yang seragam. Gambar 3 menampilkan variasi augmentasi citra yang diterapkan dalam *dataset*, meliputi *crop*, *grayscale*, *blur*, *noise*, dan penyesuaian *brightness*. Hasil akhir prapemrosesan menghasilkan 5.829 data citra pada subset pelatihan. Jumlah ini diperoleh karena proses augmentasi hanya diterapkan pada subset pelatihan, sedangkan subset validasi dan pengujian tidak diberikan augmentasi agar hasil evaluasi tetap objektif dan merepresentasikan kondisi nyata.

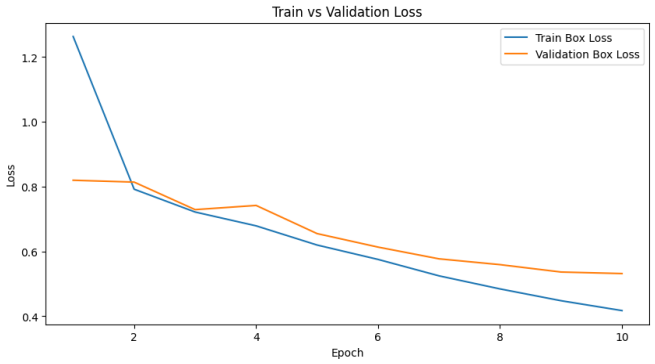
C. PELATIHAN MODEL

Proses pelatihan model dilakukan melalui siklus *epoch* bertahap, dengan seluruh *dataset* diproses pada setiap putaran. Pada pelatihan pertama, model dilatih selama 10 *epoch* dengan ukuran *batch* sebesar 16, nilai *intersection over union* (IoU) sebesar 0,6, dan laju belajar sebesar 0,01. Namun, hasil pelatihan yang diperoleh belum menunjukkan kinerja yang baik karena selama proses pelatihan, model menunjukkan indikasi *overfitting*. Hal ini dapat dilihat dari perbedaan yang signifikan antara nilai pada kurva *training loss* yang terus menurun hingga mencapai titik rendah. Sementara itu, Gambar 4 menunjukkan bahwa nilai pada kurva *validation loss* cenderung tidak mengalami peningkatan yang signifikan. Nilai *loss* pada *train box loss* secara konsisten menurun hingga akhir proses pelatihan, yang mengindikasikan bahwa model mampu mempelajari dan menyesuaikan parameter terhadap data pelatihan. Namun demikian, nilai *loss* pada *validation box loss* tetap stagnan dan tidak mengalami penurunan setelah beberapa *epoch*. Mulai dari *epoch* ke-4 hingga *epoch* ke-10, jarak antara kurva *train box loss* dan *validation box loss* terlihat makin jelas, yang menunjukkan terjadinya *overfitting* pada model.

Kondisi ini perlu diperbaiki melalui penerapan prosedur model *fine-tuning*, yaitu pelatihan ulang model pralatih (*pretrained model*) dengan menggunakan data yang telah diperbaiki. Penerapan prosedur *fine-tuning* memungkinkan efisiensi waktu dalam proses pelatihan dibandingkan dengan melatih model dari awal (*from scratch*). Proses *fine-tuning* dilakukan dengan memperbaiki *dataset* serta kombinasi parameter yang digunakan selama pelatihan guna memperoleh hasil pelatihan yang lebih optimal.

D. FINE–TUNING

Proses *fine-tuning* diawali dengan pemeriksaan kualitas seluruh citra dalam *dataset* yang dihasilkan dari tahap prapemrosesan. Hasil analisis menunjukkan bahwa data hasil prapemrosesan yang diambil menggunakan kamera laptop mengalami penurunan kualitas akibat proses pengubahan ukuran (*resizing*). Citra tampak menyempit dan bentuk isyarat



Gambar 4. Kurva *train loss* dan *validation loss* dari model awal.

menjadi tidak proporsional. Oleh karena itu, dilakukan seleksi ulang data utama dengan mempertahankan citra yang memiliki konsistensi proporsi dan bentuk objek untuk penyempurnaan model. Dari total 3.430 citra data utama dengan rata-rata 70 citra per kelas label, diperoleh 40 citra dengan kualitas terbaik untuk setiap kelas label, sehingga keseluruhan data yang digunakan menjadi 1.960 citra. Seluruh data tersebut diambil menggunakan kamera ponsel dengan rasio 1:1. Tabel II menunjukkan distribusi data sebelum dan sesudah proses seleksi citra berkualitas tinggi. Citra terpilih kemudian melalui tahap prapemrosesan data kembali, tetapi dengan teknik yang berbeda. Tahap prapemrosesan untuk menyiapkan *dataset* yang digunakan dalam proses *fine-tuning* memanfaatkan pustaka dalam Python sebelum dilakukan anotasi ulang pada platform Roboflow.

