# PENGEMBANGAN SISTEM PENGENALAN BAHASA ISYARAT MENGGUNAKAN ALGORITMA *LONG SHORT TERM MEMORY* DENGAN *OUTPUT REALTIME TRANSLATION*

## LAPORAN TUGAS AKHIR

Laporan ini Disusun untuk Memenuhi Salah Satu Syarat Memperoleh
Gelar Sarjana Strata 1 (S1) pada Program Studi Teknik Informatika
Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang

**DISUSUN OLEH :**

**LANANG SHAKTI PRAYOGA**
**NIM 32602100062**

**PROGRAM STUDI TEKNIK INFORMATIKA**
**FAKULTAS TEKNOLOGI INDUSTRI**
**UNIVERSITAS ISLAM SULTAN AGUNG**
**SEMARANG**
**2025**

*FINAL PROJECT*

# *DEVELOPMENT OF A SIGN LANGUAGE RECOGNITION SYSTEM USING LONG SHORT TERM MEMORY ALGORITHM WITH REALTIME TRANSLATION OUTPUT*

*Submitted to fulfill the requirements to obtain a Bachelor's degree (S1) in the Informatics Engineering Department, Faculty of Industrial Technology, Sultan Agung Islamic University.*


**ARRANGED BY :**

**LANANG SHAKTI PRAYOGA**

**NIM 32602100062**

**MAJORING OF INFORMATICS ENGINEERING**

**INDUSTRIAL TECHNOLOGY FACULTY**

**SULTAN AGUNG ISLAMIC UNIVERSITY**

**SEMARANG**

**2025**

**LEMBAR PENGESAHAN**
**TUGAS AKHIR**

# PENGEMBANGAN SISTEM PENGENALAN BAHASA ISYARAT MENGGUNAKAN ALGORITMA LONG SHORT TERM MEMORY DENGAN OUTPUT REALTIME TRANSLATION

**LANANG SHAKTI PRAYOGA**
**NIM 32602100062**

Telah dipertahankan di depan tim penguji ujian sarjana tugas akhir
Program Studi Teknik Informatika
Universitas Islam Sultan Agung
Pada tanggal : 28 Februari 2025

**TIM PENGUJI UJIAN SARJANA :**

| | | |
|---|---|---|
| **Much Taufik, ST., MIT**<br>NIK. 210604034<br>(Ketua Penguji) | ..... | 28-02-2025 |
| **Bagus SWP, S.Kom, M.Cs**<br>NIDN. 1027118801<br>(Anggota Penguji) | ..... | 28-02-2025 |
| **Imam Much Ibnu Subroto, ST, M.Sc, Ph.D**<br>NIK. 210600017<br>(Pembimbing) | ..... | 28-02-2025 |

Semarang, 28 Februari 2025

Mengetahui,
Kaprodi Teknik Informatika
Universitas Islam Sultan Agung

**Much Taufik, ST., MIT**
NIK. 210604034

## SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan dibawah ini :

Nama : Lanang Shakti Prayoga

NIM : 32602100062

Judul Tugas Akhir : PENGEMBANGAN SISTEM PENGENALAN BAHASA ISYARAT MENGGUNAKAN ALGORITMA LONG SHORTTERM MEMORY DENGAN OUTPUT REAL-TIME TRANSLATION

Dengan bahwa ini saya menyatakan bahwa judul dan isi Tugas Akhir yang saya buat dalam rangka menyelesaikan Pendidikan Strata Satu (S1) Teknik Informatika tersebut adalah asli dan belum pernah diangkat, ditulis ataupun dipublikasikan oleh siapapun baik keseluruhan maupun sebagian, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka, dan apbila di kemudian hari ternyata terbukti bahwa judul Tugas Akhir tersebut pernah diangkat, ditulis ataupun dipublikasikan, maka saya bersedia dikenakan sanksi akademis. Demikian surat pernyataan ini saya buat dengan sadar dan penuh tanggung jawab.

Semarang, 27 Februari 2025

Yang Menyatakan,


Lanang Shakti Prayoga

## PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH

Saya yang bertanda tangan dibawah ini :

Nama : Lanang Shakti Prayoga

NIM : 32602100062

Program Studi : Teknik Informatika

Fakultas : Fakultas Teknologi industri

Dengan ini menyatakan Karya Ilmiah berupa Tugas akhir dengan Judul : PENGEMBANGAN SISTEM PENGENALAN BAHASA ISYARAT MENGGUNAKAN ALGORITMA LONG SHORT TERM MEMORY DENGAN OUTPUT REAL-TIME TRANSLATION Menyetujui menjadi hak milik Universitas Islam Sultan Agung serta memberikan Hak bebas Royalti Non-Eksklusif untuk disimpan, dialihmediakan, dikelola dan pangkalan data dan dipublikasikan diinternet dan media lain untuk kepentingan akademis selama tetap menyantumkan nama penulis sebagai pemilik hak cipta. Pernyataan ini saya buat dengan sungguh-sungguh. Apabila dikemudian hari terbukti ada pelanggaran Hak Cipta/Plagiarisme dalam karya ilmiah ini, maka segala bentuk tuntutan hukum yang timbul akan saya tanggung secara pribadi tanpa melibatkan Universitas Islam Sultan agung.

Semarang, 27 Februari 2025

Yang menyatakan,


Lanang Shakti Prayoga

# KATA PENGANTAR

Alhamdulillah, segala puji bagi Allah Ta'ala yang telah memberikan penulis rahmat dan karunianya yang luar biasa serta telah memberikan kekuatan serta memberikan memberikan kemudahan dalam menyelesaikan Tugas Akhir yang berjudul pengembangan sistem pengenalan bahasa isyarat menggunakan algoritma *long short term memory* dengan *output realtime translation*. Penyusunan laporan Tugas Akhir ini merupakan salah satu kewajiban untuk memperoleh gelar Sarjana S1 pada Program Studi Teknik Informatika Universitas Islam Sultan Agung Semarang.

