Jurnal Teknik Informatika (JUTIF)
Vol. 5, No. 4, August 2024, pp. 1051-1061
DOI: https://doi.org/10.52436/1.jutif.2024.5.4.1492
p-ISSN: 2723-3863
e-ISSN: 2723-3871

DETECTION OF ACTIONS BISINDO (INDONESIAN SIGN LANGUAGE) INTO TEXT-TO-SPEECH USING LONG SHORT-TERM MEMORY WITH MEDIAPIPE HOLISTICS

Risda Rosdiana Agustin[*1], Hendra Maulana[2], Eka Prakarsa Mandyartha[3]

[1,2,3]Informatics, Faculty Computer Science, Universitas Pembangunan Nasional "Veteran" Jawa Timur, Indonesia
Email: [1]19081010156@student.upnjatim.ac.id, [2]hendra.maulana.if@upnjatim.ac.id, [3]eka_prakarsa.fik@upnjatim.ac.id

(Article received: October 26, 2023; Revision: November 7, 2023; published: July 29, 2024)

Abstract

Sign language is frequently used by those who have difficulty hearing or speaking to communicate. Because it is a non-verbal language that expresses meaning through hand and body gestures, sign language is an essential form of communication for people who rely on it. The objective of this work is to develop a detection that can understand actions made in Indonesian Sign Language (BISINDO), translate them into text, and use speech recognition (Text-to-Speech) to provide audio responses. In particular at Sekolah Luar Biasa, the main objective is to assist and enhance communication among persons with impairments. Long Short-Term Memory (LSTM) and Mediapipe Holistics are use to achieve its objectives. It is demonstrated how LSTM and Mediapipe Holistics enhance performance and accuracy using two different dataset types. The first dataset landmarks created using the Mediapipe Holistics model, while the second dataset provides original shots devoid of landmarks. Batch size and epoch settings are among the many parameters needed for training and testing processes. Model using the landmark-free dataset only manages to reach an accuracy of approximately 89.33%, the model using the landmark with mediapipe of accuracy of about 96.67%. Furthermore, the landmark-based model exhibits strong F1 scores, recall, and precision. The research successfully recognizes a number of BISINDO acts, such as "saya" (I), "kamu" (you), "ayah" (father), "ibu" (mother), and others present in the dataset. On the basis of the gestures it has identified can also make speech.

Keywords: *BISINDO, LSTM, Mediapipe Holistics, Sign language.*

DETEKSI GERAKAN BISINDO (BAHASA ISYARAT INDONESIA) MENJADI TEXT-TO-SPEECH MENGGUNAKAN LONG SHORT-TERM MEMORY (LSTM) DENGAN MEDIAPIPE HOLISTICS

Abstrak

Dalam kehidupan sehari-hari, penyandang tunarungu dan tunawicara menggunakan bahasa isyarat untuk berkomunikasi [20]. Bahasa isyarat adalah bahasa yang tidak menggunakan bunyi ucapan manusia atau tulisan [11]. Bahasa isyarat menggunakan gerakan tubuh untuk menunjukkan maknanya [3]. Penelitian ini mengembangkan deteksi gerakan Bahasa Isyarat Indonesia (BISINDO) menjadi teks dan menggabungkan suara (Text-to-Speech) sebagai respons suara. Dalam upaya untuk mempermudah media pembelajaran penyandang seperti Sekolah Luar Biasa dalam mempelajari kosa kata jadi kalimat dan berkomunikasi kepada masyarakat [19]. Penelitian ini menggunakan Long Short-Term Memory dan Mediapipe. Untuk membuktikan bahwa Long Short-Term Memory (LSTM) dan Mediapipe Holistics dapat meningkatkan performa dan akurasi, terdapat perbandingan dua jenis dataset yang digunakan. Dataset pertama berisi gambar dengan landmark tubuh yang dibuat oleh model Mediapipe Holistics, dan dataset kedua berisi gambar asli tanpa landmark. Proses pelatihan dan pengujian dilakukan dengan berbagai parameter, seperti batch size dan epoch. Hasilnya menunjukkan bahwa model dengan dataset yang berisi landmark memiliki akurasi tertinggi sekitar 96,67%, sedangkan model tanpa landmark memiliki akurasi sekitar 89,33%. Model dengan dataset yang berisi landmark juga memiliki skor recall, presisi, dan F1 yang baik. Penelitian ini mampu dengan mendeteksi berbagai gerakan BISINDO, seperti "saya", "kamu", "ayah", "ibu", dan gerakan lainnya yang terdapat dalam dataset. Selain itu, program ini dapat menghasilkan suara berdasarkan gerakan yang terdeteksi.

Kata kunci: *Bahasa Isyarat, BISINDO, LSTM, Mediapipe Holistics*

1. PENDAHULUAN

Manusia dapat berkomunikasi sehingga mereka dapat membuat kerangka rujukan untuk menafsirkan keadaan apa pun yang mereka hadapi [21]. Namun, halnya dengan penyandang tunarungu. Dalam kehidupan sehari-hari mereka, penyandang tunarungu dan tunawicara menggunakan bahasa isyarat untuk berinteraksi satu sama lain [7]. Bahasa isyarat, menurut Pratikja (2018) [11], didefinisikan sebagai bahasa yang tidak menggunakan bunyi tulisan atau ucapan manusia saat digunakan [17]. Bahasa isyarat digunakan melalui gerakan badan dan mimik muka [6]. Pada tahun 2021 menurut data penyandang disabilitas tunarungu dan tunawicara dari KEMENSOS (Kementrian Sosial Republik Indonesia), jumlah penyandang tunarungusebanyak 13.800 dan tunawicara sebanyak 5.580 orang. Gerakan Kesejahteraan Tunarungu Indonesia (GERKATIN) mengembangkan bahasa BISINDO, yang dikembangkan oleh masyarakat tunarungu sendiri. Bahasa isyarat sebagian besar digunakan oleh para penyandang disabilitas, dan hanya sedikit hanya sedikit orang lain yang memahaminya, seperti keluarga, aktivis, dan guru di Sekolah Luar Biasa (SLB) [14].

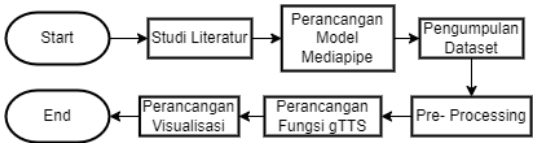
Oleh karena itu dibutuhkan suatu metode yang dapat membaca gerakan isyarat tangan, tubuh, dan wajah untuk diterjemahkan sebagai teks, sehingga bisa berguna sebagai salah satu media yang menunjang pembelajaran anak-anak pada Sekolah Luar Biasa (SLB) untuk media pembelajaran dalam mengolah kosa kata jadi kalimat dan berkomunikasi dengan masyarakat lainnya.

