Jurnal Teknik Informatika (JUTIF)
Vol. 6, No. 1, February 2025, pp. 259-268

DOI: https://doi.org/10.52436/1.jutif.2025.6.1.3912
p-ISSN: 2723-3863
e-ISSN: 2723-3871

LEVERAGING DEEP LEARNING APPROACH FOR ACCURATE ALPHABET RECOGNITION THROUGH HAND GESTURES IN SIGN LANGUAGE

Nadiyan Syah Wahyu Nugroho[*1], Muhammad Pajar Kharisma Putra[2]

[1,2]Informatics, Faculty of Engineering and Computer Science, Universitas Teknokrat Indonesia, Indonesia
Email: [1]nadiansyah_wahyu_nugroho@teknokrat.ac.id, [2]pajarkharisma@teknokrat.ac.id

(Article received: October 08, 2024; Revision: October 25, 2024; published: February 20, 2025)

Abstract

Sign language is one way of communication used by people who cannot speak or hear (deaf and speech impaired), so not everyone can understand sign language. Therefore, to facilitate communication between normal people and deaf and speech-impaired people, many systems have been created to translate gestures and signs in sign language into understandable words. Artificial intelligence and computer vision-based technologies, such as YOLOv9 offer solutions to recognize hand gestures more quickly, accurately, and efficiently. This research aims to develop a hand gesture detection system for alphabetic sign language using YOLOv9 architecture, with the aim of improving the accuracy and speed of hand gesture detection. The data used consists of 6500 sign language alphabet hand gesture images that have been labeled with bounding boxes and processed using image augmentation techniques. The model was trained on the Kaggle platform and evaluated using performance metrics such as Accuracy, Precision, Recall, F1-Score, and Intersection over Union (IoU). The results show that the YOLOv9 model achieves an average detection accuracy of 97%, with precision and recall above 90% for most classes. In addition, YOLOv9 shows advantages over other algorithms such as SSD MobileNet v2 and Faster RCNN, both in terms of speed and accuracy. In conclusion, YOLOv9 proved to be very effective in detecting sign language hand gestures, thereby speeding up and facilitating communication. This research is expected to contribute to the development of more inclusive technologies in various fields, such as education, public services, and employment opportunities, which support better communication between sign language users and the general public.

Keywords: *Alphabet, Sign Language, Computer Vision, Intersection over Union (IoU), YOLO.*

MEMANFAATKAN PENDEKATAN DEEP LEARNING UNTUK PENGENALAN ALFABET YANG AKURAT MELALUI GERAKAN TANGAN DALAM BAHASA ISYARAT

Abstrak

Bahasa isyarat merupakan salah satu cara komunikasi yang digunakan oleh orang-orang yang tidak dapat berbicara atau mendengar (tuna rungu dan tuna wicara), sehingga tidak semua orang dapat memahami bahasa isyarat. Oleh karena itu, untuk memudahkan berkomunikasi antara orang normal dengan orang tuna rungu dan tuna wicara, banyak sistem yang diciptakan untuk menerjemahkan gerakan dan tanda dalam bahasa isyarat ke dalam kata-kata yang dapat dimengerti. Teknologi berbasis kecerdasan buatan dan *comuter vision*, seperti *YOLOv9* menawarkan solusi untuk mengenali gerakan tangan dengan lebih cepat, akurat, dan efisien. Penelitian ini bertujuan mengembangkan sistem deteksi gerakan tangan untuk bahasa isyarat alfabet menggunakan arsitektur *YOLOv9*, dengan tujuan untuk meningkatkan akurasi dan kecepatan deteksi gerakan tangan. Data yang digunakan terdiri dari 6500 gambar gestur tangan alfabet bahasa isyarat yang telah dilabeli dengan *bounding boxes* dan diproses menggunakan teknik augmentasi gambar. Model dilatih di platform *kaggle* dan dievaluasi menggunakan metrik performa seperti *Accuracy, Precision, Recall, F1-Score*, serta *Intersection over Union (IoU).* Hasil penelitian menunjukkan bahwa model *YOLOv9* mencapai akurasi deteksi rata-rata sebesar 97%, dengan *precision* dan *recall* di atas 90% untuk sebagian besar kelas. Selain itu, *YOLOv9* menunjukkan keunggulan dibandingkan algoritma lain seperti *SSD MobileNet v2* dan *Faster RCNN*, baik dalam hal kecepatan maupun akurasi. Kesimpulannya, *YOLOv9* terbukti sangat efektif dalam mendeteksi gerakan tangan bahasa isyarat, sehingga dapat mempercepat dan mempermudah komunikasi. Penelitian ini diharapkan berkontribusi pada pengembangan teknologi yang lebih inklusif di berbagai bidang, seperti pendidikan, layanan publik, dan kesempatan kerja, yang mendukung komunikasi yang lebih baik antara pengguna bahasa isyarat dan masyarakat umum.

Kata kunci: *Alfabet, Bahasa Isyarat, Computer Vision, Intersection over Union (IoU), YOLO.*

1. PENDAHULUAN

Bahasa isyarat merupakan alat komunikasi bagi individu dengan gangguan pendengaran atau kesulitan berkomunikasi secara verbal[1]. Dengan menggunakan gerakan tangan, bahasa isyarat memungkinkan pengguna untuk menyampaikan ide, pikiran, dan informasi kepada orang lain. Di antara berbagai jenis bahasa isyarat, alfabet isyarat memiliki peran penting, terutama untuk mengeja kata-kata atau nama-nama yang tidak memiliki gerakan khusus[2]. Misalnya, saat seseorang harus menyebutkan nama tempat, nama orang, atau istilah teknis yang jarang digunakan, gerakan tangan alfabet menjadi satu-satunya alat yang dapat diandalkan untuk menyampaikan informasi tersebut secara akurat[3].

Meskipun memiliki peran yang sangat penting, penggunaan bahasa isyarat di masyarakat umum masih sangat terbatas. Salah satu penyebab utama masalah ini adalah kurangnya pendidikan bahasa isyarat di sekolah-sekolah umum dan tempat kerja, sehingga banyak orang yang tidak memiliki kemampuan untuk berkomunikasi dengan pengguna bahasa isyarat[4]. Akibatnya, terdapat kesenjangan komunikasi antara pengguna bahasa isyarat dan masyarakat umum, yang dapat menghalangi akses terhadap layanan penting seperti kesehatan, pendidikan dan kesempatan kerja[5].