1) PRAPEMROSESAN *FINE–TUNING DATASET*

Tahap awal dalam prapemrosesan data adalah penerapan teknik penghilangan latar belakang (*background removal*) pada citra. *Dataset* hasil seleksi pada tahap *fine-tuning* diunggah ke dalam folder Google Drive dan diproses menggunakan Google Colab. Tujuan penghilangan latar belakang adalah menghilangkan elemen latar, sehingga model dapat berfokus pada objek utama dalam citra tanpa terganggu oleh elemen lain yang tidak relevan. Proses ini dilakukan dengan membuka seluruh citra dari folder masukan menggunakan kode Python dan mengeksekusinya dengan pustaka rembg. Hasil penghilangan latar belakang kemudian disimpan menggunakan fungsi os.makedirs ke dalam folder hasil *remove background*.

Tahap berikutnya adalah proses pengubahan ukuran untuk menyeragamkan ukuran citra menjadi 640 × 640 piksel agar proses pelatihan lebih terstruktur dan sesuai dengan kebutuhan masukan model. Proses pengubaharn ukuran dilakukan dengan mengambil data citra dari folder hasil penghilangan latar belakang. Proses ini memanfaatkan Python Imaging Library (PIL), dengan metode thumbnail() untuk mempertahankan rasio aspek citra. Hasil pengubahan ukuran kemudian disimpan ke dalam folder hasil pengubahan ukuran baru.

Untuk memastikan citra yang digunakan dalam pemodelan memiliki kualitas visual yang baik, tidak kabur, dan tetap mempertahankan detail isyarat, dilakukan proses pemeriksaan kualitas citra menggunakan metode varians Laplacian. Metode ini digunakan untuk mendeteksi tingkat kekaburan berdasarkan perhitungan variasi nilai piksel dengan menetapkan nilai ambang sebesar 50. Citra yang memenuhi kriteria kemudian disimpan ke dalam folder akhir hasil prapemrosesan menggunakan Python.

Folder terakhir yang berisi hasil akhir prapemrosesan menggunakan Python kemudian diunduh dan dilakukan

TABEL II
DISTRIBUSI *DATASET*

Katagori	Jumlah Kelas	Rata-rata Citra Awal per Kelas	Total Citra	Rata-rata Citra Terpilih per Kelas	Total Citra
Alfabet	25	70	1.750	40	1.000
Angka	9	70	630	40	360
Kata Dasar	15	70	1.050	40	600
Total	49		3.430		1.960

TABEL III
SKENARIO PELATIHAN *FINE-TUNING* MODEL

	Jumlah Data Utama per Kelas Label				Jumlah Data setelah Augmentasi per Kelas Label				*Epoch* Terbaik
	Pelatihan	Validasi	Pengujian	Total	Pelatihan	Validasi	Pengujian	Total	
Skenario pertama	7	2	1	10	14	2	1	17	21
Skenario kedua	14	4	2	20	28	4	2	34	21
Skenario ketiga	22	6	2	30	44	6	2	52	25
Skenario keempat	30	8	2	40	60	8	2	70	24

pelabelan ulang, pembagian *dataset*, serta augmentasi tanpa melalui kembali proses pengubahan ukuran pada platform Roboflow. Proses pelabelan, pembagian, dan augmentasi citra mengikuti prosedur prapemrosesan data sebelum tahap *fine-tuning*, tetapi dilakukan secara bertahap dengan penambahan 10 data, sehingga membentuk kelompok data yang terdiri atas 10, 20, 30, dan 40 citra data utama untuk setiap kelas label. Hasil akhir dari prapemrosesan data berupa empat folder terkompresi (*zipped folder*) yang dihasilkan dari proses pelabelan, pembagian, dan augmentasi sesuai dengan jumlah data utama yang digunakan.