Beberapa penelitian sebelumnya, yang dilakukan oleh Ayu Nijmatul Aliyah (2022), model Long Short-Term Memory (LSTM) [1] dan MediaPipe digunakan untuk mendeteksi gerakan bahasa isyarat. Mereka mengumpulkan data dari sembilan puluh gerakan bahasa isyarat, yang dibagi menjadi kata-kata seperti "tolong", "maaf", dan "terima kasih". Mengingat metrik, ketepatan, dan skor F1 masing- masing mencapai 0,9, hasil akurasi mencapai 0,77778. Penelitian lain yaitu Husna Moetia Putri (2022) [4] juga menggunakan LSTM dan MediaPipe untuk mengidentifikasi bahasa isyarat. Data mereka diperoleh dari tiga puluh kelas gerakan bahasa isyarat. Dengan 10 model kelas dan batch size 64, akurasi mencapai 92%, tetapi dengan 30 model kelas, akurasi turun menjadi 65% pada epoch 500. Klasifikasi menghadapi masalah pencahayaan yang tidak merata dan kesulitan menemukan gerakan serupa. Kesimpulannya, penggunaan LSTM dan MediaPipe mampu mengenali gerakan bahasa isyarat dengan baik, tergantung pada jumlah kelas dan pengaturan parameter seperti batch size dan epoch.

Setelah mengetahui permasalahan yang ada dan sebagai upaya untuk menunjang pembelajaran anak-anak pada SLB serta meningkatkan akurasi dari penelitian sebelumnya. Oleh karena itu, penelitian ini mengeksplorasi algoritma lain, yaitu LSTM (Long Short-Term Memory), yang berbasis RNN dan memiliki kemampuan mendeteksi real-time [2]. LSTM network adalah pilihan yang ideal karena kemampuannya untuk menyimpan ingatan dalam jumlah besar pada saat yang sama dan korelasi yang kompleks antara data yang memberikan informasi yang sangat bermanfaat [8]. Selain itu, LSTM memiliki blok memori yang memungkinkan penentuan nilai yang tepat. Selain itu, LSTM dianggap memiliki kualitas yang lebih tinggi daripada metode regresi Naive Bayesian, serta metode lain (Dong et.al 2017) [10].

2. METODE PENELITIAN

Dalam penelitian ini diperlukan langkah – langkah untuk mengerjakan proses pengerjaan sebuah penelitian dengan baik, sehingga penulis dapat menemukan, menjawab, serta menemukan tujuan dari rumusan masalah yang dijelaskan. Berikut pada gambar 2.1.



Gambar 2.1 Tahapan Penelitian

Gambar 2.1 menjelaskan bahwa langkah – langkah untuk menyelesaikan penelitian. Tahap awal merupakan studi literatur dengan mencari dan membaca referensi jurnal dari peneliti terdahulu, Tahap selanjutnya melakukan perancangan model mediapipe dan pengumpulan data seperti pembuatan dataset manual nya. Setelah itu melakukan pe processing dengan membuat label untuk setiap gerakan, pembagian data, kemudian membuat perancangan model layer LSTM dan layer Dense, Setelah itu memilih optimizer yang akan digunakan. kemudian melakukan pelatihan model dan evaluasi model seperti akurasi, presisi, recall, dan F1-Score. Setelah itu merancang fungi gTTS untuk fitur text-to-speech dan Visualisasi probabilitas.

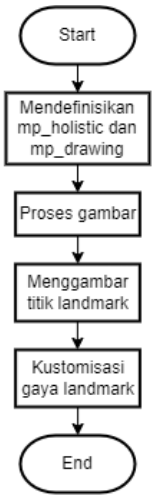
2.1. Studi Literatur

Studi pustaka bertujuan untuk mendapatkan dasar teori pendukung dengan melakukan pengumpulan data, membaca, mencatat, diskusi, serta mengolah data yang akan diteliti. Pada tahap ini, penulis mencari dan mengumpulkan data terkait metode LSTM, Gerakan Bahasa Isyarat Indonesia (BISINDO), Mediapipe Holistics, Text-to-Speech serta mengumpulakan penelitian yang terdahulu. Untuk literatur pada penelitian ini, penulis melakukan dengan cara membaca artikel, jurnal, diskusi serta

web untuk mencatat terkait hal – hal yang berkaitan pada penelitian tersebut. Untuk literatur yang digunakan pada penulis pada penelitian ini dapat ditemukan pada daftar pustaka yang tersedia di bagian akhir.

2.2. Perancangan Model Mediapipe

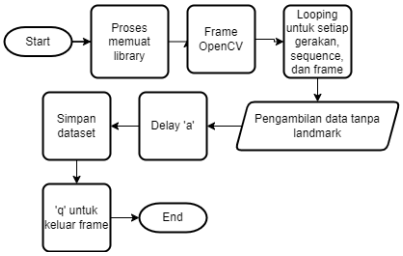
Tahapan selanjutnya dataset akan berisi informasi keypoints dari berbagai bagian tubuh yang dideteksi oleh model Mediapipe. Dengan mendefinisikan pendeteksian danpenggambaran landmark, memproses gambar RGB ke BGR, menggambar titik-titik landmark, dan kustomisasi gaya landmark yang digunakan. Berikut tahapan perancangan model Mediapipe Holistics pada gambar 2.2.



Gambar 2.2 Perancangan model mediapipe

2.3. Pengumpulan Data

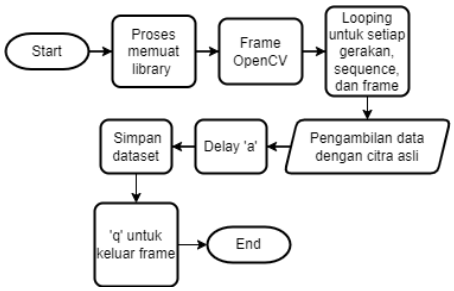
Pada tahap pengumpulan dataset program ini dilakukan dengan menggunakan kamera webcam untuk merekam video dari berbagai aksi atau gerakan yang ingin dikenali. Terdapat 2 jenis tahapan pengumpulan data yaitu pengumpulan dataset landmark dan pengumpulan datasettanpa landmark. Berikut tahapan pengumpulan dataset landmark pada gambar 2.3.



