Vol. 11, No. 2, Oktober 2026, Page. 454-467
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

Pengenalan BISINDO Berbasis MediaPipe dengan Transformer Encoder, Long Short-Term Memory, Connectionist Temporal Classification, dan Text-to-Speech

Elvina Adellia[1], Firman Noor Hasan[*2]

[1,2]Teknik Informatika, Universitas Muhammadiyah Prof. Dr. Hamka, Indonesia
Email: [1]elvinadellia01@email.ac.id, [2]firman.noorhasan@uhamka.ac.id

Abstrak

Perbedaan pemahaman antara pengguna Bahasa Isyarat Indonesia (BISINDO) dan masyarakat yang belum menguasainya masih menghambat komunikasi. Penelitian ini bertujuan mengembangkan *prototipe* pengenalan 15 kosakata atau frasa dasar BISINDO secara *real-time* dengan keluaran teks dan suara. Dataset terdiri atas 4.500 *sequence* yang direkam oleh satu peraga isyarat, yaitu peneliti sendiri. Sistem menggunakan *MediaPipe* untuk mengambil *landmark* tangan dan tubuh. Fitur koordinat dan *velocity* disusun menjadi *sequence* 30 *frame*, kemudian diproses menggunakan model *Transformer Encoder* dan *LSTM*. Hasil prediksi diolah menggunakan *CTC* dan *Levenshtein Distance*, ditampilkan melalui *Streamlit*, dan diubah menjadi suara dengan *Text-to-Speech*. Evaluasi dilakukan pada 450 *sequence* yang juga digunakan sebagai data validasi. Hasilnya menunjukkan *accuracy* 95,56%, *precision* 95,53%, *recall* 95,57%, *F1-score* 95,55%, dan *CER* 4,00%. Pada pengujian *real-time*, akurasi mencapai 92,00% pada jarak ±100 cm, 92,67% pada pencahayaan terang, dan 92,67% pada posisi kamera lurus. Namun, akurasi turun menjadi 65,33% pada jarak jauh, 81,33% pada pencahayaan redup, dan 78,00% pada posisi serong. Hasil penelitian masih terbatas pada kosakata, peraga isyarat, dan kondisi pengujian yang digunakan.

Kata kunci: BISINDO, *Connectionist Temporal Classification, Long Short-Term Memory, MediaPipe Holistic, Text-to-Speech, Transformer Encoder*.

Abstract

Differences in understanding between Indonesian Sign Language (BISINDO) users and people unfamiliar with sign language *can hinder communication. This study developed a real-time prototype for recognizing 15 basic BISINDO words or short phrases and converting the results into text and speech. The dataset contains 4,500 sequences recorded by one person, the researcher. MediaPipe was used to extract hand and body landmarks. Coordinate and velocity features were arranged into 30-frame sequences and processed using a Transformer Encoder and LSTM model. Predictions were processed using CTC and Levenshtein Distance, displayed through Streamlit, and converted into speech using Text-to-Speech. Evaluation used 450 sequences that also served as validation data. The results showed 95.56% accuracy, 95.53% precision, 95.57% recall, 95.55% F1-score, and 4.00% CER. In real-time tests, accuracy reached 92.00% at a distance of ±100 cm, 92.67% under bright lighting, and 92.67% with a front-facing camera. Accuracy decreased to 65.33% at a longer distance, 81.33% under dim lighting, and 78.00% with an angled camera. These findings remain limited to the vocabulary, recording participant, and testing conditions used.*

Keywords: *BISINDO, Connectionist Temporal Classification, Long Short-Term Memory, MediaPipe Holistic, Text-to-Speech, Transformer Encoder.*

This work is an open access article and licensed under a Creative Commons Attribution-NonCommercial ShareAlike 4.0 International (CC BY-NC-SA 4.0)

1. PENDAHULUAN

Komunikasi merupakan kebutuhan utama agar setiap orang dapat saling memahami, termasuk teman Tuli yang menggunakan bahasa isyarat dalam aktivitas sehari-hari. Bahasa Isyarat Indonesia (BISINDO) banyak digunakan oleh komunitas Tuli sebagai media komunikasi berbasis gerakan tangan, tubuh, dan ekspresi. BISINDO berbeda dengan Sistem Isyarat Bahasa Indonesia (SIBI) karena BISINDO berkembang dari komunitas pengguna, sedangkan SIBI cenderung mengikuti struktur bahasa Indonesia formal [1]-[3].

Permasalahan muncul ketika lawan bicara tidak memahami bahasa isyarat. Kondisi ini dapat membuat komunikasi menjadi lambat dan berpotensi menimbulkan salah paham, terutama pada

pendidikan, layanan publik, dan situasi sosial yang membutuhkan respons cepat. Sistem berbasis *computer vision* mulai banyak digunakan karena kamera dapat membaca pola gerakan tanpa perangkat sensor tambahan[1], [2], [4], [5]. Namun, pengenalan BISINDO secara *real-time* masih menghadapi kendala, seperti pencahayaan, jarak pengguna, sudut kamera, kestabilan landmark, dan kemiripan gerakan antar kosakata [1], [2], [6].

Hasil wawancara awal dengan dua siswa tunarungu kelas XI dan satu guru di SLB Negeri 5 Jakarta menunjukkan bahwa BISINDO lebih sering dipilih karena dianggap mudah dipahami. Kendala utama muncul ketika lawan bicara tidak memahami isyarat, sehingga komunikasi biasanya dibantu dengan membaca gerak bibir atau mengetik pesan melalui ponsel. Guru juga menekankan bahwa ketepatan terjemahan lebih penting daripada kecepatan karena kesalahan terjemahan dapat mengubah makna pesan. Temuan ini menguatkan perlunya sistem penerjemah BISINDO yang dapat berjalan secara langsung, tetapi tetap memperhatikan kejelasan hasil terjemahan.

Beberapa penelitian sebelumnya telah mengembangkan sistem pengenalan bahasa isyarat menggunakan berbagai metode. Fauzi et al. mengembangkan pengenalan BISINDO menjadi suara menggunakan *MediaPipe* dan *SVM* [7]. Ahwani et al. menggunakan *MediaPipe* dan *CNN* untuk mengenali alfabet BISINDO dengan akurasi *real-time* sebesar 78,46% [2]. Sementara itu, Prayoga et al. menggunakan *MediaPipe* dan *LSTM* untuk mengenali 10 kosakata Kata Kolok dengan akurasi 95% [4]. Penelitian Tan et al. juga menerapkan *CTC* dan mekanisme *attention* untuk penerjemahan bahasa isyarat [8]. Penelitian internasional menunjukkan bahwa *Transformer* dan *CTC* dapat digunakan untuk mengenali urutan gerakan bahasa isyarat secara kontinu [9]. Namun, pendekatan berbasis *CTC* masih menghadapi kendala dalam pemrosesan secara online karena umumnya memerlukan rangkaian video lengkap [10]. Berbeda dengan penelitian tersebut, penelitian ini berfokus pada pengenalan 15 kosakata dasar BISINDO melalui kamera secara langsung dengan keluaran teks dan suara.