2) MODEL *FINE–TUNING*

Proses *fine-tuning* model menerapkan metode empat skenario pelatihan data secara bertahap, dimulai dari pelatihan menggunakan 10 citra data utama hingga 40 citra data utama untuk setiap kelas label. Pendekatan ini diterapkan sebagai upaya untuk mengantisipasi risiko *overfitting*, terutama mengingat jumlah data yang terbatas. Model dengan kinerja terbaik kemudian digunakan sebagai dasar untuk implementasi sistem. Tabel III menunjukkan distribusi data untuk setiap skenario *fine-tuning* model serta *epoch* terbaik yang diperoleh. Selama proses *fine-tuning* model, hiperparameter pelatihan disesuaikan kembali agar sesuai dengan *dataset*. Setelah beberapa kali percobaan pelatihan, konfigurasi hiperparameter optimal untuk setiap skenario *fine-tuning* berhasil diidentifikasi. Hiperparameter yang dipilih meliputi 40 *epoch*, nilai *patience* sebesar 3, ukuran *batch* sebesar 16, serta nilai IoU sebesar 0,6. Selain itu, laju belajar diturunkan menjadi 0,001, dengan parameter *freeze* diatur sebesar 1 dan *dropout* sebesar 0,3.

Proses pelatihan dengan menerapkan mekanisme *early stopping* dengan nilai *patience* sebesar 3 bertujuan untuk menghentikan proses pelatihan lebih awal apabila tidak terjadi peningkatan kinerja selama penambahan *epoch*. Parameter *freeze* dan *dropout* diterapkan untuk mencegah *overfitting* dengan cara menghilangkan sebagian neuron secara acak selama proses pelatihan model. Selama proses pelatihan, meskipun jumlah maksimum *epoch* ditetapkan sebanyak 40, pada praktiknya hasil terbaik diperoleh pada jumlah *epoch* yang tidak melebihi 30. Sebagai contoh, pada model dalam skenario keempat, hasil terbaik diperoleh pada *epoch* ke-24. Meskipun proses pelatihan dilanjutkan setelah *epoch* tersebut, tidak ditemukan peningkatan kinerja yang signifikan, sehingga *epoch* ke-24 dianggap sebagai *epoch* optimal.

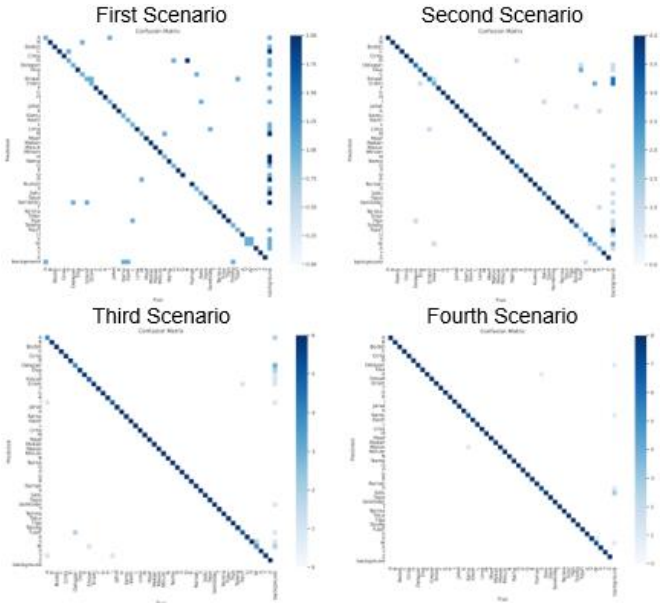
E. EVALUASI MODEL

Evaluasi model dilakukan untuk menentukan model terbaik di antara seluruh skema pelatihan model yang telah dilakukan. Model dapat dikategorikan sebagai model yang baik apabila hasil evaluasi menunjukkan kinerja yang optimal serta tidak terdapat indikasi overfitting selama proses pelatihan. Nilai matriks evaluasi yang tinggi saja tidak cukup untuk menyimpulkan bahwa model layak untuk diimplementasikan. Model yang mengalami overfitting juga dikategorikan tidak layak, karena cenderung mengenali pola dari data pelatihan dan mengalami kesulitan dalam mengidentifikasi isyarat ketika diterapkan pada lingkungan nyata. Indikasi ada atau tidaknya *overfitting* selama proses pelatihan dapat diamati melalui kurva evaluasi *train loss* dibandingkan dengan *validation loss.*

