

Implementasi YOLOv10 untuk Pengenalan Alfabet SIBI berbasis Deteksi Gerakan Tangan

DOI: <http://dx.doi.org/10.35889/jutisi.v14i3.2953>

Creative Commons License 4.0 (CC BY – NC)

Selvie^{1*}, Yohannes²

Informatika, Universitas Multi Data Palembang, Palembang, Indonesia

*e-mail *Corresponding Author*: selvietan.mukhua@gmail.com

Abstract

Sign language is a primary communication tool for people with hearing and speech impairments. One key component of the Indonesian Sign System (SIBI) is the SIBI alphabet, used for words without specific signs. However, limited public understanding creates communication gaps. This study develops a web-based SIBI alphabet recognition system using the YOLOv10 model. The dataset initially contained 6,110 images and was augmented to 15,859 images. Eight hyperparameter settings were tested, with the best result at a learning rate of 0.001, batch size 32, and Adam optimizer, achieving mAP@50 of 0.988 and mAP@50–95 of 0.971. The model was then converted to ONNX for faster inference, yielding mAP@50 of 0.953 and 797.7 ms per image. The ONNX model was integrated into a web app capable of real-time SIBI recognition. Results show this approach is effective, inclusive, and accessible as an assistive communication tool.

Keywords: Deep learning; SIBI Alphabet; YOLOv10

Abstrak

Bahasa isyarat merupakan sarana utama komunikasi bagi penyandang tunarungu dan tunawicara. Salah satu komponennya, alfabet Sistem Isyarat Bahasa Indonesia (SIBI), digunakan untuk menyampaikan kata yang tidak memiliki padanan isyarat khusus. Namun, kurangnya pemahaman masyarakat terhadap alfabet SIBI menimbulkan kesenjangan komunikasi. Penelitian ini bertujuan mengembangkan sistem pengenalan alfabet SIBI berbasis web menggunakan model YOLOv10. Dataset awal berjumlah 6.110 gambar dan diaugmentasi menjadi 15.859 gambar. Model dilatih menggunakan delapan konfigurasi *hyperparameter* yang menghasilkan performa terbaik pada *learning rate* 0,001, *batch size* 32, dan *optimizer* Adam dengan mAP@50 sebesar 0,988 dan mAP@50–95 sebesar 0,971. Untuk meningkatkan efisiensi, model dikonversi ke format ONNX dan diuji ulang menghasilkan mAP@50 sebesar 0,953 serta waktu inferensi 797,7 ms per gambar. Model ONNX kemudian diintegrasikan ke aplikasi web yang mampu mengenali alfabet SIBI secara *real-time*. Hasil menunjukkan pendekatan ini efektif sebagai alat bantu komunikasi yang inklusif dan mudah diakses.

Kata kunci: Alfabet SIBI; Deep learning; YOLOv10

1. Pendahuluan

Bahasa isyarat merupakan salah satu sarana komunikasi utama bagi individu tunarungu maupun tunawicara. Di Indonesia sendiri terdapat dua sistem bahasa isyarat yang umum digunakan, yakni Bahasa Isyarat Indonesia (BISINDO) dan Sistem Isyarat Bahasa Indonesia (SIBI). SIBI digunakan secara formal dalam lingkungan pendidikan karena mengikuti struktur tata bahasa Indonesia. Salah satu elemen penting dalam SIBI adalah alfabet SIBI, yaitu representasi visual huruf-huruf alfabet melalui konfigurasi gerakan tangan atau posisi jari tertentu. Keberadaan alfabet ini sangat penting dalam menyampaikan nama diri, istilah baru, atau kata-kata yang belum memiliki padanan gerakan khusus dalam SIBI [1].

Namun, alfabet SIBI tidak selalu mudah dipahami oleh masyarakat umum yang tidak terbiasa menggunakan bahasa isyarat. Situasi ini menimbulkan kesenjangan komunikasi antara penyandang tunarungu dengan lingkungan sekitarnya, terutama ketika tidak ada pendamping atau penerjemah yang dapat membantu. Kondisi tersebut berdampak pada keterlibatan mereka

dalam aktivitas sosial, pendidikan, maupun layanan publik. Meskipun upaya sosialisasi bahasa isyarat terus dilakukan, masih diperlukan pendekatan tambahan yang dapat meningkatkan pemahaman serta akses komunikasi lintas kelompok. Salah satu solusi yang semakin banyak dimanfaatkan adalah teknologi berbasis *deep learning*.

Sejumlah penelitian telah berupaya mengembangkan sistem pengenalan bahasa isyarat untuk mengurangi hambatan komunikasi antara penyandang tunarungu dan masyarakat luas. Secara garis besar, teknologi yang digunakan umumnya terbagi menjadi dua pendekatan, yaitu perangkat berbasis sensor dan perangkat berbasis kamera. Metode sensor, seperti sarung tangan elektronik atau akselerometer [2], mampu menangkap gerakan tangan dengan tingkat akurasi tinggi, namun membutuhkan perangkat tambahan yang relatif mahal dan kurang praktis untuk aktivitas sehari-hari. Di sisi lain, pendekatan berbasis kamera lebih mudah diakses karena dapat dijalankan menggunakan perangkat umum seperti webcam, laptop, atau smartphone [3].

Meski demikian, pemanfaatan kamera dalam sistem pengenalan bahasa isyarat juga memiliki tantangan, terutama dalam pemilihan model deteksi objek yang tidak hanya akurat tetapi juga cepat dalam melakukan inferensi [4]. Kinerja sistem sangat dipengaruhi oleh model deteksi yang digunakan, sehingga pemilihan algoritma menjadi faktor yang krusial. Oleh sebab itu, diperlukan model deteksi objek yang mampu menghasilkan akurasi baik sekaligus memiliki efisiensi tinggi untuk diimplementasikan dalam sistem pengenalan isyarat berbasis kamera.

Salah satu algoritma yang banyak digunakan untuk kebutuhan deteksi objek *real-time* adalah YOLO (*You Only Look Once*). Algoritma ini mengombinasikan kecepatan dan akurasi dalam satu tahap proses deteksi, dan telah banyak dimanfaatkan dalam aplikasi pengenalan gestur serta bahasa isyarat karena kemampuannya memproses citra secara langsung tanpa tahapan *region proposal* [5]. YOLOv10 menghadirkan sejumlah peningkatan dibandingkan generasi sebelumnya seperti YOLOv8 dan SSD. Versi ini menerapkan pelatihan tanpa NMS (*Non-Maximum Suppression*) melalui mekanisme *consistent dual assignments*, serta melakukan optimisasi komponen model untuk meningkatkan efisiensi sekaligus menjaga akurasi [6]. Hal tersebut membuat YOLOv10 tetap kompeten dan efisien untuk deteksi objek secara *real-time*.