Tabel 1. Perbandingan Penelitian Terdahulu

Penelitian	Dataset & Kelas	Metode	Hasil Evaluasi	Keterbatasan
Therry et al [1]	BISINDO (26 kelas alfabet, A-Z)	*MediaPipe, CNN*	Evaluasi 94%, *real-time* 72%	Terbatas pada alfabet dan tangan kanan
Ahwani et al [2]	BISINDO (26 kelas alfabet, A-Z)	*MediaPipe, CNN*	*Real-time* 78,46%	Pengenalan alfabet masih dipengaruhi kondisi lingkungan
Prayoga et al [4]	Dataset Kata Kolok (10 kelas kosakata)	*MediaPipe, LSTM*	*Accuracy* 95%, *F1-score* 94%	Kosakata terbatas dan gerakan mirip sulit dibedakan
Fauzi et al [7]	BISINDO (2.730 data, 35 data/kelas, 20 *frame*/data)	*MediaPipe, SVM, TTS*	*Real-time* 78%	Data per kelas terbatas dan koreksi kesalahan masih manual
Tan et al [8]	*PHOENIX14T & CSL-Daily* (2.887 & 2.343 kosakata teks)	*Transformer, CTC, Attention*	*BLEU-4:* 28,42 dan 22,47	Berfokus pada penerjemahan kalimat
Penelitian ini	BISINDO (4.500 *sequence*, 15 kosakata)	*MediaPipe*, velocity, *Transformer, LSTM, CTC, Levenshtein Distance, TTS*	*Accuracy* 95,56%, *CER* 4%, *real-time* terbaik 92,67%	Terbatas pada 15 kosakata dari satu peraga isyarat, dan belum dilakukan *ablation study*

Berdasarkan Tabel 1, penelitian sebelumnya telah menggunakan berbagai metode untuk mengenali bahasa isyarat, tetapi memiliki perbedaan pada dataset, jumlah kelas, dan hasil evaluasi. Beberapa penelitian masih mengalami penurunan akurasi pada pengujian *real-time*, sedangkan penelitian lainnya berfokus pada pengenalan alfabet, kosakata terbatas, atau penerjemahan kalimat. Perbedaan tersebut

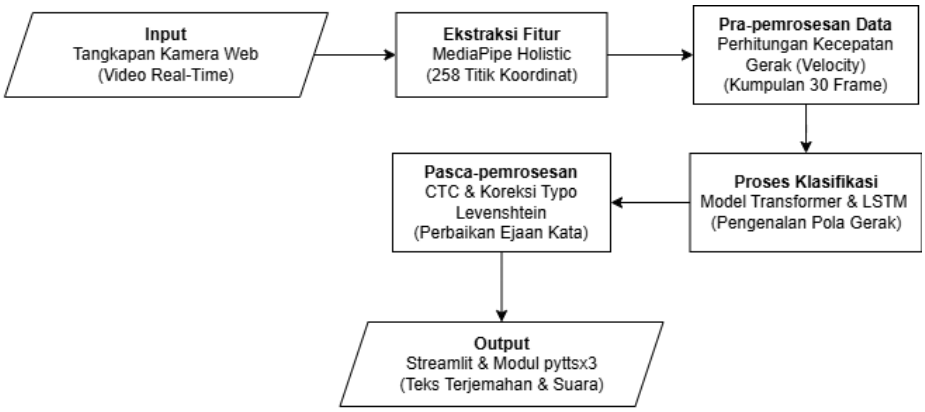
menunjukkan perlunya pengembangan sistem pengenalan BISINDO yang menggabungkan informasi posisi dan perubahan gerakan serta menghasilkan keluaran teks dan suara.

Dari penelitian-penelitian tersebut, masih terlihat beberapa celah yang perlu dikembangkan. Penelitian sebelumnya sudah banyak membahas pengenalan isyarat menggunakan *MediaPipe*, *CNN*, atau *LSTM*, tetapi sebagian besar masih berfokus pada klasifikasi gerakan atau hasil teks [1], [2], [4], [7], [11]-[15]. Belum banyak penelitian yang menggabungkan fitur koordinat dan *velocity*, pemodelan *temporal* menggunakan *Transformer Encoder* dan *LSTM*, perbaikan teks dengan *CTC* dan *Levenshtein Distance*, serta keluaran suara dalam satu alur BISINDO *real-time* [8], [16]-[22]. Selain itu, pengujian terhadap kondisi penggunaan seperti jarak, pencahayaan, dan sudut kamera juga masih perlu diperjelas karena kondisi kamera dapat memengaruhi kualitas *landmark* [1], [2], [23]. Oleh karena itu, penelitian ini mengembangkan alur penerjemah BISINDO yang tidak hanya mengenali gerakan, tetapi juga menstabilkan hasil teks dan mengubahnya menjadi suara.

Kebaruan penelitian ini terletak pada penggabungan fitur koordinat dan *velocity*, *Transformer Encoder*, *LSTM*, *CTC*, *Levenshtein Distance*, serta *Text-to-Speech* dalam satu sistem pengenalan BISINDO secara *real*-time. Kontribusi penelitian ini meliputi empat hal, yaitu: (1) menyusun dataset 15 kosakata BISINDO sebanyak 4.500 *sequence*, (2) membentuk input 516 fitur per *frame* dari gabungan koordinat dan *velocity*, (3) membangun sistem dari input kamera hingga keluaran teks dan suara, serta (4) menguji sistem berdasarkan variasi jarak, pencahayaan, dan sudut kamera. Penelitian ini masih terbatas pada pengenalan kosakata dasar dari satu peraga isyarat yang direkam oleh peneliti sendiri dan belum dilakukan pengujian terpisah terhadap setiap metode yang digunakan.

2. METODE PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif dengan menggunakan metode eksperimental dan pengembangan sistem. Pendekatan kuantitatif digunakan karena performa sistem diukur menggunakan *accuracy*, *F1-score*, *precision*, *recall*, *Character Error Rate (CER)*, dan pengujian *real-time* [1], [24]. Metode eksperimental digunakan karena sistem diuji pada beberapa kondisi penggunaan, yaitu jarak pengguna, pencahayaan, dan sudut kamera [1], [2], [6]. Secara umum, alur penelitian dimulai dari input video menggunakan *webcam*, ekstraksi *landmark* menggunakan *MediaPipe Holistic*, pembentukan fitur koordinat dan *velocity*, klasifikasi menggunakan model *hybrid Transformer Encoder* dan *LSTM*, *post-processing* menggunakan *CTC* dan *Levenshtein Distance*, kemudian hasil akhir ditampilkan dalam bentuk teks dan suara menggunakan *Text-to-Speech* melalui antarmuka *Streamlit* [5], [8], [16]-[23]. Gambar 1 menunjukkan alur pelaksanaan penelitian.



Gambar 1. Alur sistem penerjemah BISINDO

Gambar 1 menunjukkan bahwa sistem menerima input video *real-time* melalui *webcam*. Setiap *frame* video diproses untuk mengambil *landmark* tangan dan tubuh menggunakan *MediaPipe Holistic*. Hasil *landmark* kemudian diubah menjadi fitur koordinat dan *velocity* dalam bentuk *sequence* 30 *frame*. *Sequence* tersebut diproses oleh model *Transformer Encoder* dan *LSTM* untuk menghasilkan prediksi kosakata. Hasil prediksi kemudian dirapikan menggunakan *CTC* dan *Levenshtein Distance* sebelum ditampilkan sebagai teks dan diubah menjadi suara.

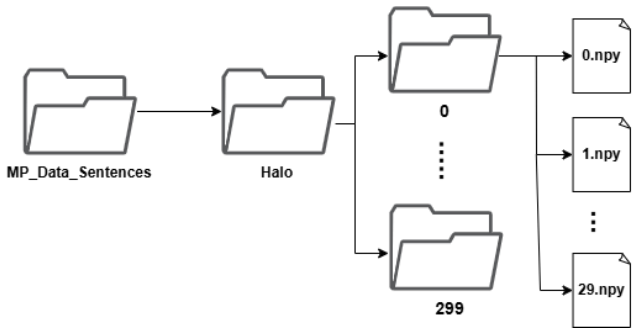
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

2.1. Dataset dan Struktur Penyimpanan

Dataset penelitian ini dikumpulkan dari satu peraga isyarat, yaitu peneliti sendiri. Seluruh kosakata BISINDO diperagakan oleh peneliti menggunakan *webcam* dengan resolusi 1280 × 720 piksel. Perekaman dilakukan dengan memperagakan 15 kosakata BISINDO secara berulang, yaitu "apa", "dimana", "halo", "hati-hati", "kamu", "maaf", "nama", "rumah", "sama-sama", "saya", "sekarang", "siapa", "teman", "terimakasih", dan "tolong". Setiap kosakata direkam sebanyak 300 *sequence* yang terdiri dari 30 *frame*, sehingga total dataset berjumlah 4.500 *sequence* atau 135.000 *frame*. Setiap *frame* diproses menggunakan *MediaPipe* untuk mengambil koordinat tangan dan tubuh, kemudian disimpan dalam format *NumPy(.npy)* [1], [4], [7], [11]-[15], [25].