Penulis menyadari bahwa selesainya laporan ini tidak lepas dari bimbingan, bantuan, saran serta fasilitas yang diberikan berbagai pihak. Oleh karenanya, pada kesempatan ini dengan segenap rendah hati, tak lupa penulis sampaikan rasa hormat dan terimakasih yang mendalam kepada :

1. Rektor Universitas Islam Sultan Agung Semarang Prof. Dr. H. Gunarto, S.H., M.H.
2. Dekan Fakultas Teknologi Industri Universitas Islam Sultan Agung, Dr. Ir. Novi Marlyana, S.T., M.T., IPU., ASEAN.Eng.
3. Ketua Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung, Moch Taufik, ST, MIT.
4. Koordinator Tugas Akhir Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung, Badieah, ST., M.Kom.
5. Dosen Pembimbing Imam Much Ibnu Subroto, ST, M.Sc, Ph.D atas waktu yang telah diluangkan dan bimbingan akademis yang telah diberikan hingga selesai nya Tugas Akhir ini.
6. Segenap Dosen Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung yang telah memberikan bimbingan, ilmu dan masukannya.
7. Orang Tua penulis yang terus mendoakan penulis hingga terselesainya laporan ini. Penulis menyadari bahwa tiada sesuatu hal pun di dunia ini yang sempurna begitu pula dengan laporan Tugas Akhir ini.

Semarang,

Lanang Shakti Prayoga

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# ABSTRAK

Komunikasi merupakan hak setiap manusia di seluruh dunia akan tetapi yang sangat disayangkan adalah masih banyak masyarakat non disabilitas yang belum memahami apa itu bahasa isyarat dan urgensinya bagi para penyandang disabilitas tuna rungu. Penelitian ini bertujuan untuk mengembangkan model deteksi BISINDO dengan bantuan mediapipe holistic dan menggunakan deep learning dengan algoritma *Long Short-Term Memory* (LSTM). Pada penelitian ini peneliti melakukan lima skenario berbeda untuk melihat performa model pada saat *training* dan *deployment*. Berdasarkan percobaan diperoleh hasil bahwa untuk skenario pertama mendapat skor *training* 99% dan skor *deployment* 100%, sedangkan untuk skenario kedua mendapat skor *training* 99% dan *deployment* 77.5%, untuk skenario ketiga mendapat skor *training* 99% dan *deployment* 42.5%, lalu untuk skenario keempat mendapat skor *training* 99% dan *deployment* 35.83% dan untuk skenario kelima mendapat skor *training* 99% dan *deployment* 17.5%. Berdasarkan hasil skor saat training dan deployment dapat disimpulkan bahwa skor evaluasi *training* tidak menentukan performa model pada saat *deployment*, pada tahap *deployment* performa prediksi model banyak dipengaruhi oleh variasi kelas yang harus diprediksi oleh model. Penelitian ini diharapkan dapat membantu masyarakat dalam memahami BISINDO serta berkontribusi terhadap pengembangan sistem penerjemahan bahasa isyarat berbasis kecerdasan buatan.


Kata kunci : Bahasa Isyarat Indonesia, *Deep Learning*, *Long Short Term Memory*, Mediapipe holistic



## *ABSTRACT*

*Communication is a fundamental right for every human being worldwide; however, it is unfortunate that many non-disabled individuals still lack an understanding of sign language and its urgency for the deaf community. Therefore, a technology-based solution is needed to enhance communication accessibility. This study aims to develop a BISINDO detection model utilizing Mediapipe Holistic and deep learning with the Long Short-Term Memory (LSTM) algorithm. In this research, the author conducted five different scenarios to evaluate the model's performance during training and deployment. The experimental results indicate that in the first scenario, the training score reached 99%, while the deployment score was 100%. In the second scenario, the training score was 99%, and the deployment score was 77.5%. The third scenario achieved a training score of 99% and a deployment score of 42.5%. In the fourth scenario, the training score was 99%, while the deployment score was 35.83%. Lastly, the fifth scenario obtained a training score of 99% and a deployment score of 17.5%. Based on the training and deployment evaluation scores, it can be concluded that training evaluation scores do not determine the model's performance during deployment. During deployment, the model's prediction performance is significantly influenced by the variation of classes that need to be predicted. This study is expected to help the public understand BISINDO and contribute to the development of sign language translation systems based on artificial intelligence.*


*Keywords : Indonesian sign language, Deep learning, Long Short Term Memory, Mediapipe holistic*

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Komunikasi merupakan kebutuhan dasar manusia untuk menyampaikan gagasan, isi pikiran bahkan perasaan. Namun sayangnya tidak semua orang memiliki kemampuan untuk berkomunikasi secara verbal. Umumnya keterbatasan ini disebabkan adanya kerusakan sebagai atau keseluruhan dari fungsi pendengaran. Sebagaimana diungkap Hallahan dan Kauffman, secara umum disabilitas rungu dikategorikan kurang dengar dan tuli, termasuk keseluruhan kesulitan mendengar dari yang ringan hingga berat. Tuli ialah orang yang kehilangan kemampuan mendengarnya sehingga terdapat hambatan proses penyampaian informasi melalui pendengaran, baik memakai alat bantu dengar maupun tidak. Sedangkan, kurang dengar adalah orang yang masih dapat menerima informasi yang disampaikan menggunakan alat bantu mendengar (Aisyah Muhammad Amin dkk., 2022).