Berdasarkan latar belakang tersebut, penelitian ini berfokus pada pengembangan sistem pengenalan alfabet SIBI berbasis *deep learning*. Sebelum diintegrasikan ke dalam perangkat lunak, model YOLOv10 diuji menggunakan delapan konfigurasi *hyperparameter* yang merupakan kombinasi dari *batch size*, *learning rate*, dan jenis *optimizer*. Setiap konfigurasi dievaluasi menggunakan metrik *mean Average Precision* (mAP) untuk menilai akurasi, sekaligus waktu inferensi untuk mengukur kecepatan pemrosesan. Model dengan performa terbaik kemudian digunakan sebagai bagian dari sistem perangkat lunak.

2. Tinjauan Pustaka

Sejumlah penelitian sebelumnya telah mengeksplorasi pengembangan sistem pengenalan bahasa isyarat menggunakan pendekatan *deep learning*, khususnya menggunakan pendekatan deteksi objek dari algoritma YOLO. Penelitian [7] menggunakan YOLOv8 untuk mengenali isyarat abjad SIBI. Studi tersebut mengevaluasi empat model dengan variasi parameter pelatihan dan menemukan konfigurasi optimal yang mencapai mAP sebesar 0.853. Penelitian ini menunjukkan pemilihan parameter pelatihan dapat berdampak signifikan terhadap akurasi model.

Studi lain oleh [8] menggunakan YOLOv7 untuk mengenali abjad BISINDO. Dengan parameter pelatihan yang optimal (*epoch* = 100, *batch size* = 64, *learning rate* = 0.001), model ini berhasil mencapai mAP 0.995, dengan *recall* dan *precision* tinggi. Namun, penerapan model pada sistem *real-time* menunjukkan penurunan performa, yang menandakan masih adanya tantangan dalam integrasi ke sistem yang ringan dan responsif.

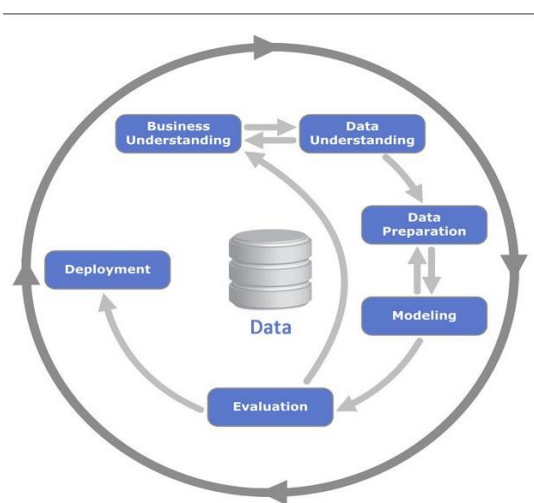
Penelitian [9] juga memanfaatkan metode YOLO dan NMS untuk mengenali isyarat SIBI. Dengan pelatihan selama 100 *epoch*, model mencapai akurasi tinggi sebesar 98,5% mAP, dan menunjukkan hasil deteksi yang konsisten pada tahap evaluasi. Studi ini menekankan potensi YOLO dalam pengenalan gerakan tangan secara akurat.

Penelitian lain yang dilakukan oleh [10] dan [11] menggunakan YOLOv8 dan YOLOv5 dalam konteks yang serupa. YOLOv8 menunjukkan akurasi deteksi sebesar 99,5%, namun memiliki rata-rata waktu inferensi yang cukup tinggi, yaitu sekitar 4046 ms. Di sisi lain, YOLOv5 digunakan untuk mengembangkan aplikasi pembelajaran SIBI, dan hasil deteksinya menunjukkan tingkat akurasi 77%, dengan beberapa kelas mengalami ambiguitas karena gestur yang serupa.

Hingga kini, belum ditemukan studi yang secara langsung menilai performa YOLOv10 dalam mendeteksi alfabet SIBI, meskipun versi-versi sebelumnya seperti YOLOv5, YOLOv7, dan YOLOv8 telah banyak digunakan untuk tugas serupa. Penelitian ini menghadirkan kebaruan dengan melakukan eksperimen terhadap 8 kombinasi *hyperparameter* pada YOLOv10, mengevaluasi performa akurasi dan efisiensi model, serta mengintegrasikan model terbaik ke dalam perangkat lunak pengenalan alfabet SIBI.

3. Metodologi

Penelitian ini menggunakan kerangka kerja CRISP-DM (*Cross-Industry Standard Process for Data Mining*) sebagai panduan dalam merancang sistem pengenalan alfabet SIBI berbasis YOLOv10. CRISP-DM merupakan metodologi standar dalam pengembangan sistem berbasis data mining yang mencakup enam tahap utama: *business understanding*, *data understanding*, *data preparation*, *modeling*, *evaluation*, serta *deployment* [12]. Alur setiap tahap CRISP-DM ditampilkan pada Gambar 1.



Gambar 1 Metode CRISP-DM [13]

3.1 Business Understanding

Penyandang tunarungu dan tunawicara masih menghadapi hambatan komunikasi karena tidak semua orang memahami bahasa isyarat, termasuk alfabet SIBI. Padahal alfabet SIBI diperlukan untuk menyampaikan istilah atau kata yang belum memiliki gerakan khusus. Oleh karena itu, penelitian ini diarahkan untuk membangun sistem pengenalan alfabet SIBI berbasis *deep learning* yang dapat bekerja secara *real-time*. Fokus utama penelitian adalah mengevaluasi performa YOLOv10 dengan beberapa variasi *hyperparameter* guna memperoleh model paling optimal sebelum diintegrasikan ke perangkat lunak.

3.2 Data Understanding

Dataset penelitian terdiri dari 6.110 citra alfabet SIBI, yang diperoleh dari dataset publik Roboflow [14] dan hasil pengambilan gambar secara mandiri. Setiap huruf A–Z memiliki jumlah gambar yang merata, yakni 235 citra per kelas. Semua gambar menampilkan variasi latar belakang untuk meningkatkan kemampuan generalisasi model. Contoh representasi visual alfabet SIBI ditunjukkan pada Tabel 1.