Dataset dibagi secara acak berdasarkan *sequence* menggunakan *train_test_split* dengan perbandingan 90:10, yaitu 4.050 *sequence* untuk pelatihan dan 450 *sequence* untuk validasi sekaligus evaluasi model. Pembagian dilakukan tanpa pemisahan berdasarkan identitas peraga isyarat karena seluruh data berasal dari satu orang. Selain itu, penggunaan kembali lima *frame* terakhir memungkinkan adanya kesamaan data antar *sequence* yang berdekatan. Oleh karena itu, hasil evaluasi masih terbatas pada data dari peraga isyarat yang sama.

Data disimpan berdasarkan label kosakata agar proses pelatihan dan pengujian lebih terstruktur. Setiap folder label berisi 300 folder *sequence*, mulai dari *sequence* 0 sampai 299. Di dalam setiap *sequence* terdapat 30 file *frame* dalam format *NumPy*, yaitu *0.npy* sampai *29.npy*. Struktur penyimpanan dataset ditunjukkan pada Gambar 2.



Gambar 2. Struktur penyimpanan dataset

Gambar 2 menunjukkan dataset disusun berdasarkan label kosakata BISINDO. Setiap label memiliki folder *sequence*, kemudian setiap *sequence* berisi file frame hasil ekstraksi fitur. Struktur ini membantu menjaga keteraturan dataset dan mengurangi risiko kesalahan pemanggilan label saat proses pelatihan model.

Dataset pada penelitian ini masih terbatas pada 15 kosakata atau frasa dasar BISINDO. Jumlah tersebut cukup untuk menguji *prototipe* awal, tetapi belum mewakili percakapan BISINDO yang lebih beragam dan kompleks. Karena itu, hasil penelitian ini belum dapat digeneralisasi untuk semua bentuk komunikasi BISINDO. Pengembangan berikutnya dapat dilakukan dengan menambah jumlah kosakata, melibatkan lebih banyak subjek, menggunakan variasi lingkungan perekaman, serta menguji bentuk kalimat yang lebih panjang.

2.2. Ekstraksi Fitur dan Pra-pemrosesan

MediaPipe Holistic digunakan untuk mengekstraksi *landmark* tangan dan tubuh sebagai fitur utama dalam pengenalan gerakan BISINDO [5], [23]. *Landmark* wajah tidak digunakan karena penelitian ini berfokus pada 15 kosakata dasar yang lebih banyak dibedakan melalui gerakan tangan dan pose tubuh. Selain itu, penggunaan *landmark* wajah akan menambah jumlah fitur yang diproses, sehingga data menjadi lebih besar dan proses *real-time* menjadi lebih berat. Dengan menghilangkan *landmark* wajah, sistem dapat lebih fokus pada bagian tangan dan tubuh yang menjadi sumber gerakan utama. Namun, keputusan ini tetap menjadi keterbatasan karena sistem belum dirancang untuk mengenali isyarat yang sangat bergantung pada ekspresi wajah atau gerak bibir [6], [23]. Setiap tangan memiliki 21 titik *landmark* dengan tiga koordinat, yaitu x, y, dan z. Karena sistem membaca tangan kanan dan kiri, jumlah fitur tangan dihitung menggunakan Persamaan (1) dan Persamaan (2) [23].

p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

$$F_{kanan} = 21\ Landmark \times 3\ koordinat\,(x, y, z) = 63 \quad (1)$$

$$F_{kiri} = 21\ Landmark \times 3\ koordinat\,(x, y, z) = 63 \quad (2)$$

Selain tangan, sistem menggunakan 33 titik pose tubuh. Setiap titik pose memiliki nilai x, y, z, dan *visibility* [23]. Oleh karena itu, jumlah fitur pose tubuh dihitung menggunakan Persamaan (3).

$$F_{pose} = 33\ Landmark \times (3\ Koordinat + 1\ Visibility)(x, y, z, v) = 132 \quad (3)$$

Fitur koordinat tangan dan tubuh kemudian digabungkan menjadi 258 fitur koordinat. Jumlah tersebut dihitung menggunakan Persamaan (4).

$$Fitur\ koordinat\ =\ 126\ fitur\ tangan\ +\ 132\ fitur\ pose\ =\ 258 \quad (4)$$

Penelitian ini juga menggunakan fitur *velocity* agar model tidak hanya membaca posisi *landmark* pada satu *frame*. *Velocity* dihitung dari perubahan posisi *landmark* antar-*frame*, sehingga model dapat memperoleh informasi arah dan perubahan gerakan. Fitur ini dibutuhkan karena beberapa isyarat dapat memiliki bentuk tangan yang mirip, tetapi berbeda pada lintasan atau arah geraknya. Pada penelitian berbasis pose, *joint velocity* dan *bone velocity* digunakan untuk menangkap dinamika gerakan yang tidak terlihat dari koordinat statis saja [22]. Dengan demikian, penggunaan *velocity* pada penelitian ini bertujuan membantu model membaca pola gerakan secara lebih lengkap. Setelah fitur koordinat dan *velocity* digabungkan, jumlah fitur pada setiap *frame* menjadi 516 fitur, seperti ditunjukkan pada Persamaan (5) [22].

$$Fitur\ per\ frame\ =\ 258\ fitur\ koordinat\ +\ 258\ fitur\ velocity\ =\ 516 \quad (5)$$

Setiap input model terdiri dari 30 *frame*, sehingga bentuk input *sequence* yang masuk ke model ditunjukkan pada Persamaan (6).

$$Input\ sequence = 30\ frame \times 516\ fitur \quad (6)$$

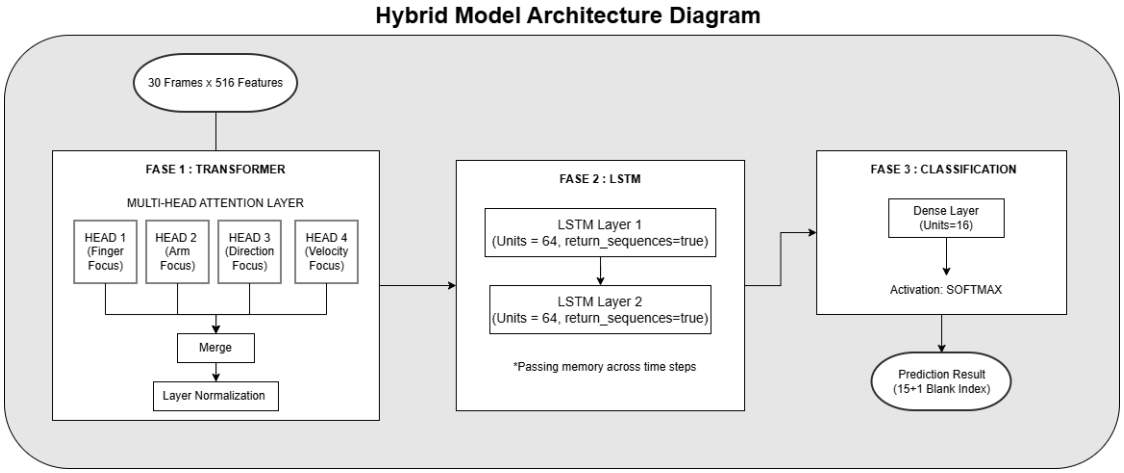
2.3. Pengembangan Model dan *Post-processing*

Model pada penelitian ini dikembangkan secara bertahap. Pada tahap awal, sistem menggunakan *MediaPipe Holistic* untuk mengambil *landmark* dan *LSTM* sebagai model utama. Tahap ini digunakan sebagai baseline awal karena *LSTM* banyak digunakan untuk membaca data gerakan yang berbentuk *sequence* [4], [12]-[15], [23]. Namun, dari pengujian awal, model *MediaPipe* dan *LSTM* saja masih kurang stabil pada beberapa kosakata yang memiliki pola gerakan mirip. Kesalahan sering muncul ketika bentuk tangan hampir sama, tetapi arah atau urutan gerakannya berbeda.

Setelah itu, *Transformer Encoder* ditambahkan sebelum *LSTM*. *Transformer Encoder* digunakan untuk membantu model memperhatikan bagian *frame* yang lebih penting melalui mekanisme *self-attention* [16], [17]. Dengan cara ini, model tidak hanya membaca urutan gerakan melalui *LSTM*, tetapi juga dapat memberi perhatian lebih pada bagian gerakan yang paling berpengaruh terhadap prediksi. Selain itu, fitur *velocity* juga ditambahkan agar model tidak hanya membaca posisi *landmark*, tetapi juga perubahan gerakan antar *frame* [22].