Ketidaktahuan masyarakat umum tentang bahasa isyarat di Indonesia membatasi interaksi antara orang dengan gangguan pendengaran dan masyarakat umum. Bagi mereka dengan keterbatasan pendengaran, hal ini menciptakan kesenjangan komunikasi yang sangat besar. Penyandang disabilitas, termasuk disabilitas rungu dan wicara, memiliki hak, kewajiban, dan peran yang sama dengan orang lain di masyarakat Indonesia (Aisyah Muhammad Amin dkk., 2022). Oleh karena itu sudah menjadi hak mereka sebagai warga negara Indonesia untuk dapat menyampaikan gagasan dan pikiranya secara bebas dan tanpa dihalangi suatu apapun.

Perkembangan teknologi, terutama dalam bidang kecerdasan buatan dan machine learning, membuka peluang baru untuk memecahkan masalah ini. *Long Short-Term Memory* (LSTM) adalah salah satu algoritma yang paling relevan untuk memahami urutan dan pola dalam data. LSTM adalah jenis jaringan saraf tiruan yang memiliki kemampuan untuk mengenali dan 2 memproses data sekuensial, seperti gerakan tangan yang berurutan dalam

bahasa isyarat. Oleh karena itu, LSTM cocok untuk mengidentifikasi Gerakan isyarat dan menerjemahkannya ke dalam teks secara langsung. Untuk meningkatkan performa dari model yang akan dibuat pustaka mediapipe juga akan coba digunakan untuk melakukan ekstraksi dari gambar video sehingga inputan yang diterima oleh LSTM menjadi lebih ringan dan dengan hal ini diharapkan selain performa model yang meningkat juga akan meningkatkan kecepatan model dalam memproses data.

Sistem pengenalan bahasa isyarat berbasis LSTM dengan *output* terjemahan *real-time* diharapkan dapat menjembatani kesenjangan komunikasi antara masyarakat umum dan pengguna bahasa isyarat. Mereka dapat mengenali rangkaian gerakan isyarat dan mengonversinya menjadikosa kata yang dimengerti. Oleh karena itu, solusi ini tidak hanya membantu komunikasi yang inklusif, tetapi juga membantu penyandang disabilitas pendengaran menjadi lebih mudah di berbagai bidang, seperti pendidikan, layanan publik, dan pekerjaan.

## 1.2 Rumusan Masalah

Bagaimana algoritma *Long Short Term Memory* dengan bantuan mediapipe holistic mampu mendeteksi dan melakukan klasifikasi terhadap bahasa isyarat secara real time.

## 1.3 Batasan Masalah

Adapun pembatasan masalah pada penulisan proposal sebagai berikut :

1. Penelitian ini berfokus dan terbatas pada pembuatan model untuk mendeteksi kombinasi 40 kosa kata dan alfabet bahasa isyarat Indonesia (BISINDO)
2. Dataset yang digunakan merupakan dataset yang diambil secara langsung menggunakan OpenCV.

## 1.4 Tujuan

1. Mengembangkan model python yang dapat mendeteksi bahasa isyarat secara *realtime* dengan menerapkan algoritma LSTM.
2. Mengukur kinerja algoritma LSTM dalam melakukan deteksi secara *real time.*

## 1.5 Manfaat

Manfaat yang didapat dari penelitian ini adalah membuktikan bahwa algoritma LSTM cukup efektif dalam melakukan deteksi bahasa isyarat dengan bentuk data berupa hasil konversi dari landmark mediapipe.

## 1.6 Sistematika Penulisan

Berikut adalah sistematika penulisan yang digunakan dalam Menyusun laporan tugas akhir :

BAB I : Pada bab 1 mengutarakan tentang latar belakang penelitian, rumusan masalah penelitian, Batasan masalah penelitian, tujan penelitian, dan manfaat penelitian.

BAB II : Pada bab 2 berisi penelitian-penelitian terdahulu dan juga dasar teori yang berhubungan dengan algoritma LSTM sehingga dapat membantu penyelesaian persoalan pada penelitian ini.

BAB III : Pada bab 3 akan berisi tentang proses pelaksanaan penelitian mulai dari mengumpulkan dataset hingga melatih model dengan akurasi terbaik.

BAB IV : Pada bab 4 akan memaparkan hasil dari penelitian yang telah dilakukan, hasil ini mencakup model yang telah dibuat, evaluasi akurasi yang didapat, dan juga implementasi dari model.

BAB V : Pada bab 5 akan menjadi bagian penutup laporan yang mana akan berisi Kesimpulan dari penelitian yang sudah dilakukan dan saran untuk penelitian yang dilakukan kedepanya.

# BAB II
# TINJAUAN PUSTAKA DAN DASAR TEORI

## 2.1 Tinjauan Pustaka

Untuk mendukung penelitian mengenai deteksi bahasa isyarat, terdapat beberapa sumber terdahulu yang berkaitan secara langsung maupun tidak langsung dengan topik penelitian ini, diantaranya :

Penelitian yang dilakukan oleh (Nyoman Tri Anindia Putra dkk., 2023) mengenai Penerapan *Library* Tensorflow, Cvzone, dan Numpy pada Sistem Deteksi Bahasa Isyarat Secara *Real Time*. *Library* yang digunakan pada penelitian tersebut antara lain adalah tensorflow, cvzone, dan numpy. Tujuan dari penelitian tersebut adalah untuk membuat sistem yang bisa mendeteksi Bahasa isyarat yaitu "Deteksi Bahasa Isyarat Menggunakan Metode Detection and Classification Secara *Real Time*" Dataset yang digunakanpada penetilian adalah 26 isyarat alfabet bisindo yang masing-masing data tersebut akan dilakukan uji coba sebanyak 20 kali. Metode yang digunakan adalah *detection and classification* dengan bantuan Pustaka cvzone. Hasil penelitian yang didapat adalah dari 26 alfabet terdapat 4 alfabet yang tidak terdeteksi karena faktor cahaya pada saat melakukan uji coba, selain itu ada beberapa isyarat tangan yang memiliki bentuk mirip dengan yang lainya sehingga didapat Tingkat akurasi 47.5%.