Tabel 1 Isyarat Alfabet SIBI

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

3.3 Data Preparation

Tahap persiapan data mencakup penghapusan gambar buram atau tidak relevan, serta pembuatan anotasi *bounding box* secara manual menggunakan Roboflow. Semua gambar disesuaikan ke format input YOLOv10, yaitu 640×640 piksel. Dataset awal dibagi menjadi tiga bagian, yaitu 80% untuk pelatihan, 10% untuk validasi, dan 10% untuk pengujian. Setelah pembagian dataset, augmentasi ditambahkan untuk memperluas variasi data sehingga model dapat lebih adaptif terhadap kondisi nyata. Temuan [15] membuktikan teknik augmentasi dapat meningkatkan performa deteksi objek secara signifikan. Tabel 2 memuat ringkasan pembagian dataset setelah augmentasi diterapkan.

Tabel 2 Pembagian Dataset

Kategori Data	Jumlah	Persentase
Pelatihan	14.663	92%
Validasi	598	4%
Pengujian	598	4%

3.4 Modeling

Penelitian ini menggunakan YOLOv10m, varian menengah dalam keluarga YOLOv10 yang dirancang untuk mencapai keseimbangan antara akurasi dan kecepatan inferensi. Arsitektur YOLOv10 terdiri dari tiga bagian utama, yaitu *Backbone* sebagai ekstraktor fitur, *Neck* berbasis Path Aggregation Network yang menggabungkan informasi dari berbagai skala, dan *Head* yang menghasilkan prediksi *bounding box* serta klasifikasi objek. Mengacu pada penelitian [6], pelatihan model YOLOv10 dilakukan tanpa menggunakan proses NMS pada saat inferensi berkat penerapan *dual label assignments*, yaitu kombinasi dari pendekatan *one-to-many* dan *one-to-one matching*. Kedua metode ini digunakan secara bersamaan selama pelatihan untuk memberikan sinyal supervisi yang lebih kaya dari *one-to-many*, serta efisiensi dan presisi prediksi dari *one-to-one*. Untuk memastikan kedua cabang ini belajar secara konsisten, YOLOv10 menggunakan *consistent matching metric* yang menyelaraskan proses pembelajaran keduanya. Dengan pendekatan ini, YOLOv10 dapat dilatih secara end-to-end dengan prediksi langsung dari *one-to-one head*, sehingga hasil akhir lebih efisien tanpa mengorbankan akurasi.

Pelatihan model dilakukan di platform Kaggle Notebook dengan memanfaatkan GPU NVIDIA Tesla T4. Proses dimulai dengan instalasi pustaka yang diperlukan seperti Ultralytics YOLO, Supervision untuk visualisasi, dan Roboflow untuk mengunduh dataset SIBI. Dataset disiapkan dalam format YOLO dan dimuat menggunakan API Roboflow. Model yang digunakan adalah yolov10m.pt, yaitu model pra-latih dari repositori YOLOv10. Model dilatih menggunakan perintah *yolo train* dari Ultralytics dengan konfigurasi *batch size*, *learning rate*, dan *optimizer* yang divariasikan dan dilatih menggunakan 50 *epoch*. Seluruh proses dilengkapi dengan pelacakan metrik menggunakan W&B (Weights & Biases) untuk memantau performa model secara *real-time*.

Sebanyak 8 konfigurasi hyperparameter diuji, mengombinasikan dua nilai *batch size* (16, 32), dua *learning rate* (0.001 dan 0.0001), serta dua optimizer (Adam dan SGD). Setiap model dilatih selama 50 epoch dan dievaluasi menggunakan mAP@50, mAP@50–95, serta waktu inferensi rata-rata per gambar untuk menilai kinerja dalam mengenali alfabet SIBI. Metrik mAP dipilih karena secara umum digunakan untuk mengukur akurasi model deteksi objek.

3.5 Evaluation

Hasil evaluasi dari masing-masing konfigurasi pelatihan secara langsung diperoleh dari *output* Ultralytics yang ditampilkan pada Tabel 3.

Tabel 3 Hasil Pengujian Model

No.	Learning rate	Batch size	Optimizer	mAP50	mAP 50-95
1.	0,001	16	Adam	0,984	0,953
2.	0,001	16	SGD	0,987	0,961
3.	0,001	32	Adam	0,988	0,971
4.	0,001	32	SGD	0,989	0,968
5.	0,0001	16	Adam	0,988	0,963
6.	0,0001	16	SGD	0,984	0,954
7.	0,0001	32	Adam	0,985	0,962
8.	0,0001	32	SGD	0,985	0,957

Dari delapan kombinasi yang diuji, konfigurasi dengan *learning rate* 0.001, *batch size* 32, dan optimizer Adam menghasilkan performa tertinggi, yaitu mAP@50 sebesar 0.988 dan mAP@50–95 sebesar 0.971. Model ini kemudian diuji pada dataset pengujian untuk menilai generalisasi.

Pada format PyTorch, model mencapai mAP@50 sebesar 0.986, mAP@50–95 sebesar 0.952, namun waktu inferensinya 2.072 ms per gambar, yang tergolong lambat untuk aplikasi *real-time*. Oleh karena itu, model dikonversi ke format ONNX dan diuji kembali. Format ONNX memberikan mAP@50 sebesar 0.953, mAP@50–95 sebesar 0.953, serta waktu inferensi 797.7 ms per gambar. Ringkasan perbandingan performa kedua format model terdapat pada Tabel 4.