Gambar 2.23 Pengumpulan dataset landmark

Pada gambar 2.2 menjelaskan Setiap frame video akan diproses menggunakan model Mediapipe. Model ini akan mendeteksi dan mengekstraksi keypoints dari setiap bagian tubuh, wajah, dan tangan dalam frame tersebut. Model ini akan mendeteksi dan mengekstraksi keypoints dari setiap bagian tubuh, wajah, dan tangan dalam frame tersebut.

Pengumpulan dataset yang tidak berlandmark dilakukan dengan mengambil citra asli tanpa landmark mediapipe dengan penyimpanan formatnya .JPG. Sehingga dataset tersebut akan memuat dataset berupa citra asli dan akan menjadi perbandingan performa antara dataset berlandmark mediapipe dan dataset tidak berlandmark mediapipe, berikut tahapan pengumpulan dataset tanpa landmark pada gambar 2.3.



Gambar 2.3 Pengumpulan dataset tanpa landmark

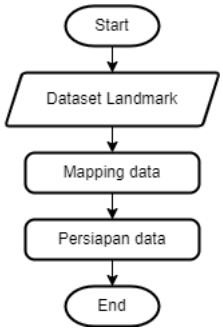
2.4. Pre Processing

Pada tahapan pre processing memiliki beberapa tahapan yaitu pra proses, pembagian data, perancangan model LSTM untuk pelatihan model dan evaluasi model.

2.4.1. Pra Proses

Pra proses merupakan tahapan yang dilakukan sebelum melakukan proses klasifikasi. Pada tahap ini dilakukan pelabelan data atau mapping, dan mempersiapkan data pada masing- masing dataset dan proses resize. Sebelum pembagian data, akan dilakukan proses mapping atau labelling sehingga kelas akan memiliki urutan dan bisa digunakan pada *preprocessing*. Setelah mapping akan dilakukan persiapan untuk masing-masing dataset.

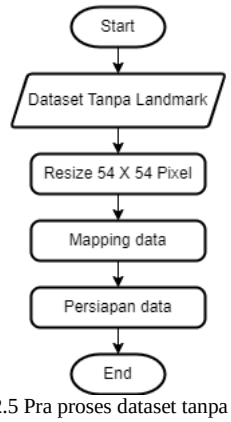
Pada dataset tanpa landmark memiliki tahap khusus yaitu proses resize agar bisa digunakan pada saat *preprocessing* pada model LSTM. Berikut tahapan mapping label pada gambar 2.4.



Gambar 2.4 Pra proses dataset landmark

Pada dataset tidak berlandmark ada langkah khusus yang dilakukan dengan mengubah ukuran citra asli yang berformat .jpg karena model LSTM menerima input dalam bentuk urutan (sequence) yang

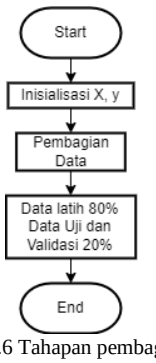
memiliki panjang dan lebar yang tetap dan dimensi yang sama, sehingga proses penyesuaian ukuran gambar diperlukan agar model LSTM dapat memprosesnya dengan benar. Apabila semakin tinggi ukuran citra, maka semakin berat dalam melakukan proses klasifikasi. Untuk penelitian ini menggunakan ukuran gambar kecil yang berukuran 54 x 54 piksel agar bisa diproses untuk proses training. Berikut tahapan pra proses dataset tanpa landmark pada gambar 2.5.



Gambar 2.5 Pra proses dataset tanpa landmark

2.4.2. Pembagian Data

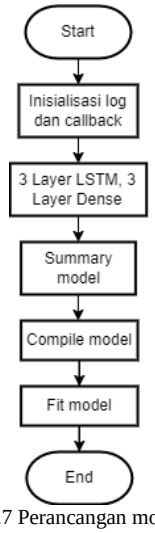
Pada tahap pembagian data menggunakan referensi dari peneliti sebelumnya. Berdasarkan Riberu (2023) [5] menyatakan tidak ada ketentuan yang baku dalam pembagian dataset sehingga beberapa rasio umum yang digunakan yaitu 80:20, 90:10, dan 70:10. Selain itu peneliti sebelumnya juga menyatakan pemilihan rasio 80:20 juga dipilih karena merujuk dari hukum pinsip patero yang menetapkan bahwa 80% konsekuensi dan 20% penyebab. Sehingga penulis membagi data dengan 80:20. Berikut tahapan pembagian data pada gambar 2.6.



Gambar 2.6 Tahapan pembagian data

2.4.3. Perancangan Model LSTM

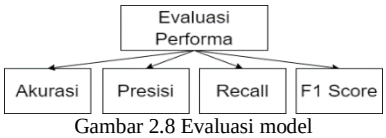
Tahap pertama yaitu dimulai dari dataset yang tidak berlandmark dulu, kemudian dataset yang berlandmark. Setelah itu memuat library yang akan digunakan, kemuadian memuat dataset yang digunakan lalu berlanjut pada pra proses. Model yang digunakan mempunyai 3 Layer LSTM, dan 2 Layer Dense. Pada penelitian ini untuk parameter yang akan dikustomisasi adalah epoch dan batch size, yang dimana kedua parameter tersebut yang akan menentukan persentase akurasi yang dihasilkan pada penelitian ini untuk perbandingan performa dataset berlandamrk dan dataset tanpa landmark mediapipe. Berikut tahapan perancangan model LSTM pada gambr 2.7.



Gambar 2.7 Perancangan model LSTM

2.4.4. Evaluasi Model

Penulis melakukan evaluasi performa atas dua skenario pengujian yang dilakukan menggunakan confusion matrix. Evaluasi ini bertujuan untuk mengetahui performa skenario pengujian mana yang memiliki nilai accuracy, precision, recall, dan f1 score yang lebih baik pada gambar 2.8.



Gambar 2.8 Evaluasi model

2.5. Perancangan model *gTTS*

Perancangan fungsi gTTS adalah tahap untuk menghasilkan output suara yang didasarkan pada tindakan yang terdeteksi dalam teks. Ini dimulai dengan menginstal library gTTS dan mengimport modul gTTS, membuat fungsi *"text_to_speech"* untuk menerima teks sebagai argumen, mengubahnya menjadi suara, dan menyimpan file suara tersebut dengan nama *"output.mp3."* Fungsi ini mengambil teks dari *"sentence"* sebagai input, dan kustomisasi *"lang='id"* menunjukkan bahasa yang digunakan, yaitu bahasa Indonesia, dan *"slow=False"* menunjukkan bahwa kecepatan output suaranya tidak akan lambat. Setelah suara yang dihasilkan disimpan dalam file *"output.m3"*, modul *"subprocess"* digunakan untuk memulai pemutaran file "output.mp3" menggunakan aplikasi bawaan perangkat, seperti Media Player, dengan *"start output.mp3."* Berikut tahapan perancangan model gTTS pada gambar 2.8.