Penelitian yang dilakukan oleh (Nababan & Budiarso, 2023) mengenai Sistem Pendeteksi Gerakan Bahasa Isyarat Indonesia Menggunakan Webcam dengan Metode *Supervised Learning*. *Library* yang digunakan pada penelitian ini antara lain adalah opencv, labelImg, pillow, tensorflow, dan untuk implementasi *machine learning* menggunakan model yang diharapkan dapat memberikan hasil deteksi yang akurat dan konsisten menggunakan SSD ResNet50. Tujuan Tujuan dari penelitian ini adalah membangun dan merancang Sistem Pendeteksi Gerakan Bahasa Isyarat Indonesia dengan Metode *Supervised Learning* yang bisa dimanfaatkan untuk keperluan komunikasi masyarakat normal dengan penderita tuna rungu-wicara. Dataset

yang digunakan merupakan 20 bahasa isyarat Indonesia (BISINDO) yaitu : Macet, Musuh, Tiba, Tidak Ada, Dua, Satu, Diam, Kehidupan, Selesai, Mau, Hatimu, Teman, Ya, Wanita, Usia, Pria, Aku, Mencintai, Kamu. Dengan bentuk data grayscale 150x150 dengan total jumlah data sebanyak 60 gambar jpg yang terbagi menjadi test, validation, dan training. Metode yang digunakan dalam penelitian ini adalah *supervised learning*. Hasil dari penelitian adalah memperoleh Tingkat akurasi sebesar 90% dari 60 kali percobaan.

Penelitian yang dilakukan oleh (Inayatul Arifah dkk., 2022) mengenai Deteksi Tangan Otomatis Pada Video Percakapan Bahasa Isyarat Indonesia Menggunakan Metode YOLO Dan CNN. Tujuan dari penelitian ini adalah untuk membandingkan akurasi dalam membaca bahasa isyarat antara metode yolo dan CNN. Dataset yang digunakan adalah video bahasa isyarat Indonesia dan beberapa video dari relawan, untuk pengambilan dilakukan didalam dan luar ruangan untuk meningkatkan variasi data. Metode yang digunakan pada penelitian ini adalah yolo dan CNN yang diharapkan dapat menghasilkan akurasi yang baik. Hasil dari penelitian ini model terbaik diraih dengan menggabungkan kedua metode yaitu yolo dan cnn, serta untuk akurasi yang diraih mencapai 89% dengan jumlah 75 epoch, 5 batch.

Penelitian yang dilakukan oleh (Hayyu Gustsa & Setyo Permadi, 2022) mengenai sistem deteksi bahasa isyarat secara *realtime* dengan tensorflow *object detection* dan python menggunakan metode *convolutional neural network*. Penelitian ini bertujuan untuk membantu orang-orang yang berkebutuhan khusus tanpa memerlukan cara yang memerlukan biaya yang sangat mahal yaitu menggunakan penterjemah manusia, dengan sistem deteksi bahasa isyarat secara *real time* ini orang-orang yang tidak tahu bahasa isyarat akan paham dengan melihat sistem yang akan menerjemahkan gerakan tangan atau isyarat ke dalam bentuk text sehingga orang-orang akan mengerti apa yang ingin disampaikan oleh orang yang memiliki kebutuhan khusus. Dataset yang digunakan merupakan gambar bahasa isyarat yang diambil langsung antara lain : halo, ya, tidak, cinta, terimakasih dengan total

200 gambar yang terbagi menjadi 150 training dan 50 testing. Metode yang digunakan pada penelitian ini adalah CNN yang merupakan pengembangan dari metode multilayer perceptron yang dikenal cukup baik dalam menerima inputan berupa data gambar. Hasil dari penelitian ini adalah mendapat hasil akurasi sebesar 93.33%.

Penelitian yang dilakukan oleh (Suyudi dkk., 2023) mengenai Pengenalan Bahasa Isyarat Indonesia menggunakan Mediapipe dengan *Model Random Forest* dan *Multinomial Logistic Regression*. Tujuan dari penelitian ini adalah mengukur kinerja dari algoritma random forest dan multinomial logistic regression dalam membaca inputan bahasa isyarat dengan bantuan mediapipe. Dataset yang digunakan adalah alfabet bahasa isyarat SIBI dengan pengecualian huruf J dan Z, setiap kelas akan memiliki 1000 data dengan 21 landmark dan koordinat X, Y, Z. Metode yang digunakan pada penelitian ini adalah *logistic regression* dan *random forest*, yang mana untuk *logistic regeression* sendiri merupakan algoritma klasifikasi biner sedangkan *random forest* pada *root* awal berupa binari akan tetapu semakin berkembang seiring dengan percabangan yang ada. Penelitian ini berhasil membuat model *random forest* dan *logistic regression* dengan akurasi tinggi untuk pengenalan bahasa isyarat indonesia. Pembuatan dataset dengan hanya menyimpan 21 titik kerangka tangan menggunakan kerangka kerja MediaPipe terbukti efektif. Untuk model *random forest* pada *tree* ke 100 mendapat nilai akurasi 97% sedangkan untuk model logistic regression mencapai akurasi 96% dengan maksimum iterasi 250.