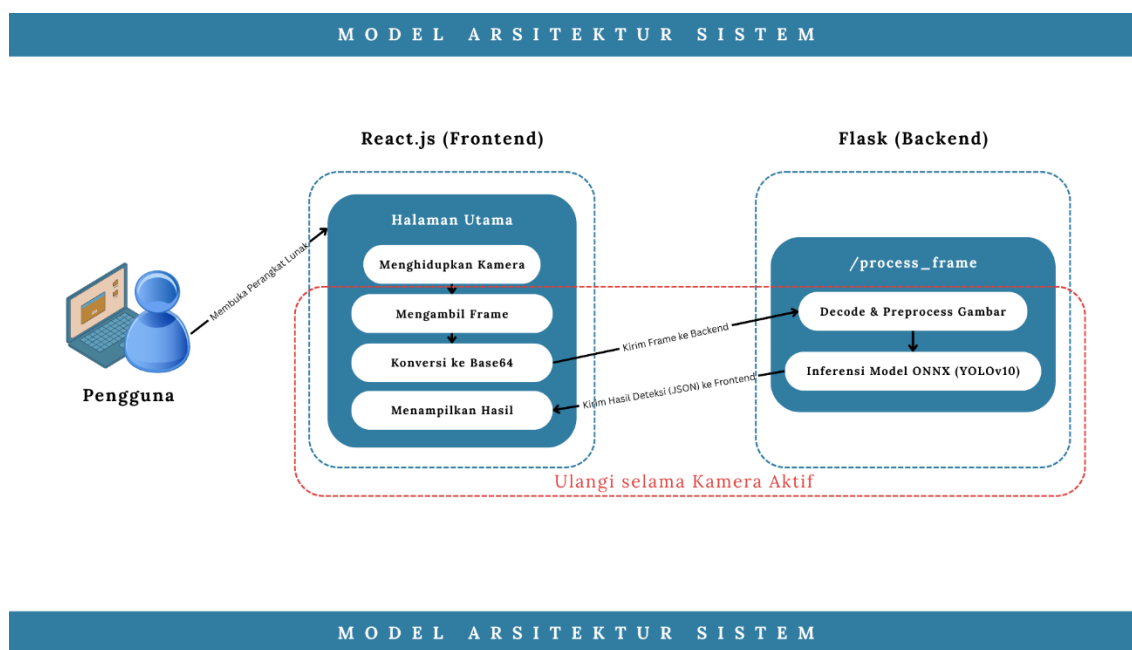
Tabel 4 Perbandingan Performa Model PyTorch dan ONNX

Format Model	mAP@50	mAP@50–95	Waktu Inferensi (ms/gambar)
PyTorch	0,988	0,971	2.072
ONNX	0,953	0,953	797,7

Meskipun akurasi sedikit menurun, model ONNX dipilih karena mampu mempercepat waktu inferensi secara signifikan tanpa mengorbankan performa. Oleh karena itu, model hasil konversi ke format ONNX dipilih sebagai model akhir yang akan diintegrasikan ke dalam perangkat lunak pengenalan alfabet SIBI.

3.6 Deployment

Setelah model dikembangkan, model tersebut diimplementasikan ke dalam perangkat lunak berbasis web yang terdiri dari *frontend* dan *backend*. *Frontend* dikembangkan menggunakan React untuk menampilkan antarmuka, mengakses kamera pengguna, serta mengelola proses pengambilan frame secara berkala. Setiap 800 milidetik, satu frame gambar dari kamera dikonversi ke format base64 dan dikirim ke *backend* melalui REST API. Pada sisi *backend*, digunakan Flask untuk menjalankan model deteksi dalam format ONNX melalui ONNX Runtime. *Backend* menerima data gambar dalam format base64, melakukan proses *decoding* dan *preprocessing* sebelum menjalankan inferensi menggunakan model YOLOv10. Hasil deteksi kemudian dikirim kembali ke *frontend* dalam format JSON. *Frontend* menampilkan hasil tersebut secara *real-time* dalam bentuk teks dan gambar di antarmuka pengguna. Proses ini berlangsung secara berulang selama kamera aktif, sehingga perangkat lunak dapat mengenali alfabet SIBI melalui kamera perangkat pengguna secara *real-time* tanpa perlu instalasi tambahan. Diagram arsitektur sistem ditampilkan pada Gambar 2.



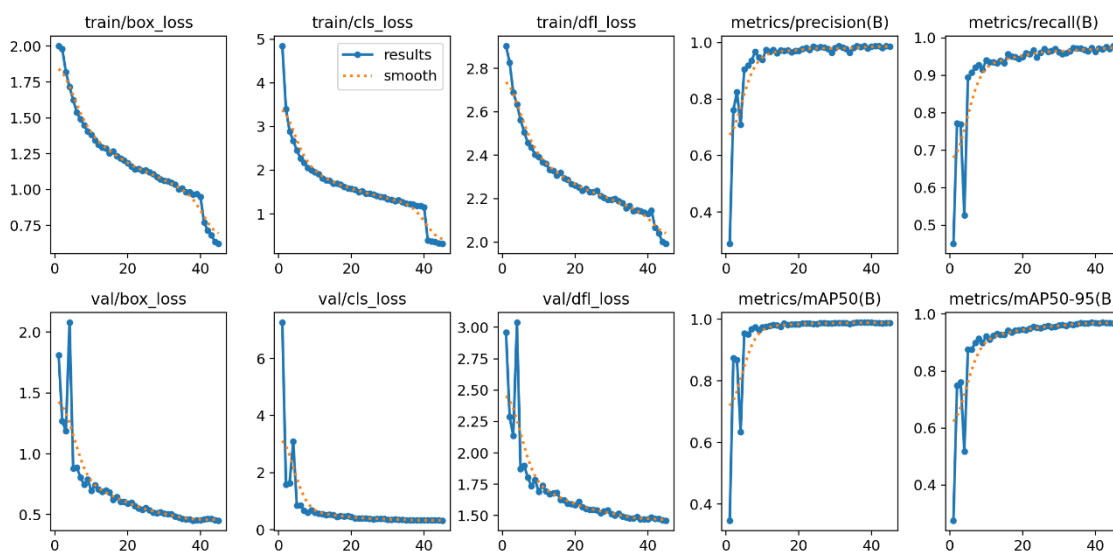
Gambar 2 Model Arsitektur Sistem

4. Hasil dan Pembahasan

4.1 Hasil Eksperimen

Dataset alfabet SIBI yang digunakan pada penelitian ini merupakan hasil augmentasi, sehingga total data yang diperoleh mencapai 15.859 gambar. Data tersebut dibagi menjadi 14.663 gambar untuk pelatihan, 598 gambar untuk validasi, dan 598 gambar untuk pengujian. Model dilatih selama 50 *epoch* menggunakan YOLOv10m dan diuji dengan 8 kombinasi *hyperparameter*, yakni gabungan dari dua nilai *learning rate* (0,001 dan 0,0001), dua *batch size* (16 dan 32), serta dua jenis *optimizer* (Adam dan SGD). Proses pelatihan dijalankan melalui perintah *yolo train* pada pustaka Ultralytics dengan dukungan GPU melalui Kaggle Notebook, dan pemantauan performa dilakukan menggunakan W&B (*Weights & Biases*).