Gambar 2.9 Perancangan gTTS

2.6. Perancangan Visualisasi

Perancangan ini membantu memvisualisasikan hasil tindakan dalam bingkai video pada saat real-time dengan menampilkan nama-nama tindakan yang digunakan pada dataset di sebelah kiri frame. Nama-nama tindakan yang mungkin diprediksi oleh model atau sistem dapat dilihat dengan jelas melalui desain visualisasi ini. Nama-nama tindakan membantu memahami hasil prediksi.

Visualisasi pada setiap tindakan nanti akan ada persegi panjang yang menunjukkan kemungkinan dari masing-masing tindakan yang diprediksi. Lebar dari setiap persegi panjang dihitung berdasarkan probabilitasnya, sehingga semakin tinggi probabilitas tindakan, semakin lebar persegi panjang nya. Tahap ini memberikan gambaran yang jelas tentang seberapa yakin sistem dalam memprediksi tindakan tertentu. Berikut tahapan visualisasi pada gambar 2.10.

Gambar 2.10 Tahapan visualisasi

2.7. Skenario Uji Coba

Pada uji coba model menggunakan dua dataset berbeda yaitu dataset tanpa landmark (citra asli) dan dataset dengan landmark yang dihasilkan oleh Mediapipe Holistics. Percobaan bertujuan untuk mengevaluasi sejauh mana penggunaan landmark dari Mediapipe dapat meningkatkan akurasi model. Proses pelatihan dilakukan dengan menggunakan optimizer Adam dan variasi batch size 64 serta jumlah variasi epoch 20, 50, 80, 100, dan 120 untuk melihat perkembangan performa model. Data dibagi menjadi uji dan validasi 20 % dengan data latih 80% atau rasio 80:20.

3. HASIL DAN PEMBAHASAN

Pada bagian ini akan membahas terkait hasil implementasi program dari Deteksi gerakan BISINDO (Bahasa Isyarat Indonesia) menjadi Text-to-Sppech menggunakan Mediapipe Holistics dan Long Short-Term Memory (LSTM).

3.1. Persiapan Dataset

Setiap dataset mempunyai 750 file dalam direktori. Setiap dataset memiliki panjang sequence yang sama, yaitu 30. Dataset landmark dengan gerakan memuat 258 ekstrak keypoints dari Mediapipe Holistics sedangkan Dataset tanpa landmark memuat 8748 fitur dari ukuran dimensi. Dataset tanpa landmark berisi citra asli yang digunakan sebagai pembanding, sedangkan dataset dengan landmark berisi citra-citra yang telah diberi landmark menggunakan Mediapipe Holistics.. Konsistensi ini memungkinkan untuk membandingkan hasil antara dua pendekatan berbeda.

3.2. Implementasi Model Mediapipe

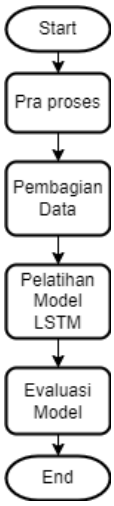
Implementasi Mediapipe Holistics pada penelitian ini menggabungkan deteksi wajah, pose, tangan kanan dan tangan kiri dalam satu model. Terdapat fungsi untuk mengubah warna dan gaya landmark yang digambar pada saat pengambilan dataset maupun pada saat pengujian real-time. Penggunaan warna pada landmark pose wajah dan tubuh berwarna merah (RGB : 255, 0, 0)]. Selain itu untuk landmark tangan kanan adalah berwarna biru (RGB : 0, 0, 255) . Kemudian landmark tangan kiri adalah berwarna hijau (0, 255, 0). Masing - masing memiliki ketebalan garis yang menghubungkan landmark sebesar 2 piksel dan memiliki ukuran lingkaran yang menggambarkan landmark sebesar 4 piksel. Dengan pengaturan warna yang berbeda-beda untuk setiap bagian (pose, tangan kiri, dan tangan kanan),sehingga dapat dengan mudah membedakan landmark yang mewakili berbagai bagian tubuh dalam tampilan video atau gambar yang dihasilkan oleh model MediaPipe Holistic.

Setelah itu, mengambil dan merepresentasikan pose tubuh, tangan kanan dan tangan kiri dalam bentuk array numerik. Landmark pose tubuh terdiri dari serangkaian titik-titik kunci yang menggambarkan posisi berbagai bagian tubuh seperti mata, hidung, mulut, bahu, siku, pergelangan tangan, pinggul, lutut, dan pergelangan kaki. Landmark tangan kiri dan tangan kanan juga terdiri dari titik-titik kunci yang mewakili posisi berbagai bagian tangan seperti jari-jari, telapak tangan, dan

pergelangan tangan Masing-masing landmark pose, tangan kanan dan kiri memiliki koordinat x, y, z. Terdapat 33 landmark untuk pose tubuh dan masing-masing dengan 4 koordinat, sehingga menjadi 33 x 4 = 132. Pada 21 landmark pada tangan kanan dan kiri dengan masing-masing 3 koordinat, sehingga menjadi 21 x 3 = 63 untuk tangan kanan dan 21 x 3 = 63 untuk tangan kiri. Kemudian jumlah total semua menjadi 132 + 63 + 63 = 258.

3.3. *Pre Processing*

Pada tahap ini menjelaskan secara detail dari perancangan dari tahapan penelitian penulis. Berikut tahapan *preprocessing* pada penelitian ini pada gambar 3.1.



Gambar 3.1 Tahapan pre processing

3.3.1. Pra Proses

Pada tahap ini diperlukan memetakan nama tindakan ke angka 0 hingga 24. Tahap ini dimulai dengan memproses dataset sebelumnya yang mengandung daftar nama gerakan yang disimpan dalam variabel "actions" pada data path. Proses ini membuat label_map dengan mengiterasi actions pada data path dan mengambil setiap nama gerakan dan memberikan label nomor urut 0 hingga 24. Berikut mapping data pada masing-masing dataset pada gambar 3.2.

```
In [11]: label_map = {label:num for num, label in enumerate(actions)}

In [6]: label_map
Out[6]: {'saya': 0,
         'kamu': 1,
         'ayah': 2,
         'ibu': 3,
         'kakak': 4,
         'adik': 5,
         'teman': 6,
         'senang': 7,
         'sedih': 8,
         'marah': 9,
         'takut': 10,
         'membantu': 11,
         'cinta': 12,
         'minta-maaf': 13,
         'terimakasih': 14,
         'bermain': 15,
         'belajar': 16,
         'tolong': 17,
         'rumah': 18,
         'sekolah': 19,
         'taman': 20,
         'kelas': 21,
         'dan': 22,
         'dengan': 23,
         'dari': 24}
```