## 2.2 Dasar Teori

### 2.2.1 Bahasa Isyarat Indonesia (BISINDO)

Bahasa isyarat merupakan bentuk komunikasi yang memanfaatkan gerakan tubuh dan ekspresi wajah sebagai simbol untuk menyampaikan makna dari bahasa lisan (Mursita dkk., t.t.) ahasa ini terutama digunakan oleh individu Tunarungu, dengan cara mengombinasikan bentuk dan Gerakan telapak tangan, gerakan lengan, posisi tubuh, serta ekspresi wajah untuk

mengekspresikan gagasan mereka. Berdasarkan hal tersebut, bahasa isyarat dapat didefinisikan sebagai sistem komunikasi yang menggunakan Gerakan tubuh dan mimik wajah, khususnya untuk kebutuhan komunitas Tunarungu. Di Indonesia, terdapat dua jenis bahasa isyarat yang umum digunakan, yaitu SIBI (Sistem Isyarat Bahasa Indonesia) dan BISINDO (Bahasa Isyarat Indonesia).

Bahasa isyarat tidak bersifat universal setiap negara atau komunitas dapat memiliki sistem bahasa isyarat yang berbeda, seperti *American Sign Language* (ASL), *British Sign Language* (BSL), dan Bahasa Isyarat Indonesia (BISINDO). Sebagai bahasa alami, bahasa isyarat memiliki struktur tata bahasa, sintaksis, dan semantik yang setara dengan bahasa verbal. Lebih dari sekadar alat komunikasi, bahasa isyarat juga mencerminkan budaya dan identitas komunitas Tuli di seluruh dunia. Hal ini menjadikan bahasa isyarat tidak hanya penting sebagai sarana komunikasi, tetapi juga sebagai bagian integral dari hak budaya dan sosial komunitas yang menggunakannya.

Pada tahun 1994 pemerintah memutuskan untuk membuat kamus Sistem Isyarat Bahasa Indonesia (SIBI), akan tetapi hal ini menuai perselisihan diantara komunitas tuli karena SIBI dinilai terlalu sistematis dan tidak sesuai dengan apa yang digunakan oleh kaum tuli dilapangan secara langsung, oleh karena itu muncul variasi bahasa isyarat lain yang dinilai lebih mewakili Indonesia dalam bahasa isyarat yaitu Bahasa Isyarat Indonesia atau yang sekarang akrab disapa BISINDO.

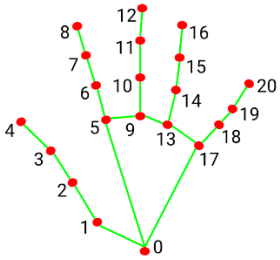
Bahasa Isyarat Indonesia (BISINDO) merupakan sistem komunikasi visual yang dikembangkan oleh komunitas Tuli di Indonesia sebagai sarana untuk menyampaikan informasi dan mengekspresikan gagasan. Sebagai bahasa alami, BISINDO memiliki struktur tata bahasa dan kosakata yang khas, yang membedakannya dari bahasa verbal maupun sistem isyarat lainnya. Penggunaan BISINDO tidak hanya merepresentasikan kebutuhan komunikasi, tetapi juga mencerminkan identitas budaya komunitas Tuli di Indonesia.

### 2.2.2 Sistem Deteksi Gerak Tangan

Teknologi baru yang memperkenalkan gerakan tangan (*hand tracking*) dapat menggantikan media pembelajaran bahasa isyarat yang sebelumnya digunakan dalam aplikasi. Pelacakan tangan merupakan metode pendeteksian pergerakan tangan dengan menggunakan kamera video. Keunggulan teks isyarat tangan adalah memberikan keunggulan utama yaitu ekonomis, alur prosedural, dan akurasi tinggi. Salah satu faktor utama munculnya teknologi ini adalah kurangnya kesadaran masyarakat tentang pembelajaran bahasa isyarat, karena dianggap sulit untuk dipahami dan tidak digunakan oleh masyarakat umum, oleh karena itu dengan adanya teknologi ini diharapkan dapat mengurangi hal tersebut dan membuat komunikasi menjadi lebih inklusif.

### 2.2.3 Mediapipe

Mediapipe adalah *Framework* yang dikembangkan oleh Google untuk memproses dan menganalisis data multimedia seperti gambar, video, dan audio secara real time (Hasyim Nur'azizan dkk., t.t.) Mediapipe menggunakan serangkaian modul yang dioptimalkan untuk menyediakan saluran yang efisien untuk berbagai aplikasi visi komputer seperti pengenalan wajah, tangan, pose, dan objek. Mediapipe mendukung berbagai platform seperti desktop, seluler, dan web, sehingga memungkinkan pemrosesan lintas platform. Arsitektur modular Mediapipe memudahkan pengembang membangun aplikasi yang menggunakan teknologi pengenalan visual secara efisien. Hal ini sering dikombinasikan dengan pembelajaran mesin seperti Tensorflow untuk mendukung aplikasi seperti *augmented reality*, pengenalan isyarat, dan interaksi berbasis isyarat (Suyudi dkk., 2023).



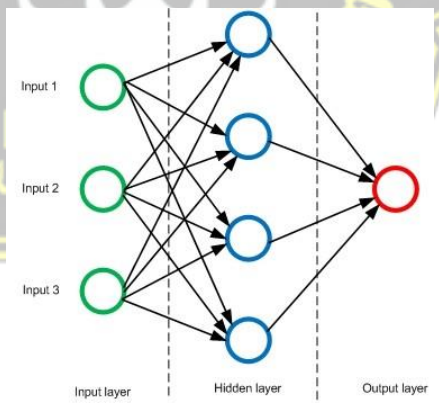

0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Gambar 2. 1 *Landmark* mediapipe

### *2.2.4 Long Short Term Memory*

*Long Short Term Memory* (LSTM) adalah arsitektur jaringan saraf berulang (RNN) yang dirancang untuk mengatasi masalah gradien hilang yang biasa ditemui di RNN standar saat memproses data berurutan dengan ketergantungan jangka panjang. RNN (*Recurrent Neural Network*) adalah jenis jaringan saraf yang memiliki koneksi siklik (Wen & Li, 2023), memungkinkan penyimpanan dan akses informasi dari waktu sebelumnya selama pengolahan data. Dengan arsitektur ini, model dapat membuat keputusan yang lebih cerdas karena mampu mempertimbangkan konteks sebelumnya dalam memproses setiap elemen dalam suatu urutan.