Gambar 5 menampilkan hasil evaluasi proses pelatihan berdasarkan kurva *train loss* dibandingkan dengan *validation loss*, yang menunjukkan bahwa model pada skenario pertama belum bekerja secara optimal karena *validation loss* tidak menunjukkan peningkatan yang konsisten dan cenderung menyimpang dari *training loss*. Nilai *validation loss* pada kurva bersifat tidak stabil dan sering berfluktuasi naik turun selama proses pelatihan, yang mengindikasikan bahwa model belum mampu belajar secara optimal dari data yang terbatas. Namun demikian, pada proses pelatihan berikutnya dalam skenario kedua hingga keempat, kurva mulai menunjukkan pola penurunan yang lebih stabil, yang mengindikasikan bahwa nilai kesalahan (*loss*) pada data pelatihan dan validasi sama-sama menurun. Hal ini menunjukkan bahwa proses pelatihan model berjalan dengan baik tanpa indikasi *overfitting*. Dengan demikian, dapat disimpulkan bahwa peningkatan jumlah data pada subset pelatihan secara efektif dapat meningkatkan kinerja model.

Kinerja masing-masing model juga dapat dilihat melalui nilai *confusion matrix*, yang memberikan gambaran mengenai kemampuan model dalam memprediksi isyarat dengan benar. *Confusion matrix* merupakan tabel yang menunjukkan jumlah prediksi model untuk setiap kelas label dan membandingkannya dengan jawaban benar yang diprediksi (label aktual). Nilai *confusion matrix* merupakan komponen utama dalam perhitungan matriks evaluasi lainnya. Gambar 6 menampilkan distribusi hasil prediksi untuk setiap skenario *fine-tuning* model pada seluruh kelas label. Hasil prediksi keluaran tersebut kemudian dihitung kembali untuk memperoleh metrik evaluasi tambahan, meliputi presisi, *recall*,

Gambar 5. Kurva *train loss* dibandingkan dengan *validation loss* pada model hasil *fine-tuning*.



Gambar 6. Visualisasi prediksi *confusion matrix* pada model hasil *fine-tuning*.

akurasi, skor F1, dan mAP. Seluruh gambar, termasuk kurva pelatihan dan *confusion matrix*, dihasilkan secara langsung dari keluaran Python selama proses pelatihan model.

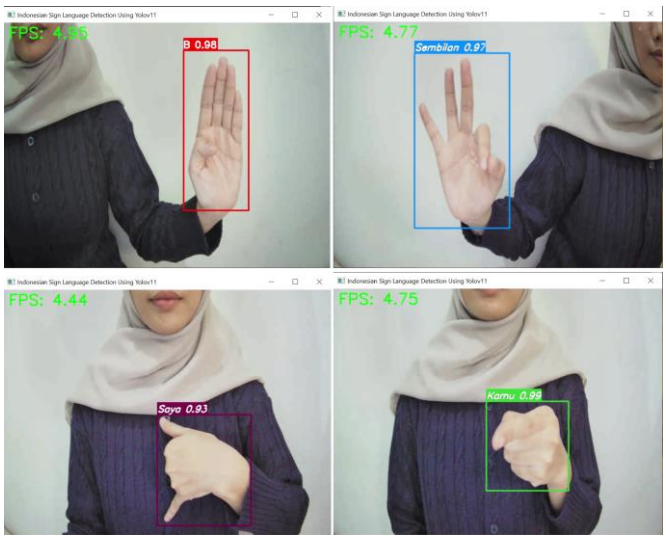
Tabel IV menunjukkan nilai rata-rata dari setiap komponen evaluasi untuk masing-masing skenario *fine-tuning* model berdasarkan hasil perhitungan nilai prediksi. Berdasarkan hasil evaluasi, model dengan kinerja terbaik diperoleh dari *fine-tuning* pada skenario keempat, dengan proses pelatihan menggunakan 40 data utama untuk setiap kelas label, yang kemudian diaugmentasi menjadi 60 citra untuk setiap kelas label, serta dengan *epoch* terbaik pada *epoch* ke-24. Hasil pelatihan menunjukkan tidak adanya indikasi *overfitting*, dengan nilai matriks evaluasi presisi mencapai 99,4%, *recall* sebesar 97,8%, akurasi sebesar 99,9%, skor F1 sebesar 98,7%, serta nilai mAP sebesar 94,3%.