Untuk mengatasi masalah ini, berbagai teknologi modern mulai digunakan, khususnya yang berbasis kecerdasan buatan dan *computer vision*[6]. Teknologi-teknologi ini memberikan solusi untuk mengurangi kesenjangan berkomunikasi antara pengguna bahasa isyarat dan masyarakat umum. Salah satu pendekatan yang menonjol adalah penggunaan algoritma deteksi objek yang dapat mengenali gerakan tangan dan menerjemahkannya ke dalam teks yang mudah dipahami oleh masyarakat umum[7]. Dalam konteks ini, *YOLO* (*You Only Look Once*) telah menjadi salah satu algoritma yang paling populer untuk mendeteksi objek dalam gambar atau video[8]. *YOLO* dirancang untuk mendeteksi objek dengan akurat hanya dalam satu kali pemrosesan gambar, menjadikannya salah satu pilihan terbaik untuk aplikasi yang membutuhkan pengenalan visual cepat seperti deteksi bahasa isyarat[9].

Meskipun penerapan *YOLO* pada bahasa isyarat, khususnya alfabet, masih tergolong baru. Beberapa penelitian sebelumnya telah menggunakan versi *YOLO* untuk mengenali isyarat tangan bahasa isyarat, tetapi sering kali berfokus pada pengenalan kata atau frasa yang sudah memiliki isyarat yang jelas[10]. Penelitian-penelitian ini biasanya tidak mencakup pengejaan alfabet secara khusus, yang merupakan komponen penting dalam komunikasi bahasa isyarat, terutama dalam konteks pengejaan nama atau istilah teknis[11].

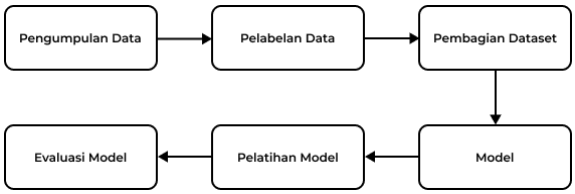
Algoritma *YOLOv9* memberikan peningkatan dalam hal akurasi dan kecepatan dalam pendeteksian objek, termasuk deteksi gerak tangan. Hal ini penting dalam aplikasi untuk bahasa isyarat, di mana kecepatan dan akurasi merupakan faktor penting yang menentukan efektivitas sistem pengenalan isyarat[12]. Dibandingkan dengan algoritma lain seperti *SSD MobileNet v2* dan *Faster R-CNN*, *YOLOv9* terbukti lebih unggul dalam hal kecepatan inferensi tanpa mempengaruhi akurasi pendeteksian objek[13][14].

Kebutuhan untuk mengembangkan sistem deteksi isyarat tangan yang efisien tidak hanya akan meningkatkan aksesibilitas bagi para penyandang disabilitas, tetapi juga berkontribusi terhadap kesadaran masyarakat umum akan pentingnya komunikasi yang inklusif[15]. Dengan teknologi seperti *YOLOv9*, pengguna bahasa isyarat dapat berinteraksi dengan masyarakat umum tanpa harus menghadapi berbagai kendala. Selain itu, teknologi ini juga berpotensi untuk meningkatkan pemahaman dan kepedulian masyarakat umum terhadap kebutuhan komunikasi pengguna bahasa isyarat.

Penelitian ini bertujuan untuk mengembangkan sistem deteksi gerakan tangan untuk alfabet bahasa isyarat menggunakan arsitektur *YOLOv9*[16]. Dengan sistem ini, diharapkan bahwa pengenalan gerakan tangan alfabet dapat dilakukan secara lebih cepat dan akurat, sehingga mengatasi hambatan komunikasi yang selama ini dihadapi oleh pengguna bahasa isyarat dalam berbagai situasi formal dan informal[17]. Selain itu, sistem ini juga diharapkan dapat mendukung pengembangan teknologi yang lebih inklusif di berbagai bidang, seperti pendidikan, layanan publik, dan kesempatan kerja, yang semuanya berkontribusi terhadap peningkatan aksesibilitas dan partisipasi penuh bagi penyandang disabilitas dalam kehidupan masyarakat umum[18].

2. METODE PENELITIAN

Penelitian ini menggunakan model *YOLOv9* sebagai deteksi gestur gerak tangan, yang terkenal karena keefektifannya dalam mendeteksi objek[16]. Data yang digunakan dalam penelitian ini terdiri dari beragam koleksi gambar gerakan tangan alfabet yang dikumpulkan melalui platform *roboflow*. Dalam proses pelatihan model, data kemudian dianotasikan dengan menggunakan teknik augmentasi gambar yang bertujuan untuk meningkatkan akurasi dan kinerja model secara signifikan. Pelatihan model *YOLOv9* dilakukan dengan menggunakan lingkungan berbasis *cloud* pada platform *kaggle*. Pengujian ini menggunakan satu set data uji yang sebelumnya tidak ditemui oleh model, dengan tujuan utama menghitung metrik kinerja, termasuk *Accuracy, Precission, Recall, F-1 Score* dan *Intersection over Union (IoU).* Hasil evaluasi dianalisis untuk memberikan penilaian tentang efisiensi model dalam mendeteksi gerakan tangan secara akurat. Berikut adalah tahapan-tahapan penelitian yang ditampilkan pada Gambar 1.



Gambar 1. Tahapan Penelitian

2.1. Pengumpulan Data

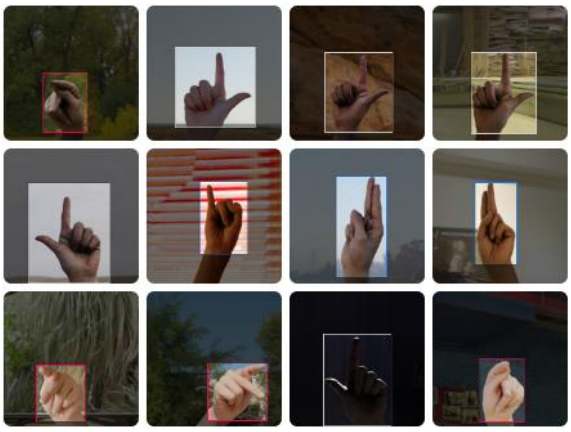
Dalam penelitian ini, dataset yang digunakan terdiri dari 6500 gambar yang mewakili gestur tangan untuk bahasa isyarat alfabet. Dataset tersebut dibagi ke dalam 26 kelas, di mana setiap kelas mewakili satu huruf dalam alfabet. Setiap kelas tidak hanya berisi satu variasi gambar, melainkan mencakup berbagai variasi gestur tangan untuk setiap huruf. Variasi yang dimaksud meliputi beberapa faktor, seperti perbedaan kondisi pencahayaan, sudut pandang kamera yang beragam, serta latar belakang yang berbeda-beda. Tujuan dari variasi ini adalah untuk menciptakan model pengenalan gestur yang mampu bekerja dengan baik dalam kondisi dunia nyata yang dinamis, di mana pencahayaan, sudut kamera, dan latar belakang dapat sangat bervariasi.