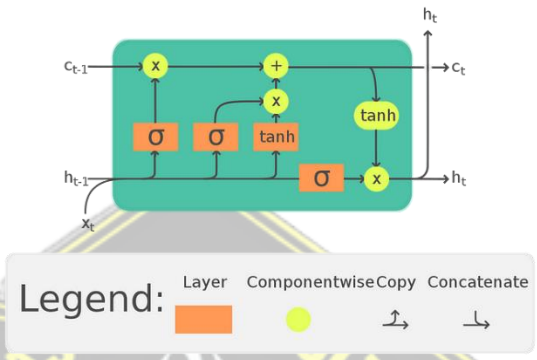
Namun, RNN memiliki keterbatasan dalam menangani ketergantungan jarak jauh pada data urutan, karena hanya dapat melihat sinyal sebelumnya hingga sekitar 10 langkah. Hal ini disebabkan oleh masalah *vanishing gradient*, yang membuat RNN sulit mengingat informasi dari waktu yang lebih lama. Akibatnya, model hanya mampu menyimpan informasi terbatas dari waktu sebelumnya. Kelemahan ini telah diatasi dengan pengembangan LSTM (*Long Short-Term Memory*).




Gambar 2. 2 *Layer* RNN

Pada gambar 2.2 menampilkan lapisan layer yang dimiliki oleh RNN sedangkan pada LSTM sendiri memiliki struktur berupa gerbang (gerbang input, gerbang lupa, gerbang keluaran) yang mengontrol aliran informasi dalam jaringan (Tri Hermanto dkk., t.t.). Struktur ini memungkinkan LSTM untuk menyimpan, menghapus, dan memperbarui informasi dalam "memori" internalnya, memungkinkan jaringan mengingat informasi penting dari

langkah sebelumnya dalam urutan tersebut dan melupakan informasi yang kurang relevan. Oleh karena itu, LSTM sangat efektif untuk tugas-tugas yang melibatkan data berkelanjutan, seperti pengenalan ucapan, pemrosesan bahasa alami, dan analisis deret waktu, di mana informasi konteks dari data sebelumnya penting untuk prediksi yang akurat.




Gambar 2. 3 Komponen *layer* LSTM

Gambar 2.3 menampilkan komponen layer pada LSTM, komponen gerbang digunakan untuk mengontrol informasi yang masuk ke memori, yang bertanggung jawab untuk menyelesaikan masalah kerugian dan pembagian. Koneksi berulang menambah status atau memori ke jaringan dan memungkinkan jaringan menggunakan observasi dalam urutan. Memori internal berarti keluaran jaringan bergantung pada konteks terakhir dalam antrean masukan, bukan masukan yang disajikan sebagai jaringan. Ada 3 macam gerbang dalam LSTM yaitu *forget gate, input gate,* dan *output gate.*

*Forget gat*e berfungsi untuk menentukan informasi mana yang akan disimpan atau dihapus. Gerbang ini menerima dua jenis informasi: *hidden state* dari sel sebelumnya dan input dari waktu saat ini. Informasi tersebut digabungkan dan diproses menggunakan fungsi sigmoid, yang menghasilkan nilai antara 0 hingga 1. Jika nilai mendekati 0, informasi akan dihapus, sedangkan nilai yang mendekati 1 menunjukkan bahwa informasi tersebut akan disimpan.

*Input gate* bertugas mengolah hidden state dari sel sebelumnya dan input baru dari waktu saat ini. Kedua informasi ini digabungkan, lalu diproses menggunakan fungsi sigmoid dan tanh. Fungsi sigmoid menghasilkan nilai

antara 0 hingga 1 untuk menentukan informasi yang akan diperbarui, di mana nilai mendekati 0 menandakan informasi tidak penting, sedangkan nilai mendekati 1 menunjukkan informasi penting. Fungsi tanh menghasilkan nilai antara -1 hingga 1, yang membantu sel memori dalam mempelajari informasi secara lebih efektif.

*Output gate* bertugas menentukan *hidden state* yang akan diteruskan ke sel berikutnya. Gerbang ini menerima *hidden stat*e dari sel sebelumnya serta input baru dari waktu saat ini. Informasi tersebut digabungkan dan diproses menggunakan fungsi sigmoid. Selanjutnya, *cell state* yang diperbarui diproses melalui fungsi tanh. Hasil dari fungsi tanh dikalikan dengan keluaran fungsi sigmoid untuk menghasilkan informasi yang akan disimpan dalam *hidden state* baru. *Hidden state* dan *cell state* yang telah diperbarui kemudian dikirim ke sel berikutnya.