Model YOLOv10m yang digunakan terdiri dari 159 *layer*, total 16,5 juta parameter, serta beban komputasi sebesar 63,6 GFLOPs. Kombinasi yang memberikan performa terbaik adalah *learning rate* 0,001, *batch size* 32, dan *optimizer* Adam, yang menghasilkan mAP@50 sebesar 0,988 dan mAP@50–95 sebesar 0,971. Visualisasi proses pelatihan yang ditampilkan pada Gambar 2 menampilkan tren *train_loss*, *val_loss*, *precision*, *recall*, serta mAP pada berbagai threshold. Selama proses training, nilai *box_loss* dan *cls_loss* menurun secara konsisten, sementara metrik evaluasi meningkat stabil seiring bertambahnya epoch. Pada akhir pelatihan, nilai *train/box_loss* mencapai 0,62, *train/cls_loss* sebesar 0,31, dan *train/dfl_loss* sebesar 1,99. Untuk data validasi, *val/box_loss* tercatat 0,45, *val/cls_loss* 0,32, dan *val/dfl_loss* 1,46. Secara keseluruhan, model mencatat *precision* 0,987 dan *recall* 0,969. Tabel 5 menunjukkan bahwa sebagian besar huruf alfabet SIBI memiliki nilai mAP di atas 0,95.



Gambar 3 Grafik Hasil Pelatihan Model Terbaik

Tabel 5 Hasil Pelatihan Model Per Kelas

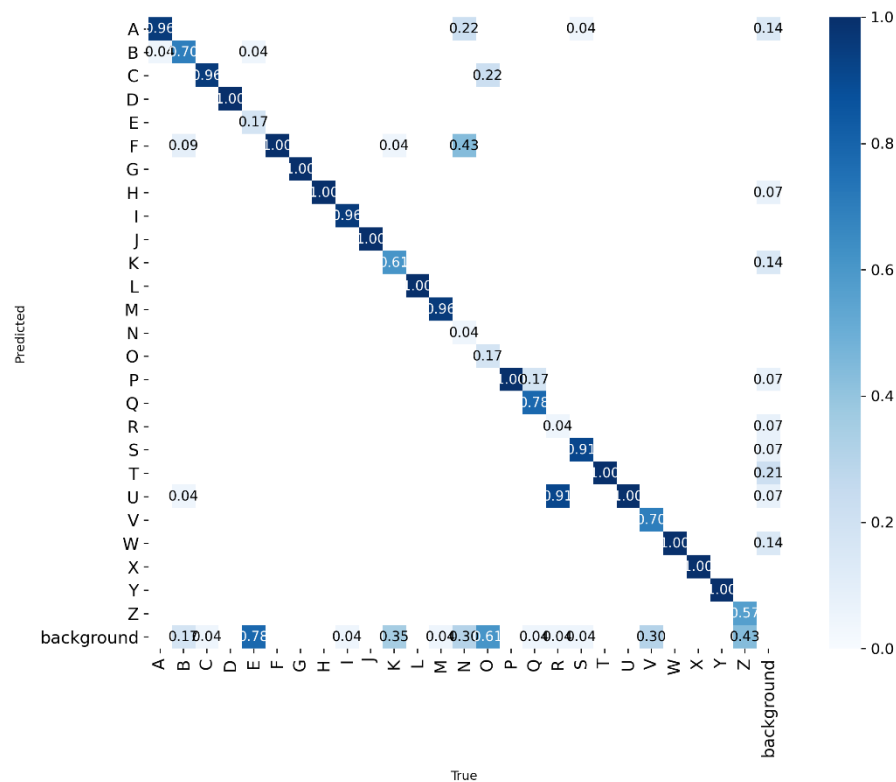
Kelas	Precision	Recall	mAP@50	mAP@50–95
All	0.987	0.969	0.988	0.971
A	0.993	0.962	0.978	0.962
B	1.000	0.991	0.995	0.982
C	0.997	1.000	0.995	0.991
D	0.943	1.000	0.995	0.995
E	0.954	1.000	0.995	0.987
F	0.996	1.000	0.995	0.982
G	0.992	1.000	0.995	0.975
H	0.995	1.000	0.995	0.968
I	1.000	0.922	0.995	0.992
J	1.000	0.926	0.975	0.941
K	0.947	0.900	0.926	0.900
L	0.993	0.960	0.978	0.971

Kelas	Precision	Recall	mAP@50	mAP@50-95
M	0.976	0.960	0.993	0.976
N	1.000	0.898	0.993	0.972
O	1.000	0.934	0.977	0.942
P	0.992	1.000	0.995	0.935
Q	1.000	0.959	0.995	0.933
R	1.000	0.977	0.995	0.995
S	1.000	0.992	0.995	0.958
T	0.928	1.000	0.988	0.974
U	0.990	1.000	0.995	0.990
V	0.994	0.963	0.980	0.980
W	0.995	1.000	0.995	0.993
X	1.000	0.904	0.995	0.995
Y	0.990	1.000	0.995	0.992
Z	0.988	0.947	0.990	0.966