Dengan memiliki dataset yang lebih beragam, model yang dikembangkan diharapkan dapat memiliki kemampuan generalisasi yang tinggi[19]. Artinya, model tersebut akan mampu mengenali dan mendeteksi gestur tangan dengan akurasi yang baik, meskipun terdapat variasi dalam kondisi pengambilan gambar. Hal ini penting untuk memastikan bahwa sistem pengenalan gestur tangan yang dikembangkan dapat bekerja secara efektif di berbagai situasi nyata dan tidak hanya terbatas pada kondisi yang sudah dikenal. Berikut beberapa contoh dataset yang ditampilkan pada Gambar 2.

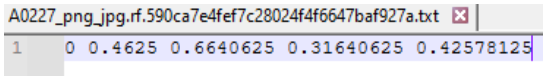

Gambar 2. Dataset Gestur Bahasa Isyarat Alfabet

2.2. Pelabelan Data

Setelah pengumpulan data, langkah selanjutnya adalah dengan melakukan anotasi atau pelabelan pada setiap gambar. Proses ini melibatkan penandaan area tertentu dalam gambar yang berisi gestur tangan, menggunakan *bounding box* yang mengelilingi objek tersebut secara tepat. Setiap *bounding box* diberi label yang sesuai dengan huruf alfabet yang diwakili oleh gestur tangan tersebut, dengan tujuan agar model dapat mengenali dan memahami pola-pola visual yang membentuk setiap gestur. Proses anotasi harus dilakukan dengan sangat teliti. Hasil pelabelan ditampilkan pada Gambar 3.


Gambar 3. Hasil Labeling

Setiap gambar yang sudah dilabeling akan mendapatkan file *.txt* berisikan informasi koordinat *bounding box* dan label objek dalam format yang sesuai dengan *YOLOv9* yang ditampilkan pada Gambar 4. Format ini mencakup label kelas yang mengidentifikasi objek, serta koordinat posisi dan ukuran *bounding box* yang dinyatakan dalam skala relatif terhadap ukuran gambar. Hal ini memungkinkan *YOLOv9* untuk mendeteksi objek dengan akurat, bahkan pada gambar dengan berbagai resolusi, sehingga proses deteksi dapat dilakukan secara efisien dan konsisten.



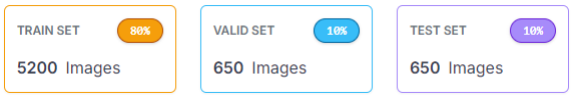
Gambar 4. Koordinat Bounding Box

2.3. Pembagian Dataset

Setelah tahap anotasi atau pemberian label pada dataset, langkah selanjutnya adalah membagi 6500 gambar tersebut menjadi tiga bagian utama yaitu *Training Set*, *Validation Set*, dan *Test Set* seperti yang ditampilkan pada Gambar 5. Pembagian ini penting untuk memastikan model pengenalan gestur tangan dapat dilatih dengan baik serta diuji secara akurat untuk mengevaluasi kinerjanya.

a) *Training Set* berjumlah 5200 gambar, atau sekitar 80% dari total dataset. Bagian ini digunakan untuk melatih model agar dapat memahami pola dalam dataset dan mengenali masing-masing huruf dalam bahasa isyarat[20].
b) *Validation Set* terdiri dari 650 gambar, atau 10% dari dataset. Dataset ini digunakan untuk memvalidasi kinerja model selama proses pelatihan. Dengan menggunakan *validation set*, model dapat bekerja pada data yang belum pernah digunakan selama pelatihan, sehingga memungkinkan model untuk mengoptimalkan parameter tanpa menggunakan data uji *(test set)*.
c) *Test Set* juga terdiri dari 650 gambar, yaitu 10% dari dataset. *Set* ini digunakan untuk menguji model setelah pelatihan selesai. Berbeda dari *validation set*, *test set* digunakan untuk mengevaluasi kinerja akhir model dalam

mengenali gestur tangan pada data yang benar-benar baru, memastikan bahwa model dapat bekerja dengan baik pada situasi nyata.

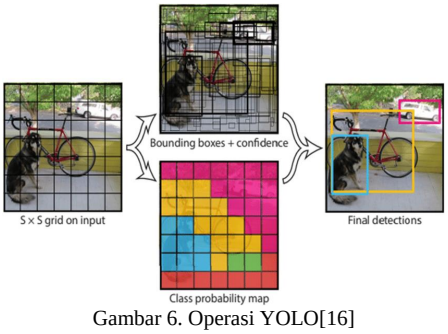


Gambar 5. Pembagian Gambar Dataset

Pembagian dataset dirancang untuk mengoptimalkan proses pelatihan, validasi, dan pengujian model secara keseluruhan, sehingga model yang dihasilkan dapat memberikan hasil yang akurat dan dapat diandalkan.

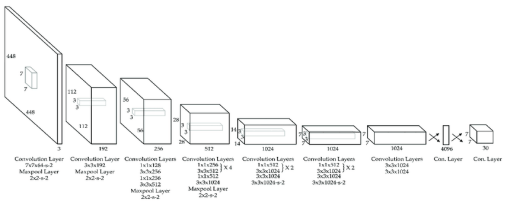
2.4. *YOLO (You Only Look Once)*

YOLO adalah salah satu algoritma deteksi objek yang paling populer dan banyak digunakan dalam *computer vision*. Algoritma ini dikenal karena kemampuannya untuk mendeteksi dan mengenali objek dalam gambar atau video dengan sangat cepat dan akurat[16]. Nama *You Only Look Once* merujuk pada pendekatan algoritma ini yang hanya memerlukan satu kali pemrosesan pada gambar untuk mendeteksi semua objek di dalamnya, berbeda dengan metode lain yang membutuhkan beberapa kali pemrosesan. Berbeda dengan metode deteksi objek tradisional yang sering menggunakan teknik *sliding window* atau *region proposals*, yang memerlukan beberapa langkah dan sering kali lambat, *YOLO* dapat melakukan semua prediksi sekaligus. Hal ini memungkinkan *YOLO* untuk bekerja sangat cepat, bahkan mampu memproses video secara *real-time*.



Gambar 6. Operasi YOLO[16]

Dalam mendeteksi objek pada sebuah gambar, dimulai dengan membagi gambar input menjadi grid berukuran S×S seperti yang ditampilkan pada Gambar 6. Setiap sel dalam grid bertanggung jawab untuk mendeteksi objek jika bagian tengah objek tersebut jatuh di dalam sel tersebut. Setiap sel menghasilkan beberapa kotak batas *(bounding boxes)* dan nilai keyakinan *(confidence)*. Nilai *confidence* ini juga dipertimbangkan untuk klasifikasi objek, seperti sepeda atau anjing. *Bounding box* yang memiliki *confidence* tertinggi (setelah *non-max suppression*) ditampilkan pada gambar akhir. Warna-warna dalam *class probability map* menunjukkan probabilitas berbagai kelas objek di dalam grid tersebut.