Gambar 3.2 Mapping label

Setelah mapping, proses ini melibatkan pengisian dua list kosong: 'sequence' dan 'labels' yang digunakan untuk menyimpan data latih. Langkah pertama adalah mengambil setiap gerakan dari daftar 'actions'. Kemudian, setiap 'sequence' diambil sebanyak yang ditentukan oleh 'no_sequences', yaitu 30. Sedangkan 'Sequence_length' digunakan untuk menentukan jumlah frame yang akan diambil dari file citra .npy pada dataset berlandmark dan .jpg pada dataset tanpa landmark yang tersimpan di direktori data path. Berikut tahapan pra proses persiapan dataset yang berlandmark pada gambar 3.3:

```
sequences, labels = [], []
for action in actions:
    for sequence in range(no_sequences):
        window = []
        for frame_num in range(sequence_length):
            res = np.load(os.path.join(DATA_PATH, action, str(sequence), "{}.npy".format(frame_num)))
            window.append(res)
        sequences.append(window)
        labels.append(label_map[action])
```

Gambar 3.3 Persiapan dataset landmark

Berikut persiapan dataset tanpa landmark yang mempunyai format .jpg sebagai pembeda pada gambar 3.4.

```
sequences, labels = [], []
for action in actions:
    for sequence in range(no_sequences):
        window = []
        for frame_num in range(sequence_length):
            img_path = os.path.join(DATA_PATH, action, str(sequence), f"{frame_num}.jpg")
            img = cv2.imread(img_path)
            window.append(img)
        sequences.append(window)
        labels.append(label_map[action])
```

Gambar 3.4 Persiapan dataset tanpa landmark

Kemudian terdapat proses resize hanya dilakukan untuk dataset tanpa landmark. Pada dataset tanpa landmark ini menggunakan ukuran 54x54 piksel. Memori GPU yang digunakan pada penelitian ini memiliki keterbatasan sehingga diperlukan tahap resize untuk bisa melakukan pelatihan dengan memori yang cukup. Selain itu, model LSTM memiliki jumlah neuron pada lapisan input yang tetap, sehingga dimensi input harus tetap. Oleh karena itu, dengan meresize gambar, dimensi setiap gambar memiliki dimensi yang konsisten. Resize juga dapat mempercepat waktu pelatihan. Berikut proses rezie untuk dataset tanpa landmark pada gambar 3.4.

```
width, height = 54, 54  # Ukuran yang diinginkan

for i in range(len(sequences)):
    for j in range(len(sequences[i])):
        sequences[i][j] = cv2.resize(sequences[i][j], (width, height))
```

```
for i in range(len(sequences)):
    for j in range(len(sequences[i])):
        sequences[i][j] = sequences[i][j] / 255.0
```

```
sequences = np.array(sequences)

sequences = sequences.reshape((len(sequences), 30, 54*54*3))
```

Gambar 3.5 Proses resize

Dari proses pada gambar 3.4 ada dua perulangan. Perulangan pertama '*for i in range (len(sequences))'* memberikan akses ke setiap elemen dalam list sequences, dan perulangan kedua *'for j in range(len(sequences[i]))'* memberikan akses ke setiap gambar dalam elemen *sequences[i]*. Fungsi

cv2.resize() dalam *OpenCV* digunakan untuk melakukan resize gambar. Gambar yang awalnya terdapat pada *sequences[i][j]* akan diubah menjadi nilai width dan height, yaitu 54x54 piksel. Setelah resize selesai, gambar-gambar tersebut akan diubah dalam array numpy. Kemudian mengubah bentuk array NumPy menjadi tiga dimensi. Dimulai dengan dimensi '*len (sequences)'*, yang menunjukkan jumlah gambar yang ada dalam *sequences*, dimensi kedua adalah '30' yang digunakan untuk mengatur gambar dalam urutan sebanyak '30', dan dimensi ketiga adalah 54 x 54 x 3, yang menunjukkan ukuran gambar yang diresize menjadi 54 x 54 piksel dan memiliki tiga channel warna (RGB).

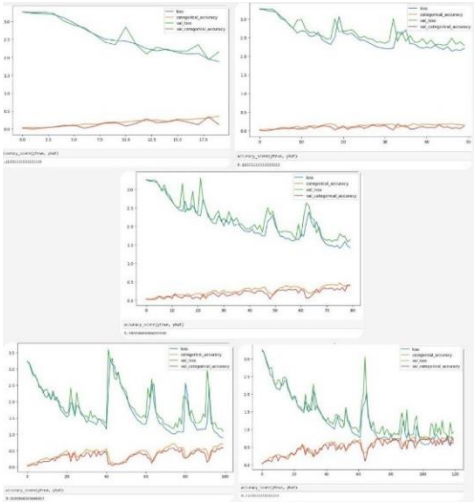
3.3.2. Implementasi Model LSTM

Sebelum layer lstm, menambahkan Tensorboard dengan mendefinisikan dulu log_dir dan tb_callbacks. log akan menyimpan log pada saat training, sedangkan callbacks untuk memantau performa model. Berikut model LSTM pada penelitian ini pada gambar 3.6.

```
log_dir = os.path.join('Logs')
tb_callback = TensorBoard(log_dir = log_dir)

model = Sequential()
model.add(LSTM(64, return_sequences=True, activation='tanh', input_shape=(30, 258)))
model.add(LSTM(128, return_sequences=True, activation='tanh'))
model.add(LSTM(64, return_sequences=False, activation='tanh'))
model.add(Dense(64, activation='tanh'))
model.add(Dense(32, activation='tanh'))
model.add(Dense(25, activation='softmax'))

model.summary()

model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['categorical_accuracy'])

history=model.fit(X_train, y_train, validation_data=(x_val, y_val), batch_size=128, epochs=100, callbacks=[tb_callback])
```