### 2.2.5 OpenCV

OpenCV (*Open Source Computer Vision Library*) adalah perpustakaan sumber terbuka untuk pemrosesan gambar dan aplikasi visi komputer(Hasyim Nur'azizan dkk., t.t.). OpenCV dikembangkan oleh Intel dan pertama kali dirilis pada tahun 2000. Ini menyediakan berbagai alat dan algoritma untuk pemrosesan gambar, deteksi objek, pengenalan wajah, analisis video, dan tugas berbasis visi komputer lainnya. OpenCV mendukung berbagai bahasa pemrograman seperti Python, C++, Java, dan MATLAB serta bekerja di berbagai platform seperti Windows, macOS, Linux, Android, dan iOS. Salah satu fitur unggulannya adalah dapat memproses gambar dan video dalam berbagai format, serta mendukung akselerasi perangkat keras dengan GPU. OpenCV banyak digunakan sebagai alat pengembangan visi komputer berbasis teknologi di industri dan penelitian untuk aplikasi seperti pengenalan wajah, deteksi gerakan, pelacakan objek, dan *augmented reality*.

### 2.2.6 Python

Python adalah bahasa pemrograman tingkat tinggi yang populer karena sintaksisnya yang sederhana, fleksibilitas, dan dukungan luas untuk

paradigma pemrograman yang berbeda. Bahasa ini banyak digunakan dalam ilmu data dan pembelajaran mesin karena ekosistem perpustakaannya yang kaya seperti NumPy, Pandas, Matplotlib, dan Scikit-learn yang menyederhanakan manipulasi dan analisis data. Python juga cocok untuk mengembangkan model pembelajaran mesin menggunakan TensorFlow, salah satu perpustakaan pembelajaran mendalam Google yang paling popular.

TensorFlow memiliki API yang kuat dan mudah digunakan dengan Python yang dapat digunakan pengembang untuk membangun, melatih, dan menguji model jaringan saraf dan algoritma pembelajaran mesin lainnya secara lebih efisien. Sintaks Python yang mudah dipahami memungkinkan pengembang untuk fokus bereksperimen dengan algoritma dan model, daripada kompleksitas bahasa. Selain itu, Python memiliki komunitas yang besar dan aktif, dengan tutorial ekstensif, dokumentasi, dan dukungan di berbagai forum, sehingga sangat membantu dalam pemecahan masalah baik bagi pemula maupun pakar. Python dapat digunakan di berbagai platform pada skala pengembangan dan produksi, menjadikannya pilihan ideal untuk mengembangkan aplikasi pembelajaran mesin, terutama saat menggunakan TensorFlow untuk pembelajaran mendalam.

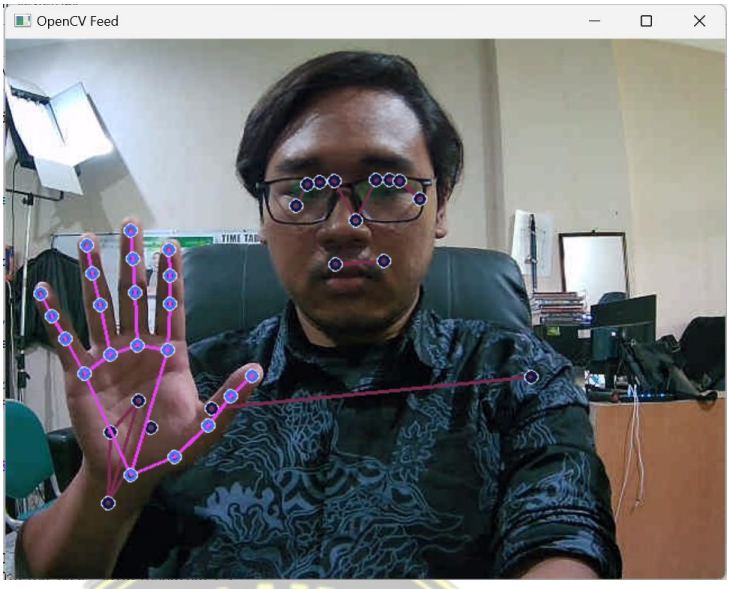
# BAB III
# METODE PENELITIAN

## 3.1 Metode Penelitian

### 3.1.1 Setup Mediapipe dan OpenCV

Pada tahap ini Langkah awal yang harus dilakukan adalah mengintegrasikan antara opencv dan mediapipe, karena secara bawaan kedua *library* tersebut tidak langsung terhubung dan membutuhkan beberapa konfigurasi agar keduanya dapat berjalan berdampingan. Mediapipe yang digunakan dalam penelitian ini adalah mediapipe holistic yang mana varian mediapipe ini akan mendeteksi 3 jenis landmark sekaligus yaitu landmark pada wajah, telapak tangan, dan persendian tubuh. Dengan digunakanya mediapipe ini diharapkan dapat menghasilkan model yang akurat dalam memprediksi bahasa isyarat.


Gambar 3. 1 Sebelum integrasi mediapipe




Gambar 3. 2 Sesudah integrasi mediapipe

Pada gambar 3.1 dan gambar 3.2 menampilkan perbandingan sebelum dan sesudah melakukan integrasi antara opencv dan mediapipe, dapat dilihat pada gambar 3.1 citra yang ditangkap oleh opencv masih berupa gambar langsung dari webcam tanpa adanya landmark mediapipe, sedangkan pada gambar 3.2 setelah melakukan integrasi antara opencv dan mediapipe citra yang ditangkap oleh opencv sudah dikombinasikan dengan landmark mediapipe. Sehingga dengan adanya landmark yang sudah tampil menandakan bahwa integrasi antara opencv dan mediapipe sudah berhasil dan ekstraksi landmark mediapipe ditahap berikutnya dapat dilaksanakan.

### 3.1.2 Pengambilan Dataset

Setelah opencv dan mediapipe dapat berjalan berdampingan Langkah berikutnya adalah melakukan pengambilan data melalui opencv dengan cara mengambil landmark yang dihasilkan oleh mediapipe. Secara bawaan ketika mengaktifkan salah satu fitur yang dimiliki mediapipe yaitu landmark, maka mediapipe akan memberikan output berupa vector tiga dimensi, yang mana setiap vector tersebut akan mewaliki sumbu x, y, dan z.