Pengujian awal juga dilakukan pada keluaran model sebelum menggunakan *post-processing*. Pada tahap tersebut, model sudah dapat mengenali kosakata, tetapi hasil teks masih dapat muncul berulang, kurang stabil, atau menghasilkan *blank output*. Oleh karena itu, *CTC* digunakan untuk merapikan keluaran model dengan menghapus *blank* dan menggabungkan prediksi yang berulang [8], [9], [18]. *Levenshtein Distance* kemudian digunakan sebagai pemeriksa akhir agar teks hasil prediksi tetap sesuai dengan daftar kosakata BISINDO yang digunakan [19]. Dengan demikian, baseline awal pada penelitian ini adalah *MediaPipe Holistic* dan *LSTM*, sedangkan sistem akhir menggunakan tambahan *Transformer Encoder*, fitur *velocity*, *CTC*, dan *Levenshtein Distance*. Arsitektur model ditunjukkan pada Gambar 3.

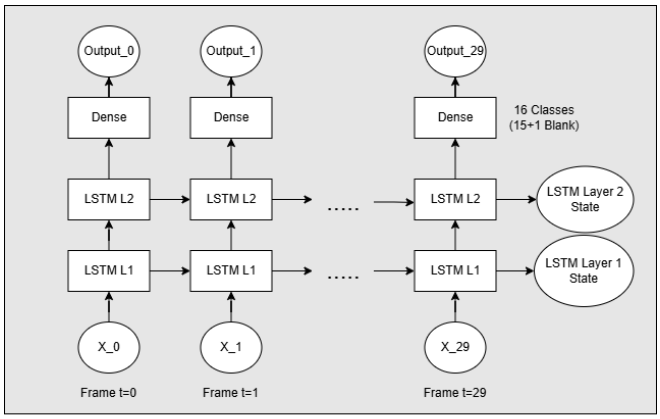
Vol. 11, No. 2, Oktober 2026, Page. 454-467
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI



Gambar 3. Arsitektur model *Transformer Encoder* dan *LSTM*

Gambar 3 menunjukkan arsitektur model yang menerima *input sequence* berupa 30 *frame* dengan 516 fitur pada setiap *frame*. Model menggunakan satu blok *Multi-Head Attention* dengan empat *attention head* dan dimensi kunci (*key dimension*) 64, kemudian dilanjutkan dengan *residual connection* dan *Layer Normalization* [9], [16]. Hasilnya diteruskan ke dua *layer* LSTM yang masing-masing memiliki 64 unit dan menggunakan *return_sequences=True* untuk membaca urutan gerakan [4], [12]-[15]. Selanjutnya, *Dense layer* dengan aktivasi *softmax* menghasilkan prediksi terhadap 16 kelas, yaitu 15 kosakata BISINDO dan satu kelas *blank* untuk kebutuhan CTC [8], [9].

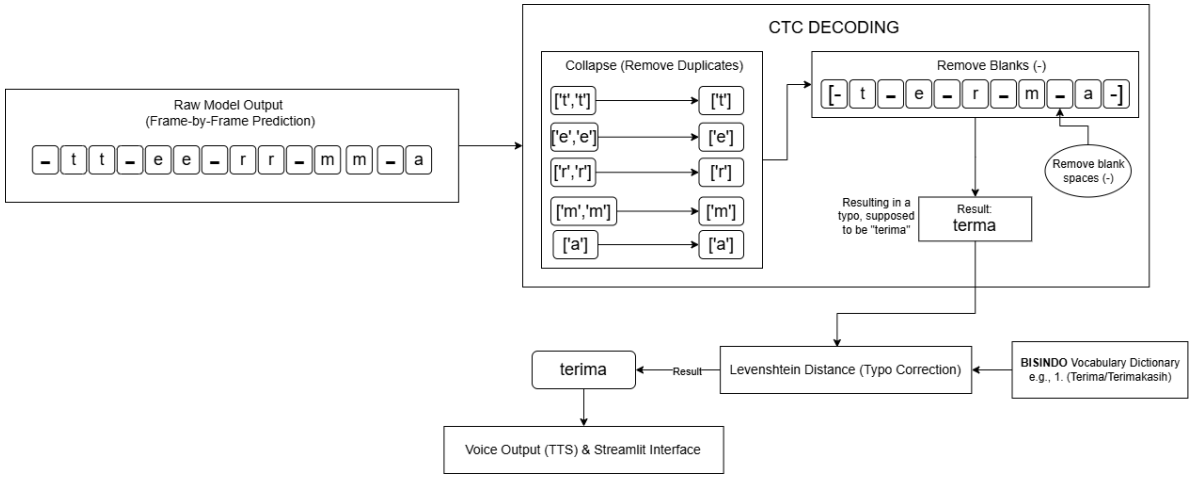
Pelatihan model menggunakan *optimizer* Adam dengan *learning rate* 0,001, *batch size* 32, dan maksimal 50 *epoch*. Proses pelatihan menggunakan *EarlyStopping* untuk menghentikan pelatihan apabila *validation loss* tidak membaik selama 15 *epoch*, kemudian mengembalikan bobot model terbaik. Dataset dibagi secara acak dengan *random_state=42* menggunakan perbandingan 90:10, yaitu 4.050 *sequence* untuk pelatihan dan 450 *sequence* untuk validasi sekaligus evaluasi.



Gambar 4. Proses *LSTM* dalam membaca *sequence*

Gambar 4 menunjukkan proses *LSTM* dalam membaca *sequence* dari *frame* ke-0 sampai *frame* ke-29. Setiap *frame* direpresentasikan sebagai *vektor* fitur X_0, X_1, hingga X_{29}, dengan masing-masing berisi 516 fitur. Pada *Layer 1*, setiap *frame* diproses secara berurutan dan menghasilkan *hidden state*. Hasil dari *Layer 1* kemudian diteruskan sebagai input ke *Layer 2* untuk memperkuat pola gerakan. Setelah itu, hasil akhir dari *LSTM* masuk ke *dense layer*. *Dense layer* digunakan untuk mengubah fitur hasil pembelajaran menjadi nilai prediksi kelas. Pada penelitian ini, *dense layer* menggunakan aktivasi *softmax* untuk menghasilkan prediksi terhadap 16 kelas, yaitu 15 kosakata BISINDO dan satu kelas *blank* untuk kebutuhan *CTC* [4], [8], [12]-[15].

Vol. 11, No. 2, Oktober 2026, Page. 454-467
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI



Gambar 5. Proses CTC decoding dan koreksi teks

Gambar 5 menunjukkan proses pengolahan hasil prediksi menggunakan *CTC dan Levenshtein Distance*. Pada tahap pelatihan, *CTC* digunakan sebagai fungsi *loss* untuk membantu model mempelajari hubungan antara urutan gerakan dan label kosakata [8], [9]. Selanjutnya, *CTC decoding* digunakan untuk menghapus *blank output* dan menggabungkan prediksi yang berulang [8], [9]. Hasil prediksi kemudian diperiksa menggunakan *Levenshtein Distance* dengan membandingkan kata terhadap daftar 15 kosakata BISINDO dan memilih kata yang memiliki jarak karakter paling kecil [19]. Jika hasil prediksi tidak sesuai dengan kelas kosakata yang tersedia, sistem tidak menghasilkan terjemahan. Dengan demikian, *CTC* digunakan untuk mengolah urutan prediksi, sedangkan *Levenshtein Distance* digunakan sebagai pemeriksaan akhir agar teks tetap sesuai dengan kosakata yang digunakan dalam penelitian. Hasil prediksi yang telah diproses kemudian ditampilkan sebagai teks melalui antarmuka *Streamlit* dan diubah menjadi suara menggunakan *Text-to-Speech* untuk membantu komunikasi dengan masyarakat yang belum memahami BISINDO [20], [21].