Gambar 3.6 Model pelatihan LSTM

Model yang digunakan mempunyai 3 Layer LSTM, dan 3 Layer Dense. Layer pertama memiliki 64 unit dengan memiliki input shape dengan panjang urutan 30 dan setiap urutan untuk dataset berlandmark memiliki 258 fitur sedangkan dataset tidak berlandmark memiliki 8748 fitur., layer kedua memiliki 128 unit, layer ketiga memiliki 64 unit sedangkan layer dense memiliki 64, 32 dan layer terakhir memiliki 25 yang sesuai dengan jumlah kelas yang akan dilatih dengan masing-masing menggunakan aktivasi tanh. Tahap selanjutnya memuat optimizer yang digunakan yaitu Adam beserta parameter batch size, loss dan metrik (categorical_crossentropy dan categorical_accuracy) dan dilatih dengan epoch 20, 50, 80, 100, dan 150. Penggunaan tensorboard juga untuk melacal loss dan memeriksa metrik selama pelatihan. Tahap terakhir mengevaluasi model seperti confussion matrix, akurasi, presisi, recall, dan f1-score lalu menyimpan modelnya dalam bentuk .h5py. Pada pelatihan menggunakan dataset tanpa landmark atau citra asli berupa format .jpg. Penggunaan landmark ini sebagai tolak ukur dengan model dataset dengan landmark. Pelatihan model dijalankan dengan batch size 64 serta variasi jumlah epoch yaitu 20, 50, 80, 100, dan 120. Hasil dari pelatihan ini pada gambar 3.7 menunjukkan :



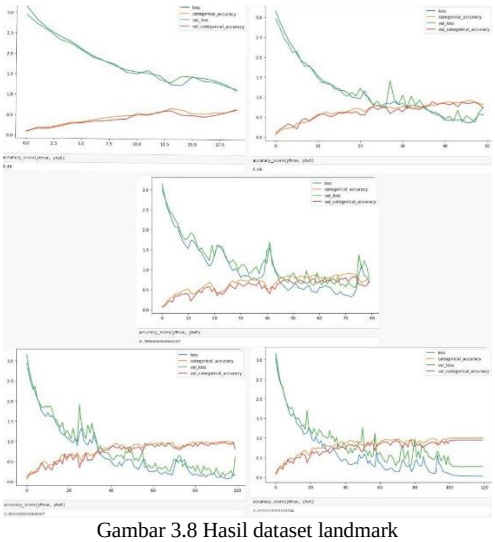
Gambar 3.7 Hasil pelatihan dataset tanpa landmark

Pada gambar 3.7 menunjukkan hasil analisis untuk dataset tanpa landmark hasil tertinggi pada pelatihan menggunakan batch size 64 dan epoch 120 dengan memperoleh nilai akurasi 0,7333333. Berikut hasil keseluruhan epoch pada tabel 3.1.

Tabel 3.1 Hasil dataset tanpa landmark

Epoch	Batch Size	Akurasi
20	64	0,2133
50	64	0,1933
80	64	0,3866
100	64	0,5866
120	64	0,7333

Sedangkan pada pelatihan menggunakan dataset landmark yang menggunakan model mediapipe dan format .npy. Penggunaan landmark ini sebagai tolak ukur dengan model dataset dengan landmark. Pelatihan model dijalankan dengan batch size 64 serta variasi jumlah epoch yaitu 20, 50, 80, 100, dan 120. Hasil dari pelatihan ini pada gambar 3.8 menunjukkan:



Gambar 3.8 Hasil dataset landmark

Pada gambar 3.8 menunjukkan hasil analisis untuk dataset landmark hasil tertinggi pada pelatihan menggunakan batch size 64 dan epoch 120 dengan

memperoleh nilai akurasi 0,9733. Berikut hasil keseluruhan epoch pada tabel 3.2.

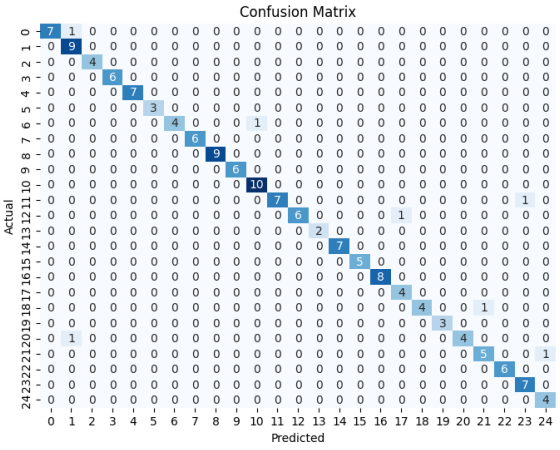
Tabel 3.2 Hasil dataset landmark

Epoch	Batch Size	Akurasi
20	64	0,68
50	64	0,7133
80	64	0,8933
100	64	0,9006
120	64	0,9533

Sehingga diperoleh hasil perbandingan antara dataset tanpa landmark dan dataset dengan landmark membuktikan bahwa penggunaan landmark dari Mediapipe Holistics mampu membantu memperbaiki performa model. Model yang digunakan untuk pengujian real-time dan perhitungan recall, presisi dan F1-score adalah model dengan akurasi tertinggi yaitu Dataset Landmark dengan akurasi 0.9533.

3.3.3. Evalulasi Model

Setelah proses training model, dan mendapatkan akurasi model tertinggi akan dihitung evaluasi model nya meliputi confussion matrix, recall, presisi dan f1 score. Berikut hasil confusion matrix nya pada gambar 3.9:



Gambar 3.9 Confusion matrix

Dari gambar 3.9 akan diperoleh hasil recall, presisi dan f1 score sebagai berikut:

Tabel 3.3 Evaluasi model

Evaluasi	Hasil
Recall	0,954
Presisi	0,961
F1 Score	0,955

3.4. Implementasi *gTTS*

Pada tahap menggunakan if USE_VOICE, pengecekan kondisi akan dilakukan untuk setiap kata tindakan yang dikenali oleh model melalui loop. Jika opsi USE_VOICE diatur sebagai True yang menunjukkan bahwa pengguna ingin menggunakan suara, maka setiap kata tindakan akan diubah menjadi suara melalui fungsi text_to_speech(word). Fungsi ini menggunakan modul gTTS untuk mengonversi teks menjadi suara pada gambar pada gambar 3.10 berikut.

```
            if USE_VOICE:
                text_to_speech(word)
        sentence.extend(action_words)
else:
    sentence.append(actions[np.argmax(res)])
    if USE_VOICE:
        text_to_speech(' '.join(sentence))
```

Gambar 3.10 Implementasi gTTS

Kata tindakan dimasukkan ke dalam daftar sentence setelah diubah menjadi suara. Daftar sentence ini berfungsi sebagai catatan kata-kata yang telah diucapkan. Jika USE_VOICE tidak diaktifkan menjadi "False", yang berarti pengguna tidak ingin menerima respons suara, maka tidak akan ada pemanggilan text_to_speech atau suara yang dibuat seperti pada gambar 3.11.

```
USE_VOICE = False
```