Gambar 7. Arsitektur YOLO[16]

Pada Gambar 7, *YOLO* memliliki arsitektur dengan mengubah seluruh gambar input menjadi grid berukuran 7x7, di mana setiap sel grid bertanggung jawab untuk mendeteksi objek. Gambar input berukuran 448x448x3 diproses melalui serangkaian lapisan konvolusi dan *pooling*, dimulai dengan lapisan konvolusi 7x7x64 dan dilanjutkan dengan lapisan-lapisan konvolusi lebih kecil (3x3x192, 1x1x128, 3x3x256, dan seterusnya) yang secara bertahap mengurangi ukuran spasial gambar sambil mengekstraksi fitur. *Max pooling* 2x2 dengan *stride* 2 digunakan untuk mengurangi ukuran gambar secara berkala. Setelah beberapa lapisan konvolusi, jaringan menghasilkan *output* fitur berukuran 7x7x1024 yang diratakan menjadi vektor 4096. Pada tahap akhir, jaringan menghasilkan *tensor* 7x7x30 yang mewakili prediksi *bounding box*, *confidence score*, dan probabilitas kelas untuk setiap sel grid. Arsitektur ini dirancang untuk efisiensi dan kecepatan, memungkinkan *YOLO* untuk melakukan deteksi objek secara langsung.

2.5. Pelatihan Model

Proses pelatihan model *YOLOv9* dilakukan dengan menggunakan *kaggle* sebagai platform utama karena ketersediaan *GPU* yang dapat diakses secara gratis. Penggunaan *GPU* dapat memungkinkan peningkatan efisiensi dan kecepatan dalam pelatihan pelatihan, dibandingkan dengan penggunaan *CPU* biasa. Langkah ini tidak hanya mempercepat proses pelatihan tetapi juga memungkinkan eksperimen dengan berbagai parameter dan arsitektur model dalam waktu yang lebih singkat, meningkatkan produktivitas serta memungkinkan pengembangan model deteksi yang lebih akurat. Setelah itu, dataset gestur tangan diunggah dan dipreprocess untuk memastikan kualitas data. Selama fase pelatihan, parameter seperti *epoch*, *batch size*, dan *image size* disesuaikan untuk optimalisasi yang ditampilkan pada Tabel 1.

Tabel 1. Parameter Pelatihan

No	Parameter	Nilai
1	*Epoch*	500
2	*Batch Size*	128
3	*Image Size*	320

Pada Tabel Parameter Pelatihan, model dilatih menggunakan 500 *epoch*. Setiap *epoch* mewakili satu siklus penuh di mana model memproses seluruh dataset. Tujuan penggunaan *epoch* sebanyak ini adalah untuk memberikan kesempatan kepada model memperbaiki parameter secara bertahap berdasarkan

kesalahan *(error)* yang dihasilkan setelah memproses dataset. Semakin banyak *epoch*, semakin lama model belajar dari data, sehingga dapat menemukan pola yang lebih baik dan mendetail. Penggunaan 500 *epoch* diharapkan dapat meningkatkan performa model secara bertahap, mengurangi kesalahan, dan memungkinkan model untuk menyesuaikan bobot-bobotnya sehingga lebih optimal dalam mengenali pola gestur yang kompleks.

Selanjutnya, ukuran *batch* diatur ke 128. Artinya, pada setiap iterasi, model akan memproses 128 gambar secara bersamaan sebelum melakukan *backpropagation*, yaitu proses memperbarui bobot model berdasarkan kesalahan yang dihitung dari hasil prediksi. Menggunakan ukuran *batch* yang lebih besar memungkinkan pelatihan yang lebih cepat karena lebih banyak data yang diproses dalam satu waktu. Selain itu, ukuran *batch* yang besar membuat model menjadi lebih stabil karena gradien rata-rata yang dihitung lebih akurat. *GPU* juga dapat digunakan secara lebih efisien dengan ukuran *batch* yang besar, karena *GPU* didesain untuk melakukan komputasi paralel yang optimal saat menangani data dalam jumlah besar sekaligus. Hal ini mengurangi waktu pelatihan secara keseluruhan, yang sangat penting untuk model dengan jumlah *epoch* yang besar seperti dalam penelitian ini.

Ukuran gambar yang digunakan dalam pelatihan adalah 320x320 piksel. Pemilihan ukuran gambar ini bertujuan untuk mempercepat komputasi selama proses pelatihan. Gambar yang lebih kecil memerlukan daya komputasi yang lebih sedikit, sehingga memungkinkan model melakukan lebih banyak iterasi dalam waktu yang sama, meningkatkan kecepatan pelatihan. Selain itu, dalam aplikasi *real-time*, seperti deteksi gestur langsung, ukuran gambar yang lebih kecil memungkinkan sistem untuk bekerja lebih cepat tanpa mengorbankan akurasi secara signifikan. Ukuran gambar 320x320 sering dianggap sebagai keseimbangan ideal antara kecepatan dan ketepatan deteksi, karena meskipun gambar diperkecil, informasi penting yang relevan untuk pengenalan gestur tetap dapat dipertahankan.

2.6. Evaluasi Model

Evaluasi model *YOLOv9* adalah langkah penting untuk memastikan bahwa model yang telah dilatih berfungsi dengan baik dalam mendeteksi gestur tangan. Proses evaluasi dimulai dengan menguji model pada dataset validasi yang belum pernah dilihat sebelumnya untuk mengukur performa deteksi dalam situasi nyata. Beberapa metrik utama digunakan untuk mengevaluasi performa model:

a. *Accuracy* adalah metrik evaluasi yang menggambarkan proporsi prediksi yang benar terhadap total prediksi yang dibuat. Dalam penelitian, akurasi digunakan sebagai salah satu indikator kinerja dasar dalam mengukur performa model klasifikasi[21].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

b. *Precision* adalah metrik yang menilai seberapa akurat model dalam membuat prediksi positif. *Precision* dihitung sebagai rasio antara jumlah prediksi positif yang benar *(True Positives)* dan total jumlah prediksi positif yang dibuat oleh model, yang terdiri dari *True Positives* dan *False Positives*[22].

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

c. *Recall* mengukur seberapa baik model dalam mendeteksi semua *instance* positif yang ada dalam dataset. *Recall* dihitung sebagai rasio antara jumlah prediksi positif yang benar *(True Positives)* dan total jumlah *instance* positif yang ada, yang terdiri dari *True Positives* dan *False Negatives*[23].

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

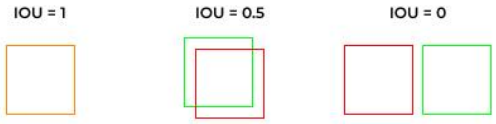
d. *F1-Score* memberikan keseimbangan antara tingkat prediksi benar *(Precision)* dan cakupan deteksi positif *(Recall)*.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

e. *Intersection over Union (IoU)* dihitung dengan membagi area tumpang tindih antara prediksi *(predicted bounding box)* dan kebenaran *(ground truth bounding box)* dengan area gabungan keduanya[24].

$$IoU = \frac{A \cap B}{A \cup B} \quad (5)$$

Pada Gambar 8 menampilkan rentang nilai *IoU* yang berkisar antara 1 hingga 0, di mana 1 menunjukkan prediksi yang sempurna (tumpang tindih penuh dengan *ground truth*), dan 0 berarti tidak ada tumpang tindih sama sekali. *Threshold* yang sering digunakan untuk menentukan prediksi benar biasanya adalah 0,5 atau 0,8. Semakin besar nilai *IoU*, semakin baik kesesuaian antara prediksi dan kebenaran yang menunjukkan akurasi deteksi yang lebih tinggi.