F. PENGUJIAN DAN IMPLEMENTASI SISTEM

Implementasi model dalam sistem waktu nyata dibangun menggunakan kode program Python dengan bantuan Notepad sebagai editor teks. Kombinasi beberapa pustaka yang diperlukan, seperti OpenCV, digunakan untuk mengakses kamera web, membaca dan menampilkan *frame* video, serta menampilkan hasil deteksi. Selanjutnya, pustaka gTTS digunakan untuk mengonversi teks hasil deteksi menjadi audio dan dengan bantuan pustaka pygame, audio tersebut diputar secara langsung. Pustaka waktu juga digunakan untuk

TABEL IV
NILAI RATA-RATA MATRIKS EVALUASI

	Model Pralatih			
	Skenario Pertama	***Skenario Kedua***	***Skenario Ketiga***	***Skenario Keempat***
Presisi	0,770	0,929	0,975	0,994
Recall	0,695	0,935	0,956	0,978
Akurasi	0,989	0,989	0,997	0,999
Skor F1	0,816	0,904	0,948	0,987
mAP	0,911	0,894	0,928	0,943



Gambar 7. Tampilan sistem identifikasi bahasa isyarat.

menghitung dan menampilkan nilai FPS sebagai indikator kecepatan deteksi.

Model dijalankan menggunakan perangkat CPU lokal melalui Anaconda Prompt sebagai antarmuka untuk menjalankan lingkungan Python dan skrip deteksi yang telah disiapkan. Implementasi ini sepenuhnya dilakukan pada perangkat keras lokal tanpa bergantung pada sumber daya berbasis awan. Gambar 7 menampilkan keluaran implementasi sistem identifikasi bahasa isyarat SIBI.

Setelah sistem diimplementasikan, dilakukan pengujian untuk menentukan kemampuan sistem dalam mengidentifikasi bahasa isyarat dengan baik dalam lingkungan nyata. Pengujian ini bertujuan untuk mengevaluasi kinerja keseluruhan sistem saat dijalankan dengan masukan citra dari kamera secara langsung. Aspek keresponsifan diuji dengan mengukur tidak adanya jeda signifikan dalam mendeteksi isyarat untuk setiap kelas label selama proses deteksi. Keresponsifan kemudian dihitung berdasarkan nilai FPS yang diperoleh dari rata-rata 10 kali percobaan deteksi untuk setiap kelas label. Aspek akurasi dievaluasi dengan menjalankan sistem dan mencatat hasil deteksi. Deteksi dianggap akurat apabila isyarat yang diidentifikasi sesuai dengan isyarat yang dilakukan serta memiliki *confidence score* > 80%. Aspek ketegaran dinilai dengan mengoperasikan sistem pada berbagai kondisi latar belakang dan pencahayaan. Keberhasilan deteksi dalam kondisi tersebut dianggap sebagai indikator keberhasilan. Terakhir, sistem keluaran suara diuji untuk memastikan bahwa audio yang dihasilkan tetap sinkron dengan teks yang dikenali selama proses operasi.

Setelah sistem dijalankan dan diuji, hasil menunjukkan bahwa keresponsifan sistem dalam mendeteksi setiap kelas label relatif konsisten, dengan rata-rata sebesar 5 FPS. Dalam penelitian ini, tidak ditetapkan nilai FPS minimum karena

fokus utama adalah memverifikasi kemampuan sistem dalam melakukan deteksi secara waktu nyata tanpa jeda yang terlihat. Meskipun nilai FPS relatif rendah akibat implementasi berbasis CPU, model tetap terbukti efektif dalam mengenali isyarat bahasa isyarat. Peningkatan FPS di masa mendatang dapat dicapai melalui penambahan data pelatihan, penerapan augmentasi yang lebih beragam, serta penggunaan perangkat keras yang lebih mumpuni.

Secara keseluruhan, hasil deteksi untuk seluruh kelas label menunjukkan akurasi yang baik, dengan *confidence score* konsisten > 80%. Namun demikian, beberapa kelas label, yaitu "v," "w," "dua," dan "enam," menunjukkan akurasi yang lebih rendah akibat sering terjadinya kesalahan klasifikasi yang disebabkan oleh kemiripan bentuk isyarat. Kelas-kelas tersebut hanya mencapai *confidence score* dalam kisaran 50% hingga 70%.

Selanjutnya, hasil pengujian ketegaran menunjukkan bahwa kategori kelas alfabet dan angka masih memiliki keterbatasan dalam kemampuan deteksi karena sistem hanya mampu mendeteksi label tersebut pada kondisi latar belakang yang bersih. Hal ini merupakan dampak dari *dataset* yang digunakan dalam proses pelatihan model, yang memiliki latar belakang bersih sebagai hasil dari proses *remove background*. Pada pengujian akhir, sistem suara juga dievaluasi dan hasilnya menunjukkan bahwa keluaran audio untuk setiap kelas label selalu sinkron dengan teks yang dideteksi oleh sistem.