2.4. Pengujian *Real-Time* dan Metrik Evaluasi

Pengujian *real-time* dilakukan menggunakan 15 kosakata BISINDO yang telah dilatih pada model. Setiap kosakata diuji sebanyak 10 kali pada setiap kondisi, sehingga jumlah pengujian pada setiap kondisi adalah 150 percobaan. Skenario pengujian meliputi variasi jarak, pencahayaan, dan sudut kamera karena kondisi tersebut dapat memengaruhi kualitas deteksi gerakan pada sistem [1], [2], [6], [23]. Rincian skenario ditunjukkan pada Tabel 2.

Tabel 2. Skenario pengujian real-time

Skenario	Kondisi	Total Test
Variasi Jarak	±50 cm, ±100 cm, ±150 cm	150 tests
Variasi Pencahayaan	Terang dan Redup	150 tests
Variasi Posisi	Lurus dan Serong	150 tests

Evaluasi model dilakukan menggunakan *confusion matrix*, *accuracy*, *F1-score*, *recall*, *precision*, dan *CER*. *Confusion matrix* digunakan untuk mengidentifikasi jumlah prediksi yang benar maupun salah pada masing-masing kelas, Adapun *accuracy*, *recall*, *F1-score*, dan *precision* dimanfaatkan untuk menilai performa model dalam melakukan klasifikasi [1], [14], [15]. *CER* digunakan untuk mengukur tingkat kesalahan karakter pada teks hasil prediksi, terutama setelah proses *CTC decoding* dan koreksi teks [8], [18], [19], [24]. Rumus evaluasi ditunjukkan pada Persamaan (7) sampai Persamaan (11).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (7)$$

$$Precision = \frac{TP}{TP+FP} \quad (8)$$

p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

$$Recall = \frac{\text{TP}}{\text{TP+FN}} \quad (9)$$

$$F1\text{-}score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \quad (10)$$

$$CER = \frac{S+D+I}{N} \quad (11)$$

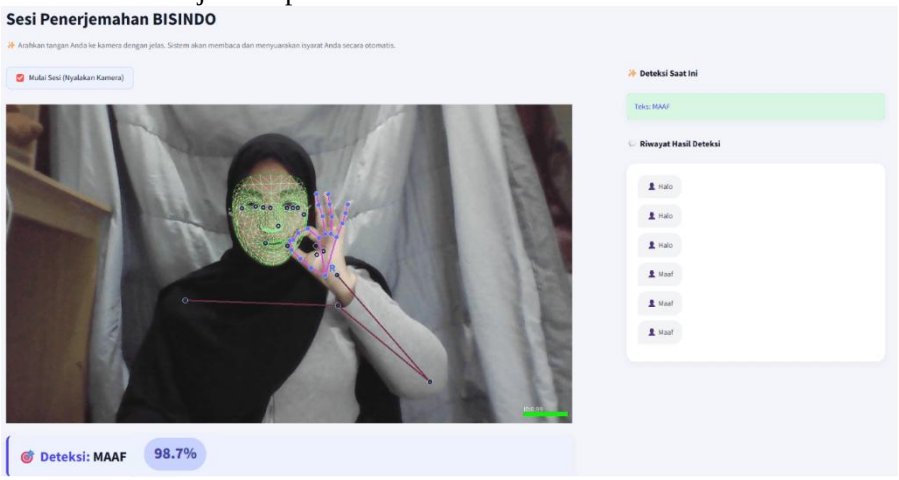
Pada Persamaan (7) sampai Persamaan (10), *TP* menunjukkan *True Positive*, *TN* menunjukkan *True Negative*, *FP* menunjukkan *False Positive*, sedangkan *FN* menunjukkan *False Negative*. Pada Persamaan (11), *S* adalah *substitution*, *D* adalah *deletion*, *I* adalah *insertion*, dan *N* adalah jumlah karakter pada teks referensi. Untuk pengujian *real-time*, akurasi dihitung dari jumlah prediksi benar dibandingkan dengan total percobaan. Misalnya, pada jarak normal terdapat 138 prediksi benar dari 150 percobaan, sehingga perhitungannya ditunjukkan pada Persamaan (12) [1], [24].

$$Accuracy\ real\text{-}time = \frac{138}{150} \text{ x } 100\% = 92{,}00\% \quad (12)$$

3. HASIL DAN PEMBAHASAN

3.1 Hasil Implementasi Sistem

Sistem penerjemah BISINDO berhasil dibangun dengan alur *input* kamera, ekstraksi *landmark MediaPipe*, *pra-pemrosesan* fitur, klasifikasi menggunakan *Transformer Encoder* dan *LSTM*, *CTC*, koreksi *Levenshtein Distance*, antarmuka *Streamlit*, dan *Text-to-Speech*. Tampilan sistem dan contoh hasil ekstraksi *landmark* ditunjukkan pada Gambar 6.



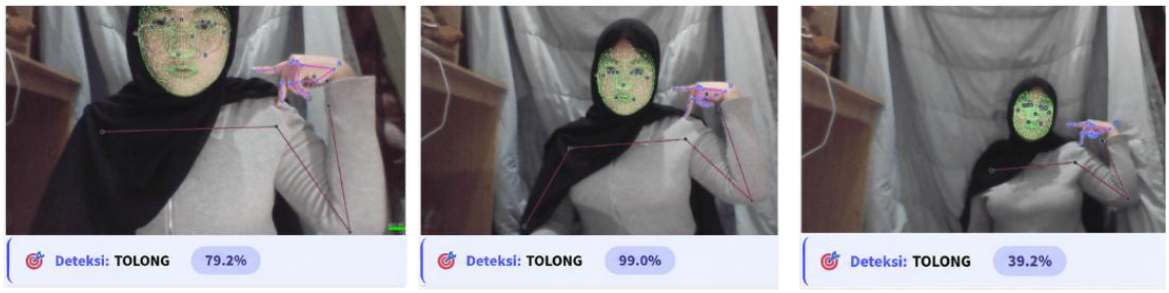
Gambar 6. Tampilan sistem dan contoh ekstraksi *landmark*

Gambar 6 menunjukkan bahwa sistem dapat menerima gerakan secara *real-time*, menampilkan hasil prediksi dalam bentuk teks, menyimpan riwayat hasil deteksi, dan menghasilkan keluaran suara. Hasil ini menunjukkan bahwa alur sistem sudah berjalan dari *input* visual sampai *output* teks dan suara. Namun, hasil implementasi ini belum digunakan untuk menyatakan bahwa metode yang digunakan lebih unggul dari semua metode lain. Implementasi ini lebih tepat dibaca sebagai bukti bahwa *pipeline* yang diusulkan dapat berjalan dan diuji pada kosakata serta kondisi yang sudah ditentukan.

3.2 Hasil Pengujian Berdasarkan Jarak

Pengujian jarak dilakukan untuk mengetahui pengaruh posisi pengguna terhadap kualitas deteksi *landmark*. Kondisi yang diuji adalah jarak dekat ±50 cm, jarak normal ±100 cm, dan jarak jauh ±150 cm. Contoh pengujian berdasarkan jarak ditunjukkan pada Gambar 7.

Vol. 11, No. 2, Oktober 2026, Page. 454-467
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI



Gambar 7. Contoh pengujian kata ”Tolong” berdasarkan variasi jarak

Gambar 7 menunjukkan bahwa jarak pengguna memengaruhi ukuran tangan dan tubuh yang terbaca oleh kamera. Pada jarak dekat, tangan terlihat lebih besar, tetapi beberapa bagian gerakan dapat terlalu dekat dengan kamera. Pada jarak normal, tangan dan tubuh terlihat lebih proporsional sehingga *landmark* dapat terbaca lebih stabil. Pada jarak jauh, ukuran tangan terlihat lebih kecil dan beberapa detail gerakan menjadi lebih sulit ditangkap oleh sistem.