Gambar 8. Rentang Nilai IoU

3. HASIL DAN PEMBAHASAN

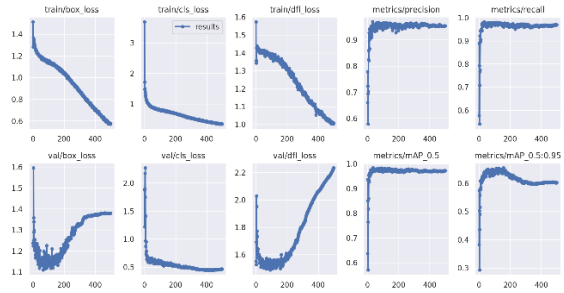
3.1. Hasil Pelatihan Model

Hasil implementasi model *YOLOv9* dalam mendeteksi gestur tangan, didapatkan performa yang cukup baik dalam hal keakuratan deteksi. Model mampu mengenali gerakan tangan dengan tingkat

akurasi tinggi, di mana nilai *precision* dan *recall* rata-rata dari keseluruhan nilai mendapatkan 0.9 ke atas. Model dilatih untuk mendeteksi 26 kelas gestur tangan yang mewakili setiap huruf alfabet yang ditunjukkan pada tabel dibawah ini.

Tabel 2. Hasil Training

Class	Precission	Recall	mAP50
A	0.962	0.976	0.965
B	0.935	0.925	0.931
C	0.986	0.971	0.978
D	0.982	1	0.995
E	0.977	1	0.995
F	0.983	1	0.995
G	0.948	0.935	0.93
H	0.965	0.974	0.976
I	0.902	0.941	0.91
J	0.985	1	0.995
K	0.966	0.978	0.978
L	0.977	1	0.995
M	0.888	0.9	0.934
N	0.939	0.963	0.965
O	0.951	1	0.989
P	0.952	1	0.98
Q	0.977	0.971	0.984
R	0.967	0.979	0.984
S	0.639	0.971	0.89
T	0.758	0.944	0.909
U	0.972	1	0.955
V	0.979	1	0.955
W	0.959	1	0.982
X	0.978	1	0.995
Y	0.954	0.97	0.953
Z	0.943	1	0.989
All	**0.939**	**0.977**	**0.969**



Gambar 9. Training dan Validation loss plots

Pada Tabel 2, hasil dari pelatihan model *YOLOv9* dalam mendeteksi gestur tangan alfabet menunjukkan kinerja yang sangat baik dengan rata-rata *precision* dan *recall* di atas 90% untuk sebagian besar kelas. Secara khusus, kelas C, J, dan F menunjukkan nilai *precision* tertinggi, yaitu di atas 98%. Nilai *precision* yang tinggi ini menunjukkan bahwa model sangat jarang salah dalam memprediksi gestur-gestur tersebut. Hal ini dapat disebabkan oleh bentuk visual yang cukup unik dan konsisten.

Sebaliknya, kelas S menunjukkan *precision* terendah sebesar 63,9%. Hal ini menandakan bahwa model sering salah memprediksi gestur S sebagai gestur lain. Rendahnya *precision* untuk kelas ini bisa disebabkan oleh kemiripan bentuk visual dengan gestur lain, misalnya E atau A, terutama pada gambar dengan kondisi pencahayaan yang buruk atau sudut pandang yang tidak ideal. Ini menunjukkan bahwa, meskipun *YOLOv9* sangat akurat pada kebanyakan gestur, ada beberapa gestur yang masih memerlukan optimasi lebih lanjut dalam pelatihan.

Recall untuk beberapa kelas seperti D, E, F, J, dan L mencapai nilai sempurna, yaitu 1. Ini berarti model mampu mendeteksi semua *instance* gestur tersebut dalam dataset uji. Sementara itu, *recall* terendah berada pada kelas M dan T, yang menunjukkan bahwa beberapa *instance* dari gestur ini tidak terdeteksi oleh model. *Recall* yang rendah menunjukkan meskipun model dapat mengenali sebagian besar gestur dengan baik, ada beberapa kondisi tertentu di mana gestur gagal dideteksi.

Dalam hal *mAP50 (mean Average Precision at 50% IoU)*, kelas seperti D, E, F, J, dan X menunjukkan performa tertinggi dengan nilai *mAP50* sebesar 99,5%. Ini mengindikasikan bahwa prediksi *bounding box* yang dihasilkan oleh model hampir sempurna dalam hal kesesuaian lokasi dan ukuran gestur. Namun, kelas S kembali menunjukkan performa yang rendah dengan *mAP50* sebesar 89%. Ini mengisyaratkan bahwa kesulitan utama model adalah dalam menentukan posisi dan ukuran yang tepat dari gestur S, yang dapat disebabkan oleh karakteristik visual gestur yang tidak terlalu berbeda dengan gestur lain.

Gambar 9 menampilkan kurva *Training Loss* dan *Validation Loss*, di mana terlihat bahwa model mengalami penurunan *loss* yang stabil selama pelatihan, yang menunjukkan bahwa model mampu belajar dengan baik tanpa gejala *overfitting* yang signifikan. Pada *epoch* ke 500, *loss* telah mencapai titik yang relatif stabil, yang menunjukkan bahwa model telah mencapai *konvergensi*..

3.2. Evaluasi Matrix

Selanjutnya mengevaluasi model dengan membandingkan beberapa algoritma yaitu, *Faster RCNN* dan *SSD MobileNet v2*. Pengujian ketiga model ini dilakukan pada dataset yang sama untuk memperoleh hasil yang lebih komprehensif dan relevan. Langkah pertama, mengimplementasikan *Faster RCNN* yang terkenal dengan akurasi tinggi. Meskipun *Faster RCNN* memberikan hasil yang baik, kecepatan prosesnya dapat menjadi kendala jika digunakan dalam penggunaan aplikasi[25]. Selanjutnya, model *SSD MobileNet v2* yang memberikan keseimbangan antara kecepatan dan akurasi[26]. Hasilnya menunjukkan bahwa *SSD* mampu mendeteksi objek dalam waktu yang lebih singkat dibandingkan *Faster RCNN*, meskipun akurasinya sedikit lebih rendah.