Gambar 3.11 Nonaktif gTTS

3.5. Implementasi Visualisasi

Langkah selanjutnya adalah implementasi fungsi untuk visualisasi. Visualisasi bisa mempermudah memberikan informasi atau tampilan yang lebih jelas dan mudah dimengerti. Dengan menggunakan visualisasi ini, dapat membandingkan tindakan mana yang memiliki probabilitas lebih tinggi dan juga membantu dalam menentukan tindakan atau objek mana yang paling mungkin untuk berhasil dalam pengambilan keputusan. Berikut penjelasan dan gambar 3.12.

```
def prob_viz(predictions, actions, input_frame):
    output_frame = input_frame.copy()
    frame_height = 480
    text_height = 8
    rect_height = 8

    for num, prediction in enumerate(predictions):
        prob = prediction if num == predictions[-1] else 0
        cv2.rectangle(output_frame, (0, text_height + num * (text_height + rect_height)),
                      (rect_width, text_height + 5 + num * (text_height + rect_height)), (0, 0, 0), 1)
        cv2.putText(output_frame, actions[num], (10, text_height + 5 + num * (text_height + rect_height)),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)

    return output_frame
```

Gambar 3.12 Implementasi Visualisasi

Dalam fungsi *prob_viz*, ada tiga argumen: prediksi, tindakan, dan input_frame. Terdapat *'predictions'* adalah array yang berisi hasil prediksi atau probabilitas untuk setiap tindakan atau objek, *'actions'* adalah array yang berisi label atau nama tindakan atau objek, dan input_frame adalah bingkai gambar atau video di mana pengenalan atau klasifikasi telah dilakukan. Selain itu, output_frame diinisialisasi sebagai replika dari bingkai masukan (*input_frame*) untuk memulai proses visual sedangkan untuk mengubah tinggi bingkai, tinggi teks, dan tinggi persegi panjang terdapat variabel *frame_height*, *text_height*, dan *rect_heigh*t yang bisa dikustomisasi.

Dalam perulangan, probabilitas diatur sesuai dengan kondisi: jika indeks num adalah indeks terakhir dalam array prediksi (indeks -1), maka probabilitas diatur sesuai dengan prediksi; jika tidak, probabilitas diatur menjadi 0. Ini berarti bahwa hanya

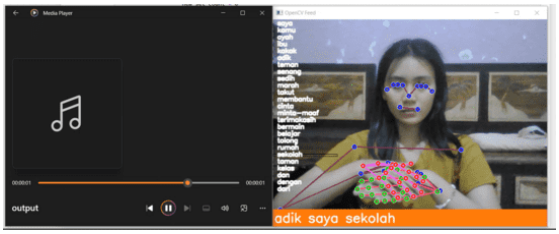
tindakan yang diprediksi sebagai yang paling mungkin akan memiliki probabilitas yang ditampilkan dalam visualisasi. Rect_width, lebar persegi panjang yang akan digambar dalam visualisasi, dihitung dengan probabilitas yang telah diatur (prob), dan dalam kode ini, lebar dihitung dengan mengalikan probabilitas dengan 150 (skala tertentu). Setelah itu, fungsi cv2.rectangle digunakan untuk menggambar persegi panjang dengan warna hitam dan ketebalan garis 1 pada bingkai keluaran (output_frame). Indeks elemen dalam array prediksi digunakan untuk mengatur lokasi dan ukuran persegi panjang.

3.6. Uji Coba Deteksi

Proses testing atau uji coba menggunakan dataset dengan landmark yang memperoleh hasil pelatihan yang cukup bagus. Proses uji coba dilakukan oleh 2 pengguna, dan berbagai background. Proses uji coba juga menggunakan setengah badan seperti pada proses pengambilan dataset. Hasil uji coba dengan pengguna dengan model yang sudah dilatih. Uji coba petama yang ditesting oleh user yaitu deteksi teks per kata, untuk memastikan bahwa masing-masing kata gerakan terdeteksi. Berikut pada gambar 3.13

Gambar 3.13 Deteksi 4 Kata

Gambar 3.13 menunjukkan deteksi untuk kata ayah, saya, membantu dan belajar yang terdeteksi menjadi satu sehingga terlihat menjadi sebuah kalimat didampingi dengan speech setiap kata nya dan akan menjadi perulangan speech dari 4 kata tersebut sehingga terdengar seperti pengucapan kalimat.



Gambar 3.14 Deteksi 3 Kata

Gambar 3.14 menunjukkan deteksi untuk kata adik, saya, sekolah yang terdeteksi menjadi satu sehingga terlihat menjadi sebuah kalimat didampingi dengan speech setiap kata nya dan akan menjadi perulangan speech dari 3 kata tersebut sehingga terdengar seperti pengucapan kalimat.

Gambar 3.15 Deteksi Masing- Masing Kata

Gambar 3.15 menunjukkan deteksi untuk masing- masing kata dari saya, kamu, belajar, tolong sehingga membuktikan bahwa deteksi ini mampu mendeteksi contoh-contoh dari gerakan tersebut.

4. DISKUSI

Penelitian ini mencapai hasil yang signifikan dalam perbandingan dengan peneliti sebelumnya, tingkat akurasinya mencapai 0,953 yang menunjukkan peningkatan yang signifikan dibandingkan dengan studi Ayu Nijmatul Aliyah (2022) [1] dan Husna Moetia Putri (2022) [4] yang mencapai tingkat akurasi sekitar 0,77778 hingga 0,92%. Penggunaan MediaPipe Holistics, yang telah terbukti menunjukkan peningkatan signifikan dalam kualitas deteksi gerakan, merupakan salah satu kontribusi utama penelitian ini. Ini menunjukkan bahwa landmark lebih membantu model memahami gerakan bahasa isyarat daripada gambar asli.

Penelitian ini menunjukkan tidak hanya akurasi yang tinggi, tetapi juga keberhasilan dalam menerapkan teks menjadi suara pada gerakan yang diidentifikasi. Dengan kata lain, model ini tidak hanya mendeteksi gerakan BISINDO dalam bentuk teks, tetapi juga dapat mengubahnya menjadi suara atau audio. Hal ini sangat bermanfaat untuk mendukung pendidikan anak-anak di sekolah luar biasa (SLB), di mana teknologi ini dapat membantu mereka belajar lebih efektif.