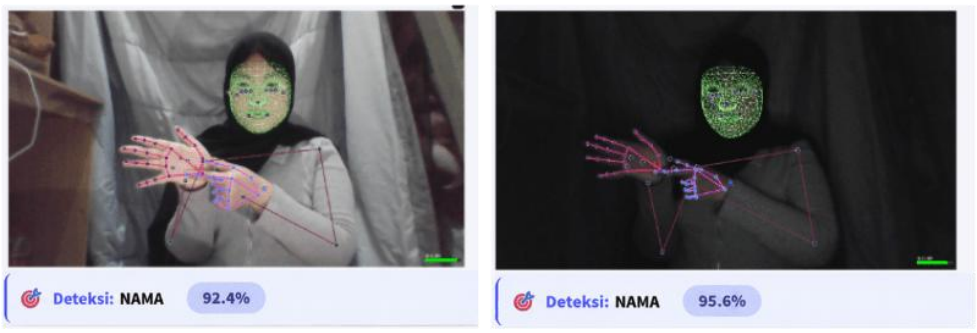
Tabel 3. Hasil pengujian berdasarkan jarak

Kondisi Jarak	Prediksi Benar	Total Test	Akurasi
Jarak Dekat (±50 cm)	130	150	86,67%
Jarak Normal (±100 cm)	138	150	92,00%
Jarak Jauh (±150 cm)	98	150	65,33%

Berdasarkan Tabel 3, hasil terbaik diperoleh pada jarak normal ±100 cm dengan akurasi 92,00%. Kondisi ini menunjukkan bahwa posisi pengguna pada jarak normal membuat tangan dan tubuh terlihat lebih proporsional oleh kamera. Pada jarak dekat, akurasi turun menjadi 86,67% karena beberapa gerakan tangan dapat terlalu dekat dengan kamera sehingga posisi *landmark* tidak selalu terbaca stabil. Penurunan paling besar terjadi pada jarak jauh ±150 cm dengan akurasi 65,33%. Hal ini terjadi karena ukuran tangan dan tubuh menjadi lebih kecil pada *frame*, sehingga detail gerakan lebih sulit ditangkap. Dengan demikian, jarak pengguna berpengaruh langsung terhadap kualitas *landmark* yang diterima model.

3.3 Hasil Pengujian Berdasarkan Pencahayaan

Pengujian pencahayaan dilakukan untuk mengetahui pengaruh cahaya terhadap hasil deteksi gerakan. Kondisi yang diuji adalah pencahayaan terang dan redup. Contoh pengujian pencahayaan ditunjukkan pada Gambar 8.



Gambar 8. Contoh pengujian kata ”Nama” berdasarkan pencahayaan

Gambar 8 menunjukkan bahwa pada pencahayaan terang, bentuk tangan dan tubuh terlihat lebih jelas sehingga titik *landmark* dapat terbaca lebih stabil. Pada pencahayaan redup, kualitas gambar

menurun dan beberapa titik *landmark* menjadi kurang konsisten. Kondisi ini berpengaruh pada hasil prediksi karena model bergantung pada fitur *landmark* yang diambil dari kamera.

Tabel 4. Hasil pengujian berdasarkan pencahayaan

Kondisi Pencahayaan	Prediksi Benar	Total Test	Akurasi
Kondisi Terang	139	150	92,67%
Kondisi Redup	122	150	81,33%

Berdasarkan Tabel 4, sistem memperoleh akurasi 92,67% pada pencahayaan terang dan 81,33% pada pencahayaan redup. Perbedaan ini menunjukkan bahwa pencahayaan berpengaruh terhadap kestabilan pembacaan *landmark*. Pada kondisi terang, bentuk tangan dan tubuh terlihat lebih jelas sehingga titik *landmark* lebih mudah dikenali. Sebaliknya, pada kondisi redup, sebagian detail tangan kurang terlihat dan posisi *landmark* menjadi kurang stabil. Kondisi ini membuat fitur yang masuk ke model tidak sejelas pada pencahayaan terang, sehingga hasil prediksi ikut menurun.

3.4 Hasil Pengujian Berdasarkan Sudut Kamera

Pengujian sudut kamera dilakukan untuk melihat pengaruh posisi tubuh terhadap hasil deteksi. Kondisi yang diuji adalah posisi lurus menghadap kamera dan posisi serong. Contoh pengujian sudut kamera ditunjukkan pada Gambar 9.



Gambar 9. Contoh pengujian kata ”Saya” berdasarkan sudut kamera

Gambar 9 menunjukkan bahwa pada posisi lurus, tangan dan tubuh terlihat lebih lengkap oleh kamera. Pada posisi serong, sebagian tangan atau tubuh dapat tertutup, sehingga fitur yang diterima model menjadi kurang lengkap. Kondisi ini dapat menyebabkan hasil prediksi menurun karena sistem membutuhkan *landmark* tangan dan tubuh yang terlihat jelas.

Tabel 5. Hasil pengujian berdasarkan sudut kamera

Sudut Kamera	Prediksi Benar	Total Test	Akurasi
Posisi Lurus	139	150	92,67%
Posisi Serong	117	150	78,00%

Berdasarkan Tabel 5, sistem memperoleh akurasi 92,67% pada posisi lurus dan 78,00% pada posisi serong. Penurunan pada posisi serong terjadi karena sebagian tangan atau tubuh tidak selalu terlihat utuh oleh kamera. Selain itu, arah gerakan dapat terlihat berbeda ketika pengguna tidak menghadap kamera secara langsung. Kondisi ini membuat fitur *landmark* yang diterima model menjadi kurang lengkap, sehingga prediksi lebih mudah bergeser ke kelas lain.

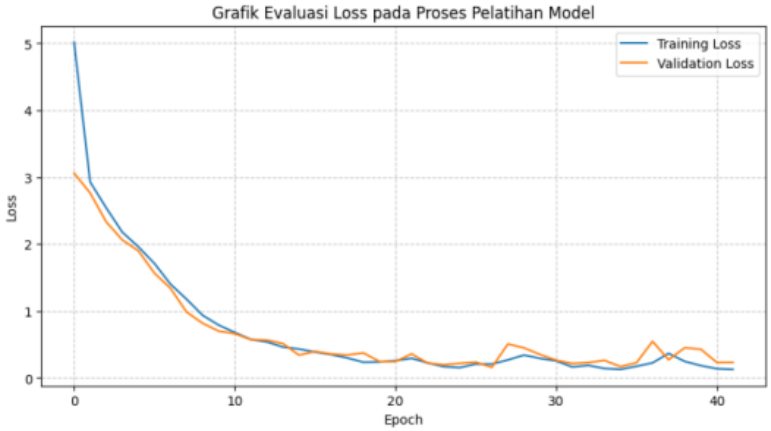
Dari pengujian jarak, pencahayaan, dan sudut kamera, terlihat bahwa sistem bekerja paling baik ketika tangan dan tubuh terlihat jelas oleh kamera. Kondisi terbaik muncul pada jarak normal, pencahayaan terang, dan posisi lurus. Sebaliknya, performa menurun ketika gerakan sulit dibaca, seperti pada jarak jauh, pencahayaan redup, dan posisi serong. Hasil tersebut menunjukkan bahwa performa

Vol. 11, No. 2, Oktober 2026, Page. 454-467
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

sistem turut dipengaruhi oleh kualitas *input* dari kamera dan posisi pengguna, bukan hanya oleh kemampuan model.

3.5 Hasil Evaluasi Model

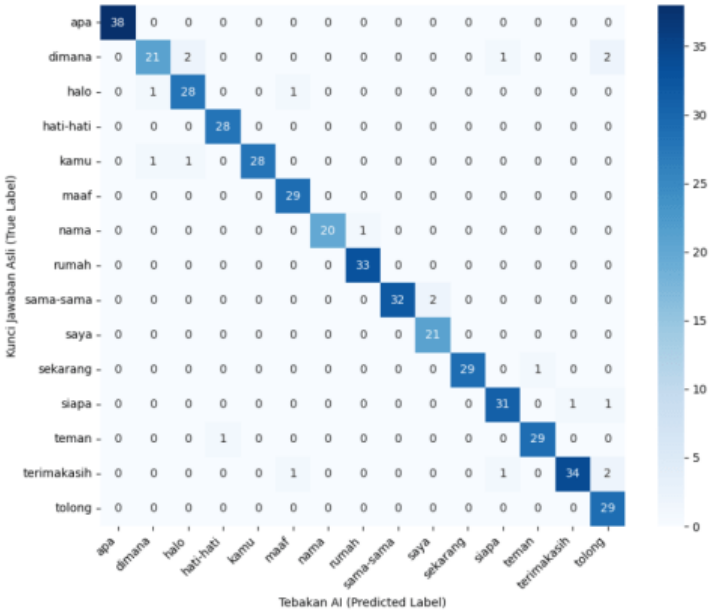
Evaluasi model dilakukan untuk mengetahui kemampuan sistem dalam mengenali kosakata BISINDO. Grafik *training loss* dan *validation loss* digunakan untuk mengamati perubahan nilai kesalahan selama pelatihan. Pada penelitian ini, 90% dataset digunakan untuk pelatihan, sedangkan 10% sisanya digunakan untuk validasi sekaligus evaluasi model. Hasil grafik ditampilkan pada Gambar 10.