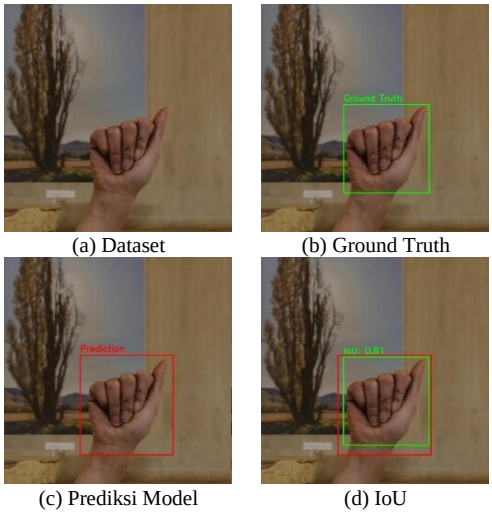
Tabel 3. Matrix Evaluasi Model

Models	Accuracy	Precision	Recall	F-1 Score
YOLO v9	097	0.99	0.98	0.98
SSD MobileNet v2	0.86	0.88	0.97	0.92
Faster RCNN	0.84	0.95	0.88	0.92

Pada Tabel 3 membandingkan kinerja ketiga model deteksi objek yaitu *YOLO v9, SSD MobileNet*

v2, dan *Faster R-CNN*, berdasarkan metrik *Accuracy*, *Precision*, *Recall*, dan *F-1 Score*. *YOLO v9* unggul dengan *accuracy* 97%, *precision* 99%, *recall* 98%, dan *F-1 score* 98%. Hal ini menunjukkan bahwa model efektif dalam mendeteksi objek. *SSD MobileNet v2* memiliki *accuracy* 86%, *precision* 88%, *recall* 97%, dan *F-1 score* 92%. Meskipun *precision* tinggi, *recall* yang lebih rendah menunjukkan bahwa model ini mungkin gagal mendeteksi beberapa objek, sehingga hasil keseluruhannya kurang baik. *Faster R-CNN* memiliki *accuracy* 84%, *precision* 95%, *recall* 88%, dan *F-1 score* 92%, yang menjadikannya model yang seimbang antara *precision* dan *recall*. Secara keseluruhan, *YOLO v9* adalah model yang paling efisien dan efektif, sedangkan *Faster R-CNN* memberikan keseimbangan antara presisi dan deteksi, sementara *SSD* berada di posisi terakhir.

Dalam memperoleh matriks, *Intersection over Union (IoU)* digunakan untuk menilai kinerja model deteksi objek. Proses dimulai dengan menentukan *bounding box* berdasarkan *ground truth*. Selanjutnya melakukan prediksi menggunakan model yang telah dilatih untuk menghasilkan *bounding box*. Setelah memperoleh dua hasil *bounding box*, selanjutnya adalah menghitung nilai *IoU*. *IoU* dihitung dengan membandingkan area irisan *(intersection)* antara kotak prediksi dan kotak *ground truth* kemudian dibagi dengan area gabungan *(union)* dari kedua kotak tersebut yang ditampilkan pada Gambar 10.



(a) Dataset (b) Ground Truth

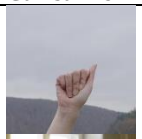
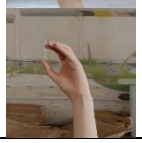
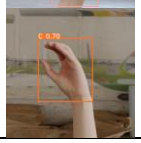
(c) Prediksi Model (d) IoU

Gambar 10. Menentukan Nilai IoU

Selanjutnya adalah dengan menentukan batas *IoU* menggunakan *threshold* dengan nilai 0.8. Nilai *threshold* ini dipilih karena memberikan keseimbangan yang baik dalam mendeteksi objek secara relevan. Dengan menetapkan batas ini, kami dapat mengkategorikan prediksi sebagai benar positif *(TP)* jika nilai *IoU* mencapai 0.8 atau lebih. Hal ini untuk memastikan bahwa model yang kami gunakan cukup akurat dalam mendeteksi objek tanpa mengorbankan kecepatan. Dalam proses evaluasi, *threshold IoU* berfungsi sebagai parameter untuk membantu mengidentifikasi seberapa baik model dalam mendeteksi. Jika nilai *IoU* di bawah 0.8, prediksi dianggap sebagai salah positif *(FP)* atau salah negatif *(FN)*, yang dapat memengaruhi metrik evaluasi lainnya seperti *precision* dan *recall*.

3.3. Pengujian Model

Tabel 4. Hasil Kinerja Model

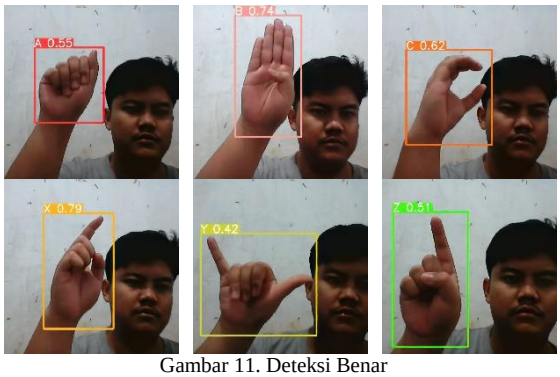
Class	Gambar Asli	Prediksi	Keterangan
A			Benar
B			Benar
C			Benar

Pada Tabel 4, menampilkan hasil pengujian kinerja model *YOLOv9* dalam mendeteksi gestur tangan yang mewakili kelas alfabet. Gambar asli di kolom kedua memperlihatkan tangan yang membentuk gestur huruf, sedangkan kolom ketiga menunjukkan hasil prediksi yang dihasilkan oleh model. Setiap kotak prediksi, terdapat skor yang menunjukkan tingkat kepercayaan model terhadap hasil deteksinya dan ditandai dengan *bounding box* yang mengelilingi area gestur, serta label kelas yang terdeteksi secara akurat oleh model selama analisisnya. Dalam gambar yang diperlihatkan, model berhasil mendeteksi gestur dengan baik, meskipun latar belakang dan kondisi pencahayaan pada gambar asli berbeda-beda. Hal ini menunjukkan kemampuan model untuk mengenali objek secara baik di berbagai kondisi. Pada kolom keterangan, *"Benar"* menunjukkan bahwa semua prediksi sesuai dengan gestur sebenarnya yang terdapat dalam gambar asli. Secara keseluruhan, hasil pada tabel memperlihatkan bahwa model *YOLOv9* mampu mendeteksi dan mengklasifikasikan gestur tangan dengan tingkat akurasi yang cukup tinggi, serta menunjukkan robustitas terhadap variasi lingkungan dan pencahayaan, sehingga hasil deteksinya tetap dapat diandalkan..