IV. KESIMPULAN

Hasil penelitian ini menunjukkan bahwa bahasa isyarat, khususnya SIBI, dapat diidentifikasi secara waktu nyata menggunakan model pelatihan berbasis YOLO. Hasil penelitian juga menunjukkan bahwa kualitas data, kuantitas data, serta nilai *epoch* berpengaruh signifikan terhadap kualitas model yang dihasilkan. Pelatihan model YOLO dalam penelitian ini mencapai akurasi terbaik pada proses *fine-tuning* skenario keempat, dengan jumlah data utama sebanyak 40 citra untuk setiap kelas label, serta total 60 citra pada subset pelatihan setelah augmentasi untuk setiap kelas label. Model yang dilatih pada *epoch* ke-24 menghasilkan nilai akurasi pelatihan sebesar 99,9% berdasarkan perhitungan nilai evaluasi *confusion matrix*. Berdasarkan hasil penelitian, sistem identifikasi bahasa isyarat yang dikembangkan mampu mengenali isyarat melalui masukan citra yang diambil secara langsung menggunakan kamera web dan diproses dalam waktu singkat. Dengan demikian, tujuan pengembangan model YOLO untuk mengidentifikasi bahasa isyarat telah tercapai.

Namun, model masih memerlukan penyempurnaan lebih lanjut untuk mengatasi keterbatasan yang ditemukan dalam penelitian ini sebelum dapat diterapkan secara luas di lingkungan nyata. Penelitian ini diharapkan dapat menjadi dasar bagi pengembangan penelitian selanjutnya. Penelitian mendatang direkomendasikan memperkaya variasi latar belakang dalam *dataset* serta menyesuaikan sistem dengan domain aplikasi yang dituju. Penerapan variasi augmentasi yang lebih beragam berpotensi mengatasi permasalahan kesalahan deteksi akibat kemiripan isyarat antarlabel, serta melengkapi *dataset* alfabet dengan memasukkan huruf J yang direpresentasikan dalam bentuk gerakan melalui pendekatan pengambilan data berbasis video dan pelatihan model. Untuk meningkatkan keterterapan yang lebih luas, implementasi sistem dalam bentuk yang lebih mudah diakses, seperti aplikasi seluler atau platform sejenis, menjadi penting agar model dapat benar-benar digunakan dalam lingkungan nyata.

KONFLIK KEPENTINGAN

Penulis menyatakan bahwa artikel berjudul "Penerapan Metode *You Only Look Once* (YOLO) untuk Identifikasi Bahasa Isyarat" ditulis tanpa adanya konflik kepentingan.

KONTRIBUSI PENULIS

Konseptualisasi, Reni Triyaningsih, Pradita Eko Prasetyo Utomo, dan Benedika Ferdian Hutabarat; metodologi, Reni Triyaningsih, Pradita Eko Prasetyo Utomo, dan Benedika Ferdian Hutabarat; perangkat lunak, Reni Triyaningsih; validasi, Pradita Eko Prasetyo Utomo dan Benedika Ferdian Hutabarat; analisis formal, Pradita Eko Prasetyo Utomo; investigasi, Reni Triyaningsih dan Pradita Eko Prasetyo Utomo; sumber daya, Pradita Eko Prasetyo Utomo dan Benedika Ferdian Hutabarat; kurasi data, Reni Triyaningsih; penulisan—persiapan draf awal, Reni Triyaningsih; penulisan—penelaahan dan penyuntingan, Pradita Eko Prasetyo Utomo dan Benedika Ferdian Hutabarat; visualisasi, Reni Triyaningsih; supervisi, Pradita Eko Prasetyo Utomo dan Benedika Ferdian Hutabarat; administrasi proyek, Pradita Eko Prasetyo Utomo dan Benedika Ferdian Hutabarat; perolehan pendanaan, Reni Triyaningsih.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Fakultas Sains dan Teknologi, Universitas Jambi serta Sekolah Luar Biasa Negeri Prof. Dr. Sri Soedewi Masjchun Sofwan, S.H., Jambi atas dukungan dan bantuan yang sangat berharga dalam keberhasilan penelitian ini. Penulis sangat menghargai kerja sama dan komitmen yang telah diberikan dalam mendukung pelaksanaan penelitian ini.