Gambar 10. Hasil *training loss* dan *validation loss*

Gambar 10 menunjukkan bahwa nilai *training loss* dan *validation loss* menurun pada *epoch* awal, kemudian mulai stabil pada nilai kesalahan yang rendah. Selisih kedua nilai tersebut relatif kecil, sehingga tidak terlihat perbedaan yang besar antara hasil pelatihan dan validasi. Namun, data validasi juga digunakan untuk menentukan bobot model terbaik dan menghitung hasil evaluasi akhir. Hal ini menjadi keterbatasan penelitian karena pengujian belum menggunakan data yang sepenuhnya terpisah, sehingga hasil akurasi berpotensi lebih tinggi dibandingkan pengujian pada data baru.

Confusion matrix digunakan untuk mengevaluasi hasil prediksi model pada masing-masing kelas. Hasil *confusion matrix* disajikan pada Gambar 11.



Gambar 11. Hasil *confusion matrix*

Gambar 11 menunjukkan bahwa sebagian besar kosakata berhasil dikenali dengan benar. Dari 450 data uji, terdapat 430 prediksi benar. Beberapa kosakata seperti "apa", "hati-hati", "rumah", dan "saya" memperoleh hasil yang baik karena pola gerakannya lebih mudah dibedakan dari kelas lain.

Namun, *confusion matrix* juga menunjukkan bahwa masih ada kesalahan pada beberapa kosakata yang memiliki pola gerakan mirip. Misalnya, kosakata "dimana" beberapa kali diprediksi sebagai "halo" atau "tolong". Kesalahan seperti ini dapat terjadi karena kemiripan bentuk tangan, posisi awal gerakan, atau arah gerakan pada beberapa *frame*. Jika *landmark* tidak terbaca stabil, model dapat menerima fitur yang kurang lengkap dan memilih kelas lain yang dianggap paling mirip. Pola *error* ini menunjukkan bahwa batas kemampuan model masih terlihat pada kosakata yang memiliki bentuk tangan atau lintasan gerak yang hampir sama.

Tabel 6. Hasil evaluasi model

Metrik	Hasil
Accuracy	95,56%
F1-score	95,55%
Recall	95,57%
Precision	95,53%
Character Error Rate (CER)	4,00%

Berdasarkan Tabel 6, model memperoleh *Accuracy* 95,56%, *F1-score* 95,55%, *recall* 95,57%, *precision* 95,53%, dan *Character Error Rate* (CER) 4,00%. Hasil ini menunjukkan bahwa sistem mampu mengenali sebagian besar kosakata BISINDO pada data evaluasi.

Perhitungan *CER* dilakukan menggunakan *Levenshtein Distance* dengan membandingkan kata hasil prediksi dan kata sebenarnya [19], [24]. Kesalahan karakter dihitung berdasarkan jumlah penggantian (*substitution*), penghapusan (*deletion*), dan penambahan karakter (*insertion*). Jumlah kesalahan tersebut kemudian dibagi dengan jumlah karakter pada kata sebenarnya. Sebagai contoh, jika kata "dimana" diprediksi menjadi "diman", terdapat satu karakter yang hilang dari enam karakter, sehingga nilai *CER* adalah 1/6 = 0,1667 atau 16,67%. Contoh lainnya, kata "terimakasih" yang diprediksi menjadi "terrimakasih" memiliki satu karakter tambahan dari sebelas karakter, sehingga nilai *CER* adalah 1/11 = 0,0909 atau 9,09%.

Pada penelitian ini, perhitungan dilakukan terhadap 450 data evaluasi. Nilai *CER* dihitung pada setiap kata, kemudian seluruh hasilnya dijumlahkan dan dibagi dengan jumlah data evaluasi untuk memperoleh nilai rata-rata. Berdasarkan hasil evaluasi program, diperoleh nilai rata-rata *CER* sekitar 0,040 atau 4,00%. Nilai tersebut menunjukkan bahwa kesalahan karakter pada teks hasil prediksi relatif rendah.

Namun, hasil akurasi ini belum dapat digunakan untuk menyatakan bahwa metode yang digunakan lebih unggul dari metode lainnya karena penelitian belum melakukan pengujian perbandingan, seperti model tanpa *Transformer*, tanpa *velocity*, atau tanpa *post-processing*. Oleh karena itu, hasil evaluasi masih terbatas pada dataset dan skenario pengujian yang digunakan.

Berdasarkan hasil pengujian, sistem memperoleh akurasi sebesar 95,56% pada data evaluasi. Namun, pada pengujian *real-time*, hasilnya lebih rendah ketika kondisi pengambilan gambar kurang ideal, yaitu 65,33% pada jarak jauh, 81,33% pada pencahayaan redup, dan 78,00% pada posisi serong. Perbedaan ini dapat terjadi karena perubahan jarak, pencahayaan, dan sudut kamera memengaruhi pembacaan titik *landmark*. Selain itu, dataset hanya direkam oleh peneliti sendiri sehingga variasi gerakan pengguna lain dan kondisi lingkungan yang berbeda belum terwakili. Hal ini menjadi keterbatasan penelitian karena kemampuan sistem pada kondisi penggunaan yang lebih beragam belum dapat dipastikan.

4. KESIMPULAN

Penelitian ini berhasil mengembangkan prototipe pengenalan BISINDO secara *real-time* yang mengubah gerakan isyarat menjadi teks dan suara. Sistem menggunakan *MediaPipe* untuk mengekstraksi *landmark* tangan dan tubuh, kemudian menggabungkan fitur koordinat dan *velocity* dengan model *Transformer Encoder* dan *LSTM*. Hasil prediksi diproses menggunakan *CTC* dan *Levenshtein Distance* sebelum ditampilkan sebagai teks dan diubah menjadi suara melalui *Text-to-*

Speech. Hasil evaluasi menunjukkan *Accuracy* 95,56%, *precision* 95,53%, *recall* 95,57%, *F1-score* 95,55%, dan *CER* 4,00%. Pada pengujian *real-time*, hasil terbaik diperoleh pada jarak ±100 cm sebesar 92,00%, pencahayaan terang 92,67%, dan posisi kamera lurus 92,67%.

Kontribusi utama penelitian ini terletak pada penggabungan beberapa komponen dalam satu alur sistem, yaitu ekstraksi *landmark*, fitur *velocity*, model *hybrid Transformer Encoder* dan *LSTM*, *post-processing* teks, serta keluaran suara. Dengan alur tersebut, sistem tidak hanya mengenali gerakan isyarat, tetapi juga menghasilkan teks yang kemudian diubah menjadi suara untuk membantu komunikasi dengan masyarakat yang belum memahami BISINDO.

Meskipun hasil pengujian menunjukkan kinerja yang baik, penelitian ini masih terbatas pada 15 kosakata atau frasa dasar BISINDO dengan dataset dari satu peraga isyarat. Oleh karena itu, sistem belum dapat digunakan untuk menerjemahkan percakapan BISINDO secara bebas dan belum diuji pada pengguna baru. Selain itu, penelitian ini belum melakukan *ablation study*, sehingga pengaruh setiap komponen terhadap kinerja model belum dapat diketahui secara terpisah. Pengembangan selanjutnya dapat dilakukan dengan menambah kosakata, melibatkan lebih banyak peraga isyarat, serta melakukan pengujian pada data dan kondisi lingkungan yang lebih beragam.

5. DAFTAR PUSTAKA

[1] R. W. L. Therry, A. Junaidi, and A. N. Sihananto, "Program Penerjemah Bahasa Isyarat Indonesia (BISINDO) Secara Real Time Menggunakan Convolutional Neural Network Dan Mediapipe," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 6, pp. 11642–11649, 2024, doi: 10.36040/jati.v8i6.11582.