3.4. Pengujian Performa YOLO

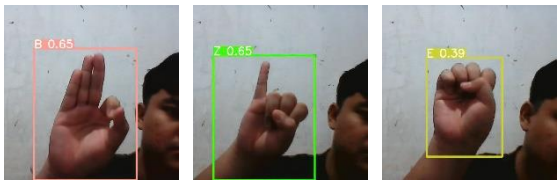
Gambar 11 menunjukkan hasil deteksi gestur tangan dengan menggunakan kamera, di mana performa model dapat dievaluasi secara lebih mendalam berdasarkan tingkat keakuratan dan ketepatan kotak prediksi yang dihasilkan oleh model tersebut. Pada gambar tersebut, model berhasil mendeteksi gestur tangan dengan akurasi tinggi, sesuai dengan jenis gerakan tangan yang telah dipelajari selama fase pelatihan. Setiap gestur yang dikenali oleh model ditampilkan dengan jelas melalui

kotak prediksi atau *bounding box* yang mengelilingi area tangan.



Gambar 11. Deteksi Benar

Penelitian ini menunjukkan bahwa *YOLOv9* memiliki performa yang baik dalam mendeteksi alfabet bahasa isyarat, namun terdapat beberapa keterbatasan yang perlu diperhatikan seperti yang ditampilkan pada Gambar 12. Pertama, model mengalami kesulitan dalam mengenali gestur yang mirip secara visual, seperti gestur S yang sering kali salah diklasifikasikan sebagai E atau A. Kesamaan bentuk dan posisi jari membuat model kesulitan membedakan gestur-gestur tersebut, terutama pada gambar dengan pencahayaan atau sudut yang kurang ideal. Untuk mengatasi hal ini, diperlukan data banyak tambahan atau teknik augmentasi yang lebih kompleks agar model dapat mengenali perbedaan halus antar gestur.



Gambar 12. Deteksi Salah

Kedua, implementasi model dalam dunia nyata masih menghadapi tantangan, khususnya pada aplikasi *real-time*. Meskipun *YOLOv9* dikenal cepat, performa deteksi dapat dipengaruhi oleh faktor-faktor seperti perubahan pencahayaan, sudut pandang kamera, atau kualitas perangkat keras yang digunakan, terutama jika komputasi dilakukan di perangkat dengan daya rendah. Ketiga, meskipun dataset yang digunakan beragam, tetapi belum mencakup seluruh variasi kondisi dunia nyata. Gerak gestur yang dilakukan dengan cepat serta perubahan pencahayaan mungkin belum cukup terwakili dalam dataset yang ada, sehingga performa model dapat menurun di situasi yang tidak terduga.

4. DISKUSI

Penelitian ini membahas deteksi alfabet dalam bahasa isyarat menggunakan gestur tangan dengan algoritma *YOLOv9*. Sistem ini dirancang untuk mempermudah komunikasi bagi individu dengan gangguan pendengaran atau kesulitan berbicara. Penelitian ini menggunakan dataset 6500 gambar gestur alfabet yang dikumpulkan melalui platform *roboflow*. Model *YOLOv9* yang dilatih dievaluasi berdasarkan akurasi, *presisi*, *recall*, *F1-score*, dan *Intersection over Union (IoU)*. Dibandingkan dengan penelitian sebelumnya yang menggunakan model *YOLOv5* dan *YOLOv8*[27], terlihat bahwa *YOLOv8* memiliki keunggulan dalam hal presisi dan kecepatan deteksi dibandingkan *YOLOv5*, dengan *mean Average Precision (mAP)* mencapai 96%. Selain itu, *YOLOv8* lebih unggul dalam klasifikasi objek dan memiliki pengurangan *loss* yang lebih cepat. Namun, kelemahan utama *YOLOv8* terlihat pada sensitivitasnya terhadap variasi latar belakang dan gambar yang kurang dikenali dengan baik.

Sementara itu, *YOLOv9* dalam penelitian ini berhasil mengatasi sebagian besar keterbatasan yang dimiliki oleh *YOLOv8*. Dengan kecepatan inferensi yang lebih tinggi dan kemampuan deteksi yang lebih andal, *YOLOv9* mencapai akurasi hingga 97%, serta *precision* dan *recall* yang konsisten di atas 90%. Peningkatan kinerja ini menunjukkan bahwa *YOLOv9* memiliki keunggulan dalam menghadapi variasi kondisi pencahayaan dan latar belakang, yang seringkali menjadi masalah dalam deteksi *real-time*.

5. KESIMPULAN

Penelitian ini menunjukkan bahwa model *YOLOv9* sangat efektif dalam mendeteksi gerakan tangan alfabet bahasa isyarat, dengan tingkat akurasi mencapai 97% dan *precision* serta *recall* di atas 90% untuk sebagian besar kelas. Dibandingkan dengan model lain, seperti *Faster R-CNN* dan *SSD MobileNet v2*, *YOLOv9* unggul dalam hal kecepatan dan akurasi, sehingga sangat cocok untuk pembuatan aplikasi. Meskipun demikian, model ini menghadapi tantangan dalam mengenali gestur mirip secara visual yang sering kali salah diklasifikasikan. Oleh karena itu, diperlukan optimasi lebih lanjut, baik melalui peningkatan variasi dataset atau teknik augmentasi. Penelitian ini diharapkan dapat berkontribusi pada pengembangan teknologi yang lebih inklusif, yang mendukung komunikasi penyandang disabilitas, seperti aplikasi penerjemah bahasa isyarat secara *real-time*. Untuk penelitian lanjutan, disarankan untuk memperluas variasi dataset dan meningkatkan robustitas model dalam berbagai kondisi.

DAFTAR PUSTAKA

[1] S. Dwijayanti, Hermawati, S. I. Taqiyyah, H. Hikmarika, and B. Y. Suprapto, “Indonesia Sign Language Recognition using Convolutional Neural Network,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 10, pp. 415–422, 2021, doi: 10.14569/IJACSA.2021.0121046.

[2] S. C. Mesbahi, M. A. Mahraz, J. Riffi, and H. Tairi, “Hand Gesture Recognition Based on Various Deep Learning YOLO Models,” *Int.*

J. Adv. Comput. Sci. Appl., vol. 14, no. 4, pp. 307–319, 2023, doi: 10.14569/IJACSA.2023.0140435.

[3] S. Al Ahmadi, F. Mohammad, and H. Al Dawsari, "Efficient YOLO-Based Deep Learning Model for Arabic Sign Language Recognition," *J. Disabil. Res.*, vol. 3, no. 4, 2024, doi: 10.57197/jdr-2024-0051.

[4] M. P. Geetha, S. Swetha, M. Subitsha, and K. V. Visnupriya, "Gesture Based Sign Language Recognition for Specially Challenged Using Yolov5," *Proc. 2024 Int. Conf. Sci. Technol. Eng. Manag. ICSTEM 2024*, pp. 1–4, 2024, doi: 10.1109/ICSTEM61137.2024.10560764.

[5] M. R. Ningsih *et al.*, "Sign Language Detection System Using YOLOv5 Algorithm to Promote Communication Equality People with Disabilities," *Sci. J. Informatics*, vol. 11, no. 2, pp. 549–558, 2024, doi: 10.15294/sji.v11i2.6007.