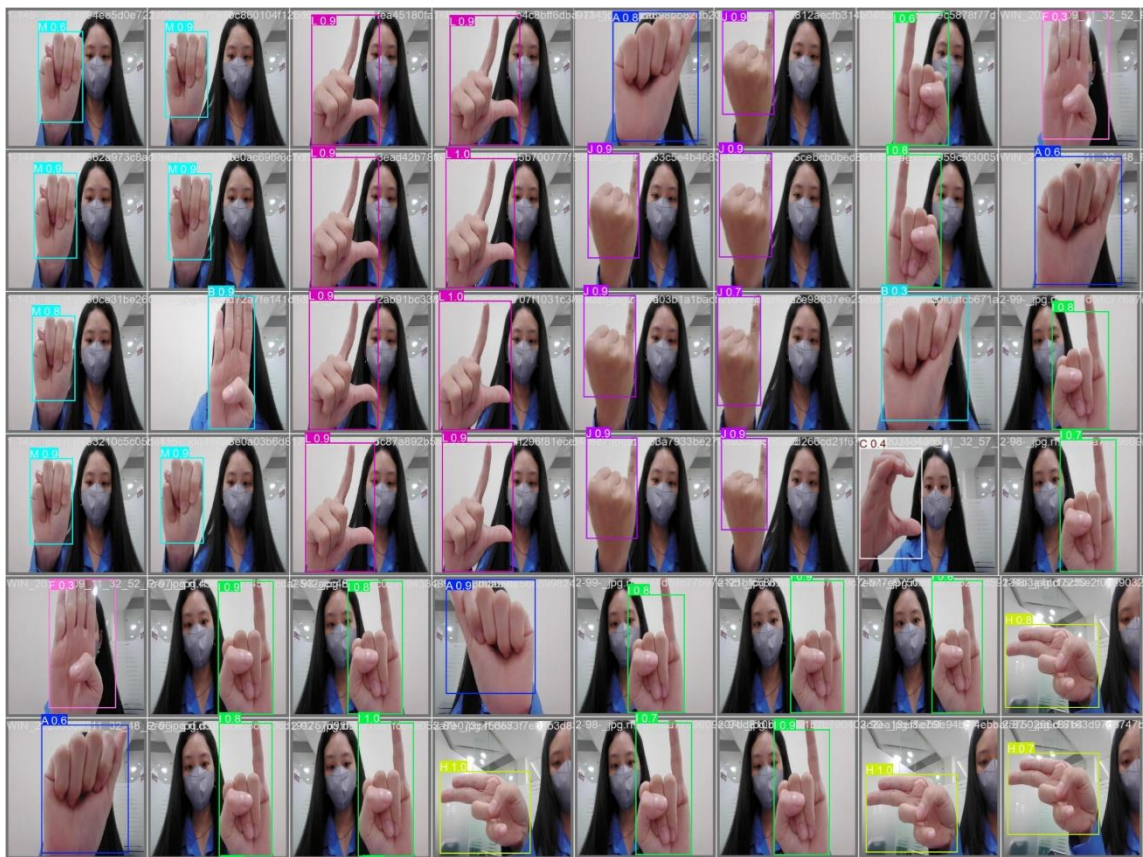


Gambar 4 Visualisasi Hasil Pelatihan Model YOLOv10

Model terbaik kemudian diuji menggunakan data uji yang belum pernah ditemui sebelumnya. Pada format PyTorch, model menghasilkan mAP@50 sebesar 0,912, mAP@50-95 sebesar 0,737, precision 0,853, dan recall 0,847. Confusion matrix pada Gambar 5 menunjukkan sebagian besar huruf dikenali dengan benar, ditandai oleh nilai diagonal utama yang tinggi, misalnya huruf D, F, G, H, J, L, P, T, U, W, X, dan Y yang mencapai nilai 1,00. Meski demikian, huruf N dan R memiliki nilai diagonal sebesar 0,04, menandakan model masih mengalami kesulitan pada huruf dengan kemiripan visual yang tinggi. Sementara itu, nilai di luar diagonal umumnya berada pada rentang 0,04 hingga 0,22, yang mengindikasikan tingkat kesalahan relatif kecil dan tidak signifikan. Secara keseluruhan, hasil ini membuktikan model telah bekerja dengan baik, meskipun masih diperlukan perbaikan pada huruf-huruf yang memiliki kemiripan visual. Hasil pengujian pengenalan alfabet SIBI dapat dilihat pada gambar 6 yang menunjukkan sebagian besar alfabet SIBI dikenali dengan benar serta memperoleh akurasi yang tinggi.



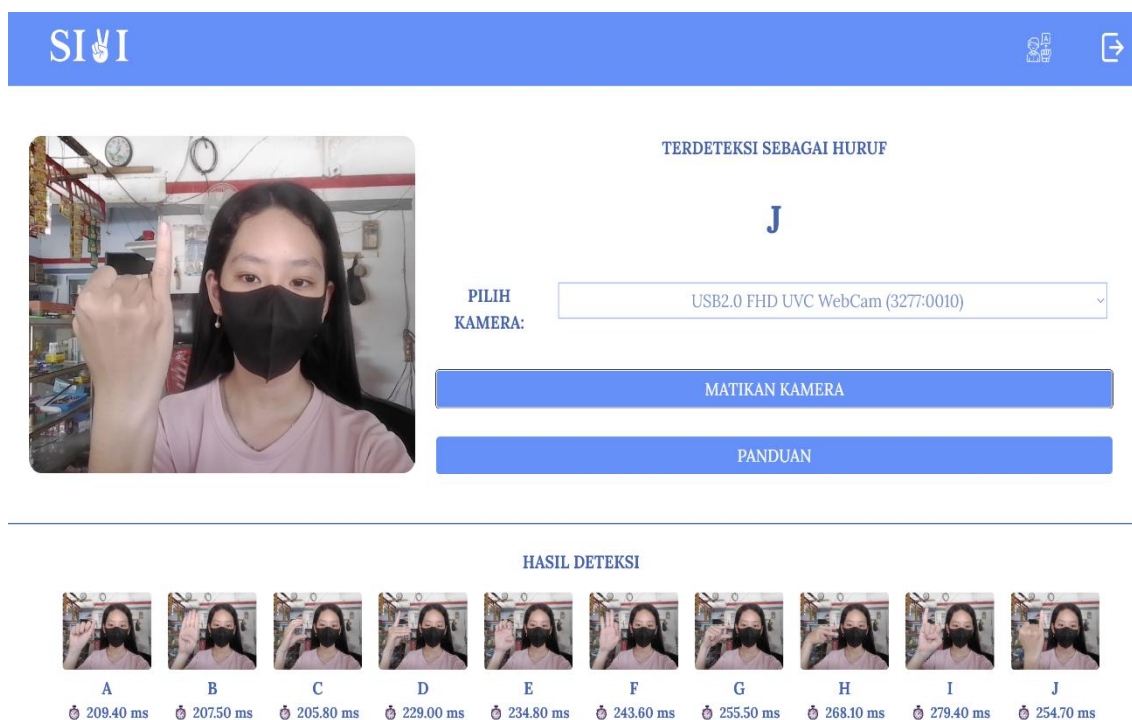
Gambar 5 Hasil Confusion Matrix



Gambar 6 Visualisasi Pengujian Deteksi Alfabet SIBI

Namun, waktu inferensi rata-rata model adalah 2.072 milidetik per gambar. Waktu ini tergolong lambat untuk penggunaan secara *real-time*. Oleh karena itu, model dikonversi ke format ONNX untuk meningkatkan efisiensi. Setelah konversi, model diuji kembali dan memperoleh mAP@50 sebesar 0,953 dan mAP@50–95 sebesar 0,953, dengan waktu inferensi yang menurun drastis menjadi 797,7 milidetik per gambar. Meski akurasi sedikit menurun, model ONNX digunakan untuk integrasi perangkat lunak karena waktu inferensinya lebih efisien.