[2] D. Ahwani, S. Budiman, and M. Rahmat, "Penerapan Mediapipe dalam Pengenalan Bisindo Berbasis Deep Learning dan Computer Vision (Studi Kasus: SLB-C B Yayasan Pendidikan Luar Biasa (YPLB) Blitar)," *INNOVATIVE: Journal Of Social Science Research*, vol. 4, no. 5, pp. 3608–3628, 2024, doi: 10.31004/innovative.v4i5.15199.

[3] A. Setiawan and F. N. Hasan, "Analisis Sentimen Tanggapan Pengguna Aplikasi Bale By BTN Menggunakan Metode Support Vector Machine (SVM)," *Jurnal Ilmiah Teknik dan Ilmu Komputer*, vol. 4, no. 4, pp. 315–326, 2025, doi: 10.55123/storage.v4i4.6469.

[4] M. P. Prayoga, G. A. A. D. Indradewi, and P. H. Suputra, "Pengenalan Kata Kolok Secara Real Time Menggunakan Mediapipe dan Algoritma Long Short-Term Memory (LSTM)," *Sintech Journal*, vol. 8, no. 2, pp. 178–187, 2025, doi: 10.31598/sintechjournal.v8i2.1945.

[5] S. Nur Budiman, S. Lestanti, S. Marselius Evvandri, and R. Kartika Putri, "Pengenalan Gestur Gerakan Jari Untuk Mengontrol Volume Di Komputer Menggunakan Library Opencv Dan Mediapipe," *Antivirus : Jurnal Ilmiah Teknik Informatika*, vol. 16, no. 2, pp. 223–232, Nov. 2022, doi: 10.35457/antivirus.v16i2.2508.

[6] Hajrah, A. azis, and I. Rahim, "Analisis Penggunaan Bahasa Isyarat Indonesia (BISINDO) pada Siswa SLB," *Jurnal Onoma: Pendidikan, Bahasa dan Sastra*, vol. 9, no. 2, pp. 1396–1402, 2023, doi: 10.30605/onoma.v9i2.2717.

[7] M. Z. Fauzi, R. Sarno, and S. C. Hidayati, "Recognition of Real-Time BISINDO Sign Language-to-Speech using Machine Learning Methods," International Conference on Computer Science, Information Technology and Engineering (ICCoSITE), 2023, pp. 986–991. doi: 10.1109/ICCoSITE57641.2023.10127743.

[8] S. Tan, T. Miyazaki, N. Khan, and K. Nakadai, "Improvement in Sign Language Translation Using Text CTC Alignment," in *ACL Anthology*, Abu Dhabi, UAE: International Committee on Computational Linguistics (ICCL), 2025, pp. 3255–3266. Accessed: Dec. 15, 2025. [Online]. Available: https://aclanthology.org/2025.coling-main.219/

[9] K. Papadimitriou and G. Potamianos, "Multimodal Locally Enhanced Transformer for Continuous Sign Language Recognition," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2023-August, pp. 1513–1517, 2023, doi: 10.21437/Interspeech.2023-2198.

[10] R. Zuo, F. Wei, and B. Mak, "Towards Online Continuous Sign Language Recognition and Translation," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language*

Processing, Stroudsburg, PA, USA: Association for Computational Linguistics, 2024, pp. 11050–11067. doi: 10.18653/v1/2024.emnlp-main.619.

[11] D. P. Kartaputra *et al.*, "Deteksi Alfabet BISINDO Menggunakan Mediapipe Holistic Secara Real-Time," *Jurnal Teknologi Informasi dan Komunikasi*, vol. 12, no. 1, pp. 46–52, Jun. 2023, doi: 10.58761/jurtikstmikbandung.v12.i1.113.

[12] H. M. Putri, Fadlisyah, and W. Fuadi, "Pendeteksian Bahasa Isyarat Indonesia Secara Real-Time Menggunakan Long Short-Term Memory (Lstm)," *Jurnal Teknologi Terapan & Sains*, vol. 3, no. 1, pp. 663–675, 2022, doi: 10.29103/tts.v3i1.6853.

[13] E. Altiarika and W. P. Sari, "Pengembangan Deteksi Realtime untuk Bahasa Isyarat Indonesia dengan Menggunakan Metode Deep Learning Long Short Term Memory dan Convolutional Neural Network," *Jurnal Teknologi Informatika dan Komputer*, vol. 9, no. 1, pp. 1–13, Mar. 2023, doi: 10.37012/jtik.v9i1.1272.

[14] C. A. Sari, E. H. Rachmawanto, Z. Saifullah, C. Jatmoko, and D. Sinaga, "Real-time detection of indonesian sign language (ISL) gestures based on long short-term memory," *Journal of Soft Computing Exploration*, vol. 5, no. 3, pp. 251–262, Sep. 2024, doi: 10.52465/joscex.v5i3.452.

[15] P. B. Utomo, R. A. Ramadhani, and H. Kurniawan, "Deteksi Gerak Tangan sebagai Pengenal Bahasa Isyarat menggunakan Mediapipe dan Long-Short Term Memory," *Jurnal SIMETRIS*, vol. 15, no. 1, pp. 121–136, 2024, doi: 10.24176/simet.v15i1.10505.

[16] A. Bahari and K. E. Dewi, "Peringkasan Teks Otomatis Abstraktif Menggunakan Transformer Pada Teks Bahasa Indonesia," *KOMPUTA : Jurnal Ilmiah Komputer dan Informatika*, vol. 13, no. 1, pp. 83–91, 2024, doi: 10.34010/komputa.v13i1.11197.

[17] M. Julkarnain, E. Mardinata, and R. Susilowati, "Penerapan Transformer-Based Neural Machine Translation untuk Bahasa Bima," *Jurnal Algoritma*, vol. 22, no. 2, pp. 793–802, Nov. 2025, doi: 10.33364/algoritma/v.22-2.2949.

[18] B. A. Firmansyah and F. A. Bachtiar, "Automatic Speech Recognition Bahasa Indonesia menggunakan Unidirectional Gated Recurrent Unit," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 5, no. 12, pp. 5180–5187, 2021, doi: 10.21512/jptiik.v5i12.10200.

[19] I. F. Akbar, Mugiarso, K. F. Ramdhania, and Rasim, "Algoritma Levenshtein Distance sebagai Solusi Efisiensi Pencarian pada Training Registration System," *Journal of Students' Research in Computer Science*, vol. 5, no. 2, pp. 147–154, Nov. 2024, doi: 10.31599/mzr20n84.

[20] D. Bernadisman and R. Lesdiansyah, "Aplikasi Text To Speech Berbasis Android Menggunakan Xml Java," *Jurnal SIBERNETIKA*, vol. 8, no. 2, pp. 62–72, 2023, doi: 10.56459/sibernetika.v8i2.68.

[21] A. Tholib, *Implementasi Algoritma Machine Learning Berbasis Web dengan Framework Streamlit*, 1st ed. Pustaka Nurja, 2023. [Online]. Available: https://pustakanurja.unuja.ac.id

[22] N. Trivedi and R. K. Sarvadevabhatla, "PSUMNet: Unified Modality Part Streams are All You Need for Efficient Pose-based Action Recognition," *ArXiv*, Aug. 2022, doi: 10.48550/arXiv.2208.05775.

[23] G. H. Samaan *et al.*, "MediaPipe's Landmarks with RNN for Dynamic Sign Language Recognition," *Electronics (Switzerland)*, vol. 11, no. 19, pp. 1–15, Oct. 2022, doi: 10.3390/electronics11193228.

[24] K. B. Yoga and C. Pramartha, "Pengenalan Tulisan Tangan Aksara Bali Menggunakan Algoritma CRNN," *JNATIA*, vol. 3, no. 4, pp. 845–854, 2025, doi: 10.24843/JNATIA.2025.v03.i04.p14.

[25] E. Ismanto *et al.*, *Revolusi Natural Language Processing: Ketika Manusia Mesin Memahami Bahasa*. PT. Penerbit Qriset Indonesia, 2025.