REFERENSI

[1] N.P.L. Wedayanti, A.P. Lintangsari, dan G.A.P. Wirawan, "Perkembangan bahasa isyarat daerah Denpasar," *Linguist. Indones.*, vol. 39, no. 2, hal. 217–223, Agu. 2021, doi: 10.26499/li.v39i2.230.

[2] D. Bragg dkk., "Sign language recognition, generation, and translation: An interdisciplinary perspective," dalam *ASSETS '19, Proc. 21st Int. ACM SIGACCESS Conf. Comput. Access.*, 2019, hal. 16–31, doi: 10.1145/3308561.3353774.

[3] M. Sholawati, K. Auliasari, dan Fx. Ariwibisono, "Pengembangan aplikasi pengenalan bahasa isyarat abjad SIBI menggunakan metode convolutional neural network (CNN)," *JATI*, vol. 6, no. 1, hal. 134–144, Feb. 2022, doi: 10.36040/jati.v6i1.4507.

[4] A. Pratiwi, "Penggunaan sistem isyarat bahasa Indonesia (SIBI) sebagai media komunikasi (Studi pada siswa tunarungu di SLB Yayasan Bukesra Ulee Kareng, Banda Aceh)," *J. Ilm. Mhs. FISIP (JIMFISIP)*, vol. 4, no. 3, hal. 1–12, Agu. 2019.

[5] I.J. Thira dkk., "Pengenalan alfabet sistem isyarat bahasa Indonesia (SIBI) menggunakan convolutional neural network," *J. Algoritma*, vol. 20, no. 2, hal. 421–432, Okt. 2023, doi: 10.33364/algoritma/v.20-2.1480.

[6] R.R.D. Jannah, "Pola komunikasi guru dalam meningkatkan kemampuan belajar siswa tunarungu di sekolah luar biasa negeri Lubuk Linggau," *Wardah*, vol. 22, no. 2, hal. 1–15, Des. 2021, doi: 10.19109/wardah.v22i2.10830.

[7] E. Juherna, E. Purwanti, Melawati, dan Y.S. Utami, "Implementasi pendidikan karakter pada disabilitas anak tunarungu," *J. Gold. Age*, vol. 4, no. 1, hal. 12–19, Jun. 2020, doi: 10.29408/jga.v4i01.1809.

[8] I. Damayanti dan S.H. Purnamasari, "Hambatan komunikasi dan stres orangtua siswa tunarungu sekolah dasar," *J. Psikol. Insight*, vol. 3, no. 1, hal. 1–9, Apr. 2019, doi: 10.17509/insight.v3i1.22311.

[9] E. Mustapić dan F. Malenica, "The signs of silence – An overview of systems of sign languages and co-speech gestures," *ELOPE, Engl. Lang. Overseas Perspect. Enq.*, vol. 16, no. 1, hal. 123–144, Jun. 2019, doi: 10.4312/elope.16.1.123-144.

[10] Renaldy dan A.B. Dharmawan, "Pengenalan citra bahasa isyarat berdasarkan sistem isyarat bahasa Indonesia menggunakan metode vision transformer," *JIKSI (J. Ilmu Komput, Sist. Inf.)*, vol. 12, no. 2, hal. 1–9, Jul. 2024, doi: 10.24912/jiksi.v12i2.31559.

[11] A. Jinan dan B.H. Hayadi, "Klasifikasi penyakit tanaman padi mengunakan metode convolutional neural network melalui citra daun (Multilayer perceptron)," *J. Comput. Eng. Sci.*, vol. 1, no. 2, hal. 37–44, Apr. 2022.

[12] Y. Hartiwi, E. Rasywir, Y. Pratama, dan P.A. Jusia, "Sistem manajemen absensi dengan fitur pengenalan wajah dan GPS menggunakan YOLO pada platform Android," *J. Media Inform. Budidarma*, vol. 4, no. 4, hal. 1235–1242, Okt. 2020, doi: 10.30865/mib.v4i4.2522.

[13] D.N. Alfarizi dkk., "Penggunaan metode YOLO pada deteksi objek: Sebuah tinjauan literatur sistematis," *J. Artif. Intel. Sist. Penunjang Keputusan*, vol. 1, no. 1, hal. 55–63, Jun. 2023.