Model ONNX yang telah dikonversi kemudian diintegrasikan ke *backend* dan berhasil dijalankan tanpa kendala. Hasil pengujian perangkat lunak yang dapat dilihat pada Gambar 6 menunjukkan model ONNX berhasil menerima input kamera dan mengenali alfabet SIBI secara *real-time*. Dengan waktu inferensi di bawah 1 detik per gambar, perangkat lunak ini mampu memberikan respons yang efisien untuk mendukung interaksi pengguna secara *real-time*.



Gambar 7 Hasil Pengujian Perangkat Lunak

4.2 Pembahasan

Hasil penelitian memperlihatkan bahwa YOLOv10m memiliki performa yang sangat baik dalam mengenali isyarat alfabet SIBI, yang ditunjukkan oleh nilai mAP yang tinggi pada tahap validasi maupun pengujian. Akurasi tinggi pada sebagian besar huruf membuktikan bahwa pendekatan ini memiliki potensi besar untuk mendukung komunikasi bagi penyandang tunarungu dan tunawicara. Dengan kemampuan deteksi yang baik dan waktu respons yang efisien setelah konversi ke format ONNX, model ini dapat diintegrasikan ke dalam perangkat lunak berbasis web dan dapat digunakan secara *real-time* melalui kamera perangkat pengguna.

Jika dibandingkan dengan penelitian sebelumnya, performa model dalam penelitian ini menunjukkan peningkatan yang cukup signifikan. Penelitian [7] yang menggunakan YOLOv8 untuk pengenalan alfabet SIBI, menghasilkan mAP tertinggi sebesar 0,853 dengan dataset berjumlah 520 gambar. Sementara itu, penelitian ini menggunakan dataset yang jauh lebih besar, yaitu 15.859 gambar dan memperoleh mAP@50 sebesar 0,988 serta mAP@50–95 sebesar 0,971. Hasil ini menunjukkan bahwa jumlah dataset yang digunakan dan pemilihan *hyperparameter* berpengaruh signifikan terhadap performa model, sekaligus menunjukkan bahwa peningkatan kompleksitas model pada YOLOv10 mampu menghasilkan akurasi yang lebih baik.

Penelitian [8] menggunakan YOLOv7 untuk mendeteksi alfabet BISINDO dan mencapai mAP@50 sebesar 0,995 dengan *recall* dan *precision* sempurna. Namun, hasil deteksi pada implementasi *real-time* tidak sebaik hasil pelatihan, hal ini menunjukkan bahwa model belum sepenuhnya stabil saat digunakan secara langsung. Sebaliknya, model YOLOv10 dalam

penelitian ini mampu mempertahankan akurasi tinggi sekaligus berfungsi stabil dalam pengujian *real-time* bahkan setelah dikonversi ke format ONNX, sehingga memperkuat temuan [9] mengenai pentingnya efisiensi dan kestabilan model pada implementasi.

Sementara itu, penelitian [10] yang menggunakan YOLOv8s untuk mendeteksi gestur tangan dengan 10 kelas mencapai akurasi sebesar 99,5%, namun waktu inferensi rata-rata mencapai 4046 milidetik per gambar. Hasil penelitian ini menunjukkan peningkatan efisiensi yang signifikan, dengan waktu inferensi yang jauh lebih cepat yaitu 797,7 milidetik per gambar setelah konversi ke format ONNX, tanpa kehilangan akurasi yang berarti. Temuan ini menunjukkan bahwa YOLOv10 lebih efisien dibandingkan versi sebelumnya, terutama untuk penerapan secara *real-time*. Dengan demikian, penelitian ini memberikan penguatan terhadap literatur sebelumnya dengan menunjukkan bahwa penggunaan YOLOv10 mampu mengatasi keterbatasan utama dari penelitian terdahulu, yaitu rendahnya efisiensi inferensi dan ketidakstabilan akurasi pada sistem *real-time*.

Namun, penelitian ini memiliki beberapa keterbatasan. Model belum diuji pada kondisi pencahayaan ekstrem, variasi sudut pengambilan gambar, serta perbedaan bentuk tangan antar individu yang dapat memengaruhi akurasi deteksi. Selain itu, model juga masih menunjukkan akurasi yang rendah pada beberapa huruf tertentu, yang berpotensi disebabkan oleh kemiripan visual antar gestur atau distribusi data yang belum optimal. Hal ini menjadi tantangan yang perlu diperhatikan dalam pengembangan selanjutnya.

Penelitian ini memberikan landasan kuat untuk pengembangan lebih lanjut, seperti peningkatan kualitas dan keragaman data, penerapan teknik augmentasi lanjutan, serta perluasan cakupan sistem untuk mengenali gestur dinamis dan kalimat. Jika dikembangkan lebih jauh, sistem ini berpotensi diintegrasikan ke dalam perangkat mobile, layanan publik, dan institusi pendidikan sebagai alat bantu komunikasi bagi komunitas disabilitas. Dengan penyempurnaan berkelanjutan, sistem ini diharapkan mampu menjadi solusi inovatif dan inklusif dalam mendukung komunikasi penyandang tunarungu dan tunawicara secara lebih luas dan efektif.

5. Simpulan

Penelitian ini berhasil mengembangkan sistem pengenalan alfabet SIBI berbasis YOLOv10m menggunakan pendekatan CRISP-DM. Model dilatih menggunakan dataset hasil augmentasi sebanyak 15.859 gambar menggunakan 8 kombinasi *hyperparameter*. Dari seluruh pengujian, kombinasi parameter terbaik diperoleh pada *learning rate* 0,001, *batch size* 32, dan *optimizer* Adam, yang menghasilkan performa tinggi dengan nilai mAP@50 sebesar 0,988 serta mAP@50–95 sebesar 0,971.

Berdasarkan hasil evaluasi, model mampu mengenali mayoritas huruf SIBI dengan tingkat akurasi yang tinggi. Meski demikian, masih ditemukan huruf-huruf yang sulit dikenali secara konsisten, seperti N dan R, yang diduga disebabkan oleh kemiripan visual antar huruf. Untuk mengatasi waktu inferensi yang cukup tinggi pada model PyTorch (2072 ms/gambar), model dikonversi ke format ONNX yang mampu menurunkan waktu inferensi menjadi 797,7 ms/gambar dengan sedikit penurunan akurasi, yaitu mAP@50 sebesar 0,953 dan mAP@50–95 sebesar 0,953. Model ONNX kemudian diintegrasikan ke dalam perangkat lunak berbasis web dan berhasil mengenali alfabet SIBI secara *real-time*. Hasil pengujian menunjukkan sistem dapat berfungsi dengan baik dan memberikan respons yang efisien.