5. KESIMPULAN

Pada penelitian ini, penulis berhasil membangun proses deteksi gerakan bahasa isyarat indonesia (BISINDO) menjadi Text-to-Speech menggunakan Long Short-Term Memory (LSTM) dan Mediapipe Holistics. Penelitian ini menggunakan 2 model dataset yang berbeda sebagai tolak ukur perbandingan persentase akurasi tertinggi untuk masing – masing model. Berdasarkan berbagai percobaan dalam tahap pelatihan dan pengujian pada penelitian ini dapat disimpulkan bahwa:

1. *Dataset* yang dilatih dan diuji dengan menggunakan model dataset tanpa landmark dengan batch size sebanyak 64 dan epoch 120 memiliki akurasi tertinggi sebesar 0.89333333333. *Dataset* yang dilatih dan diuji dengan menggunakan model dataset landmark dengan batch size sebanyak 64 dan epoch 120 memiliki akurasi tertinggi sebesar 0.9666666666. Dari kedua model dataset dan

parameter nilai batch sizze dan epoch tersebut dijelaskan bahwa akurasi tertinggi pada penelitian ini dimiliki oleh model dataset dengan landmark dengan nilai akurasi sebesar 0.9666, recall sebesar 0.9652, presisi sebesar 0,975, dan F1-score sebesar 0.9686.

2. Program mampu mendeteksi 25 gerakan BISINDO dengan tepat pada kata saya, kamu, ayah, ibu, kakak, adik, teman, senang, sedih, marah, takut, membantu, cinta, minta maaf, terimakasih, bermain, belajar, tolong, rumah, sekolah, taman, kelas, dan, dengan, dari.
3. Program mampu mengimplementasikan speech pada setiap gerakan yang terdeteksi.

DAFTAR PUSTAKA

[1] A.N. Aliyah, "Implementasi Metode Human Activity Recognition (HAR) Menggunakan Mediapipe Holistics Dan Algoritma Long Short Term Memory (LSTM)," 2022.

[2] W. Amei, D. Huailin, W. Qingfeng, and L. Ling, "A survey of application-level protocol identification based on machine learning," in International Conference on Information Management, Innovation Management and Industrial Engineering, 2011, pp. 201-204.

[3] M.R. Amiarrahman and Tri Handika, "Analisis dan implementasi algoritma klasifikasi Random Forest dalam pengenalan Bahasa Isyarat Indonesia (BISINDO)," in Prosiding Seminar Nasional Inovasi Teknologi, 2017, pp. 83-88.

[4] H. Moetia Putri, "Pendeteksian Bahasa Isyarat Indonesia Secara Real-Time Menggunakan Long Short Term Memory (LSTM)," 2022

[5] F.X. Loren Ruberu, "Sistem Deteksi Simbol pada SIBI (Sistem Isyarat Bahasa Indonesia) Secara Real Time Menggunakan Mediapipe dan LSTM," 2020

[6] A. Pathak, A. Kumar, P. Priyam, P. Gupta, and G. Chugh, "Real Time Sign Language Detection," International Journal for Modern Trends in Science and Technology, 2022, vol. 8, pp. 32-37

[7] Nofal Anam, "Sistem Deteksi Simbol pada SIBI (Sistem Isyarat Bahasa Indonesia) Menggunakan Mediapipe dan ResNet-50". 2020

[8] M. Khaliluzzaman, M.A.B.S. Sayem, and L.K. Misbah, "HActivityNet: A Deep Convolutional Neural Network for Human Activity Recognition," EMITTER International Journal of Engineering Technology, 2021, vol. 9, no. 2, pp. 357-376

[9] J.B. Bullock, "Artificial Intelligence, Discretion, and Bureaucracy," American Review of Public Administration, 2019, pp. 1-11

[10] Dong et al. (2017). "The Mediating Role Of Resilience In Relationship Between Social Support And Posttraumatic Growth Among Colorectal Cancer Survivors With Permanent Intestinal Ostomies: A Structural Equation Model Analysis". **Eouropean Journal Of OncologyNursing**. https://dx.doi.org/10.1016/j.ejon.2017.04.007

[11] M. Pradikja, H. Tolle, and K. Brata, "Pengembangan Aplikasi Pembelajaran Bahasa Isyarat Berbasis Android Tablet," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2017, vol. 2, no. 8, pp. 2877-2885

[12] A. Halder and A. Tayade, "Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning," International Journal of Research Publication and Reviews, 2021, no. 2, pp. 9-17.

[13] Amei, W., D. Huailin, W. Qingfeng, and L. Ling, "A survey of application-level protocol identification based on machine learning," in International Conference on Information Management, Innovation Management and Industrial Engineering, 2011, pp. 201-204

[14] M.R. Amiarrahman and Tri Handika, "Analisis dan implementasi algoritma klasifikasi Random Forest dalam pengenalan Bahasa Isyarat Indonesia (BISINDO)," in Prosiding Seminar Nasional Inovasi Teknologi, 2017, pp. 83-88.

[15] A. Anilkumar, K.T. A., S. Sajan, and K.A. S., "Pose Estimated Yoga Monitoring System," SSRN Electronic Journal, Icicnis, 2021, pp. 1-8. [Online]. Available: https://doi.org/10.2139/ssrn.3882498/

[16] "MediaPipe," MediapipeDev, 2020. [Online]. Available: https://mediapipe.dev/

[17] D.J.P. Manajang, S.R.U.A. Sompie, and A. Jacobus, "Implementasi Framework Tensorflow Object Detection API Dalam Mengklasifikasi Jenis Kendaraan Bermotor," Jurnal Teknik Informatika, 2020, vol. 15, pp. 171-178

[18] M. AI Body Language Decoder using MediaPipe and Python. International Journal of Advance Research, Ideas and Innovations in Technology, 2021, vol. 7, no. 3, pp. 2436-2439.

[19] R.A. Mursita, "Respon Tunarungu Terhadap Penggunaan Sistem Bahasa Isyarat Indonesa (Sibi) Dan Bahasa Isyarat Indonesia (Bisindo) Dalam Komunikasi," Inklusi, 2015, vol. 2, no. 2, p. 221. [Online]

[20] A.S. Nugraheni, A.P. Husain, and H. Unayah, "Optimalisasi Penggunaan Bahasa Isyarat

Dengan SIBI dan BISINDO Pada Mahasiswa Difabel Tunarungu di Prodi PGMI UIN Sunan Kalijaga," Holistika, pp. 28-33. [Online]. Available: jurnal.umj.ac.id/index.php/holistika

[21] G. Sutrisnadipraj, N. Shesilia K, S. Putri F, Y. Yulianto, P. Handayani, and W.P. Sembiring, "Intervensi Psikoedukasi Dalam Mengatasi Stigma Dan Hambatan Komunikasi Pada Teman Tuli Yang Tergabung Dalam Gerkatin Kepemudaan," Jurnal Bakti Masyarakat Indonesia, 2018, vol. 2, no. 1. [Online]..