[6] A. N. Sihananto, E. M. Safitri, Y. Maulana, F. Fakhruddin, and M. E. Yudistira, "Indonesian Sign Language Image Detection Using Convolutional Neural Network (CNN) Method," *Inspir. J. Teknol. Inf. dan Komun.*, vol. 13, no. 1, pp. 13–21, 2023, doi: 10.35585/inspir.v13i1.37.

[7] S. Sharma, R. Sreemathy, M. Turuk, J. Jagdale, and S. Khurana, "Real-Time Word Level Sign Language Recognition Using YOLOv4," *2022 Int. Conf. Futur. Technol. INCOFT 2022*, pp. 1–7, 2022, doi: 10.1109/INCOFT55651.2022.10094530.

[8] M. Alaftekin, I. Pacal, and K. Cicek, "Real-time sign language recognition based on YOLO algorithm," *Neural Comput. Appl.*, vol. 36, no. 14, pp. 7609–7624, 2024, doi: 10.1007/s00521-024-09503-6.

[9] A. Mujahid *et al.*, "Real-time hand gesture recognition based on deep learning YOLOv3 model," *Appl. Sci.*, vol. 11, no. 9, 2021, doi: 10.3390/app11094164.

[10] M. Bhavadharshini, J. Josephine Racheal, M. Kamali, S. Sankar, and M. Bhavadharshini, "Sign language translator using YOLO algorithm," *Adv. Parallel Comput.*, vol. 39, pp. 159–166, 2021, doi: 10.3233/APC210136.

[11] P. Battistoni, M. Di Gregorio, M. Romano, M. Sebillo, G. Vitiello, and G. Solimando, *Sign language interactive learning - measuring the user engagement*, vol. 12206 LNCS. Springer International Publishing, 2020. doi: 10.1007/978-3-030-50506-6_1.

[12] L. Chandwani *et al.*, "Gesture based Sign Language Recognition system using Mediapipe," 2023, [Online]. Available: https://doi.org/10.21203/rs.3.rs-3106646/v1

[13] A. A. Sonkamble, R. D. Chavhan, B. S. Jadhao, and S. M. Rathod, "Real-Time Indian Sign Language Detection using SSD-Mobilenet," *2022 Sardar Patel Int. Conf. Ind. 4.0 - Nascent Technol. Sustain. "Make India" Initiat. SPICON 2022*, pp. 1–5, 2022, doi: 10.1109/SPICON56577.2022.10180839.

[14] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," *Multimed. Tools Appl.*, vol. 82, no. 6, pp. 9243–9275, 2023, doi: 10.1007/s11042-022-13644-y.

[15] N. F. Attia, M. T. F. S. Ahmed, and M. A. M. Alshewimy, "Efficient deep learning models based on tension techniques for sign language recognition," *Intell. Syst. with Appl.*, vol. 20, no. September, p. 200284, 2023, doi: 10.1016/j.iswa.2023.200284.

[16] M. Shashishekhara, H. Hamza, and I. Etit-kit, "Real Time American Sign Language Detection Using Yolo-v9 Amna Imran", doi: 10.48550/arXiv.2407.17950.

[17] A. Al-Shaheen, M. Çevik, and A. Alqaraghuli, "American Sign Language Recognition using YOLOv4 Method," *Int. J. Multidiscip. Stud. Innov. Technol.*, vol. 6, no. 1, p. 61, 2022, doi: 10.36287/ijmsit.6.1.61.

[18] D. Ma, K. Hirota, Y. Dai, and Z. Jia, "Dynamic Sign Language Recognition Based on Improved Residual-LSTM network," *Proc. 7th Int. Work. Adv. Comput. Intell. Intell. Informatics* , pp. 1–6, 2021, [Online]. Available: https://iwaciii2021.bit.edu.cn/docs/2021-12/9caf45c507224bdca4933ad139fff639.pdf

[19] S. A. Khan, Z. A. Ansari, R. Singh, M. S. Rawat, F. Z. Khan, and S. K. Yadav, "Sign Translation Via Natural Language Processing," *Asian J. Res. Comput. Sci.*, vol. 11, no. 1, pp. 1–7, 2021, doi: 10.9734/ajrcos/2021/v11i130251.

[20] L. S. Teja Mangamuri, L. Jain, and A. Sharmay, "Two Hand Indian Sign Language dataset for benchmarking classification models of Machine Learning," *IEEE Int. Conf. Issues Challenges Intell. Comput. Tech. ICICT 2019*, pp. 2–6, 2019, doi: 10.1109/ICICT46931.2019.8977713.

[21] J. Zhao, X. H. Li, J. C. D. Cruz, M. S. Verdadero, J. C. Centeno, and J. M. Novelero, "Hand Gesture Recognition Based on Deep Learning," *2023 Int. Conf. Digit. Appl. Transform. Econ. ICDATE 2023*, vol. 14, no. 4, pp. 307–320, 2023, doi: 10.1109/ICDATE58146.2023.10248500.

[22] T. F. Dima and M. E. Ahmed, "Using

YOLOv5 Algorithm to Detect and Recognize American Sign Language," *2021 Int. Conf. Inf. Technol. ICIT 2021 - Proc.*, pp. 603–607, 2021, doi: 10.1109/ICIT52682.2021.9491672.

[23] D. A. Abdurrafi, M. T. Alawiy, and B. M. Basuki, "Deteksi Klasifikasi Dan Menghitung Kendaraan Berbasis Algoritma You Only Look Once (YOLO) Menggunakan Kamera CCTV," *Sci. Electro*, no. 1, pp. 1–9, 2023, [Online]. Available: https://jim.unisma.ac.id/index.php/jte/article/viewFile/21551/16069

[24] M. P. K. Putra and Wahyono, "A Novel Method for Handling Partial Occlusion on Person Re-identification using Partial Siamese Network," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 7, pp. 313–321, 2021, doi: 10.14569/IJACSA.2021.0120735.

[25] D. D. Aboyomi and C. Daniel, "A Comparative Analysis of Modern Object Detection Algorithms: YOLO vs. SSD vs. Faster R-CNN," *ITEJ (Information Technol. Eng. Journals)*, vol. 8, no. 2, pp. 96–106, 2023, doi: 10.24235/itej.v8i2.123.

[26] R. H. Abiyev, M. Arslan, and J. B. Idoko, "Sign language translation using deep convolutional neural networks," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 2, pp. 631–653, 2020, doi: 10.3837/tiis.2020.02.009.

[27] S. Tyagi, P. Upadhyay, I. Hoor Fatima, and A. Kumar Sharma, "American Sign Language Detection using YOLOv5 and YOLOv8," pp. 1–16, 2023, [Online]. Available: https://doi.org/10.21203/rs.3.rs-3126918/v1.