Penelitian ini menunjukkan potensi besar dalam mendukung komunikasi penyandang tunarungu dan tunawicara melalui teknologi *deep learning*. Ke depannya, pengembangan dapat difokuskan pada peningkatan kualitas data, pelatihan model dengan kondisi lingkungan lebih kompleks, serta perluasan sistem untuk mengenali gestur dinamis atau kalimat. Dengan penyempurnaan lanjutan, sistem ini diharapkan dapat menjadi solusi komunikasi yang inklusif dan praktis dalam kehidupan sehari-hari.

Daftar Referensi

- [1] I. M. Hamdani and N. S. Bustamin, "Pengo optimalan Komunikasi Bahasa Isyarat Abjad dengan Augmented Reality," *Jurnal Processor*, vol. 18, no. 2, pp. 144–152, Oct. 2023, doi: <https://doi.org/10.33998/processor.2023.18.2.872>.
- [2] R. Soekarta, S. Aras, R. Rezki, and N. Ainun K.D.P, "Sistem Deteksi Simbol Isyarat Pada SIBI (Sistem Isyarat Bahasa Indonesia) Menggunakan Mediapipe Berbasis Android," *Insect*

- (Informatics and Security): Jurnal Teknik Informatika, vol. 10, no. 2, pp. 90–100, Oct. 2024, doi: <https://doi.org/10.33506/insect.v10i2.4079>.
- [3] A. Rahagiyanto, "Identifikasi Ekstraksi Fitur untuk Gerakan Tangan dalam Bahasa Isyarat (SIBI) Menggunakan Sensor MYO Armband," MATRIK: Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer, vol. 19, no. 1, pp. 127–137, Nov. 2019, doi: 10.30812/matrik.v19i1.510.
- [4] A. Sofyan, H. Alwanto, and S. C. Arif, "Pengenalan Bahasa Isyarat Bahasa Indonesia Real-time Menggunakan Metode SP-Tree," Jurnal Indonesia: Manajemen Informatika dan Komunikasi, vol. 6, no. 2, pp. 1007–1014, May 2025, doi: <https://doi.org/10.63447/jimik.v6i2.1371>.
- [5] M. F. Golfantara, "Penggunaan Algoritma YOLO V8 untuk Identifikasi Rempah-Rempah," Jurnal Informatika dan Teknik Elektro Terapan, vol. 12, no. 3S1, pp. 3867–3873, Oct. 2024, doi: <https://doi.org/10.23960/jitet.v12i3s1.5221>.
- [6] A. Wang et al., "YOLOv10: Real-Time End-to-End Object Detection," arXiv.org, May 23, 2024. doi: <https://arxiv.org/abs/2405.14458>.
- [7] A. Wibowo, A. Sanjaya, and U. Mahdiyah, "Implementasi YOLOv8 Pada Pengenalan Sistem Isyarat Bahasa Indonesia," Prosiding SEMNAS INOTEK (Seminar Nasional Inovasi Teknologi), vol. 8, no. 1, pp. 139–146, 2024, doi: <https://doi.org/10.29407/inotek.v8i1.4920>.
- [8] N. Renaningtias, F. P. Utama, and A. N. A. Sobri, "View of Detection System Indonesian Sign Language (BISINDO) in Video with YOLOv7," JSAI: Journal Scientific and Applied Informatics, vol. 8, no. 1, pp. 1–8, Jan. 2025, Accessed: Jun. 23, 2025. [Online]. Available: <https://jurnal.umb.ac.id/index.php/JSAI/article/view/7067/4402>
- [9] B. A. Saputra, F. E. Putra, S. J. E. Lautt, V. H. Pranatawijaya, and R. Priskila, "Implementasi Algoritma YOLO VERSI 8 untuk Membaca Bahasa Isyarat," Jurnal Multidisiplin Inovatif, vol. 8, no. 4, pp. 195–205, 2025, Accessed: Jun. 23, 2025. [Online]. Available: <https://sejurnal.com/pub/index.php/jmi/article/view/1093/1277>
- [10] K. Aeni and A. S. Millah, "Implementasi Deteksi Objek dengan Model YOLOv8 pada Pengenalan Bahasa Isyarat," Smart Comp: Jurnalnya Orang Pintar Komputer, vol. 14, no. 1, pp. 244–255, Jan. 2025, doi: <https://doi.org/10.30591/smartcomp.v14i1.7342>.
- [11] D. Permana and J. Sutopo, "Aplikasi Pengenalan Abjad Sistem Isyarat Bahasa Indonesia (SIBI) dengan Algoritma YOLOv5," Jurnal Simantec, vol. 11, no. 2, pp. 231–240, Jul. 2023, doi: 10.21107/simantec.v11i2.19783.
- [12] M. K. H. Maulana, N. C. Wibowo, and A. R. E. Najaf, "Penerapan Algoritma K-Means untuk Segmentasi Daerah di Jawa Timur berdasarkan Indikator Kesejahteraan Masyarakat," Scientica: Jurnal Ilmiah Sains dan Teknologi, vol. 2, no. 11, pp. 21–52, Jul. 2024, Accessed: Jun. 23, 2025. [Online]. Available: <https://repository.upnjatim.ac.id/28037/>
- [13] Kuliahkomputer, "CRISP-DM," Kuliah Komputer Blogger, Jul. 12, 2018. <https://www.kuliahkomputer.com/2018/07/crisp-dm.html> (accessed Jun. 23, 2025).
- [14] SIBI, "Full220_signlanguage Detection 2 Dataset," Roboflow, 2024. https://universe.roboflow.com/sibi/full220_signlanguage-detection-2 (accessed Jun. 23, 2025).
- [15] Azzahra, "Pengaruh Augmentasi Data Pada Transliterasi Alfabet Bahasa Isyarat Indonesia (Bisindo) Berbasis Transfer Learning," Universitas Islam Negeri Ar-Raniry, 2025, <https://repository.ar-raniry.ac.id/id/eprint/42790/1/SKRIPSI%20AZ-ZAHRA.pdf>.