[14] J.S.W. Hutauruk, T. Matulatan, dan N. Hayaty, "Deteksi kendaraan secara real time menggunakan metode YOLO berbasis Android," *J. Sustain., J. Has. Penelit. Ind. Terap.*, vol. 9, no. 1, hal. 8–14, Mei 2020, doi: 10.31629/sustainable.v9i1.1401.

[15] J. Redmon, S. Divvala, R. Girshick, dan A. Farhadi, "You only look once: Unified, real-time object detection," dalam *2016 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, hal. 779–788, doi: 10.1109/CVPR.2016.91.

[16] D. Pestana dkk., "A full featured configurable accelerator for object detection with YOLO," *IEEE Access*, vol. 9, hal. 75864–75877, Mei 2021, doi: 10.1109/ACCESS.2021.3081818.

[17] A. Riansyah dan A.H. Mirza, "Pendeteksi mobil berdasarkan merek dan tipe menggunakan algoritma YOLO," *SMATIKA, STIKI Inform. J.*, vol. 13, no. 01, hal. 43–52, Jun. 2023, doi: 10.32664/smatika.v13i01.719.

[18] A. Gallu, A.R. Himamunanto, dan H. Budiati, "Pengenalan emosi pada citra wajah menggunakan metode YOLO," *KESATRIA, J. Penerapan Sist. Inf. (Komput. Manaj.)*, vol. 5, no. 3, hal. 1253–1261, Jul. 2024, doi: 10.30645/kesatria.v5i3.444.

[19] G.A. Sidik, "Deteksi tindak kekerasan dan perundungan pada anak berbasis YOLOV8 (You Only Look Once)," *Kohesi, J. Multidisiplin Saintek*, vol. 3, no. 9, hal. 71–80, Jun. 2024, doi: 10.3785/kohesi.v3i9.4044.

[20] B.K. Pratama, S. Lestanti, dan Y. Primasari, "Implementasi algoritma you only look once (YOLO) untuk mendeteksi bahasa isyarat SIBI," *J. ProTekInfo*, vol. 11, no. 2, hal. 7–14, Agu. 2024, doi: 10.30656/protekinfo.v11i2.9105.

[21] L. Mahdiyah, S. Oktamuliani, dan W.L. Putri, "Penerapan algoritma deep learning YOLOv8 pada platform Roboflow untuk segmentasi citra panoramik," *J. Fis. Unand (JFU)*, vol. 14, no. 3, hal. 228–234, Mei 2025, doi: 10.25077/jfu.14.3.228-234.2025.

[22] M. Heydarian, T.E. Doyle, dan R. Samavi, "MLCM: Multi-label confusion matrix," *IEEE Access*, vol. 10, hal. 19083–19095, Feb. 2022, doi: 10.1109/ACCESS.2022.3151048.

[23] M. Grandini, E. Bagli, dan G. Visani, "Metrics for multi-class classification: An overview," 2020, *arXiv:2008.05756*.

[24] D. Chicco dan G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genom.*, vol. 21, hal. 1–13, Jan. 2020, doi: 10.1186/s12864-019-6413-7.

[25] S.V.N. Afni, E.P. Silmina, dan I.B. Pangestu, "Computer vision used to monitor the youth during the pandemic COVID-19," dalam *Procedia Eng. Life Sci.*, 2021, hal. 1–4, doi: 10.21070/pels.v1i2.1019.

[26] T.A. Dompeipen, S.R.U.A. Sompie, dan M.E.I. Najoan, "Computer vision implementation for detection and counting the number of humans," *J. Tek. Inform.*, vol. 16, no. 1, hal. 65–76, Mar. 2021, doi: 10.35793/jti.v16i1.31471.

[27] L. Liu dkk., "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, hal. 261–318, Feb. 2020, doi: 10.1007/s11263-019-01247-4.

[28] H. Adusumalli dkk., "Face mask detection using OpenCV," dalam *2021 3rd Int. Conf. Intell. Commun. Technol. Virtual Mob. Netw. (ICICV)*, 2021, hal. 1304–1309, doi: 10.1109/ICICV50876.2021.9388375.

[29] O.P. Orochi dan L.G.Kabari, "Text-to-speech recognition using Google API," *Int. J. Comput. Appl.*, vol. 183, no. 15, hal. 18–20, Jul. 2021, doi: 10.5120/ijca2021921474.

[30] "Pygame," Pygame. Tanggal akses: 25-Sep-2025. [Online]. Tersedia: https://www.pygame.org