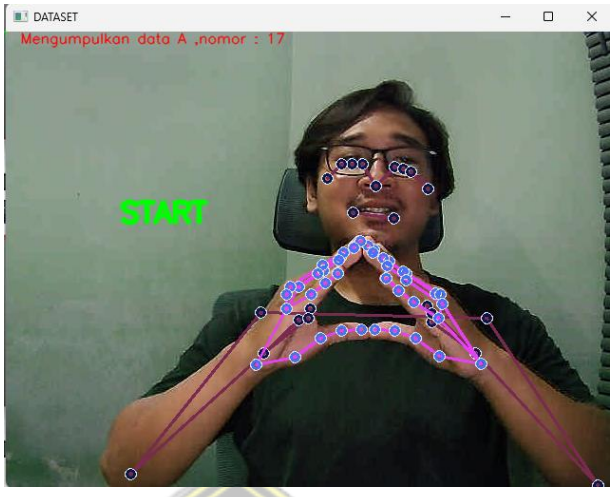
```
[27]: [x: 0.548091471
       y: 0.867001653
       z: -1.66299174e-007
       , x: 0.493345886
       y: 0.834640086
       z: -0.00556521071
       , x: 0.450162053
       y: 0.764209509
       z: -0.0114799524
       , x: 0.41792053
       y: 0.723842442
       z: -0.0204288103
```

Gambar 3. 3 *Landmark* sebelum konversi

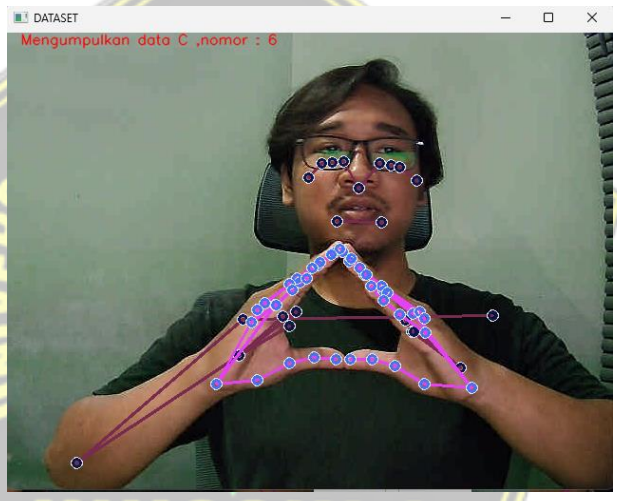
```
array([ 5.48091471e-01,  8.67001653e-01, -1.66299174e-07,  4.93345886e-01,
        8.34640086e-01, -5.56521071e-03,  4.50162053e-01,  7.64209509e-01,
       -1.14799524e-02,  4.17920530e-01,  7.23842442e-01, -2.04288103e-02,
        3.86251926e-01,  6.88320994e-01, -3.01583242e-02,  4.98702168e-01,
        6.71710312e-01, -7.70880794e-03,  4.58717018e-01,  6.19855523e-01,
       -3.35439406e-02,  4.17093396e-01,  6.03764832e-01, -5.52235283e-02,
        3.80956799e-01,  6.03773832e-01, -6.94141090e-02,  5.21820009e-01,
        6.71899676e-01, -2.26952806e-02,  4.73096102e-01,  6.10169590e-01,
       -5.01463376e-02,  4.18527663e-01,  6.09988451e-01, -7.01727197e-02,
        3.76914948e-01,  6.24107540e-01, -8.14781040e-02,  5.45133471e-01,
        6.90026999e-01, -4.01068777e-02,  4.98648226e-01,  6.34539425e-01,
       -7.06367269e-02,  4.45473999e-01,  6.43239081e-01, -8.56431127e-02,
        4.05736685e-01,  6.57893836e-01, -9.13553759e-02,  5.62408924e-01,
        7.25462258e-01, -5.75672016e-02,  5.24028599e-01,  6.95799470e-01,
       -8.51626769e-02,  4.90156710e-01,  7.02755630e-01, -9.43885893e-02,
        4.65955853e-01,  7.13792980e-01, -9.72270966e-02])
```

Gambar 3. 4 *Landmark* sesudah konversi

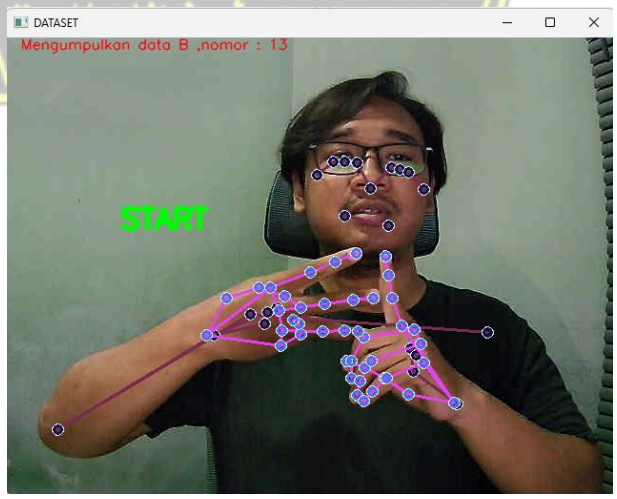
Pada gambar 3.3 merupakan *output* awal dari mediapipe sebelum menerapkan fungsi flatten untuk menyederhanakanya, yang mana dapat dilihat untuk format awal yag diberikan oleh mediapipe merupakan vector tiga dimensi, kemudian pada gambar 3.4 merupakan data *landmark* mediapipe setelah menerapkan fungsi flatten sehingga data dapat diterima oleh model sebagai data untuk proses pelatihan model.



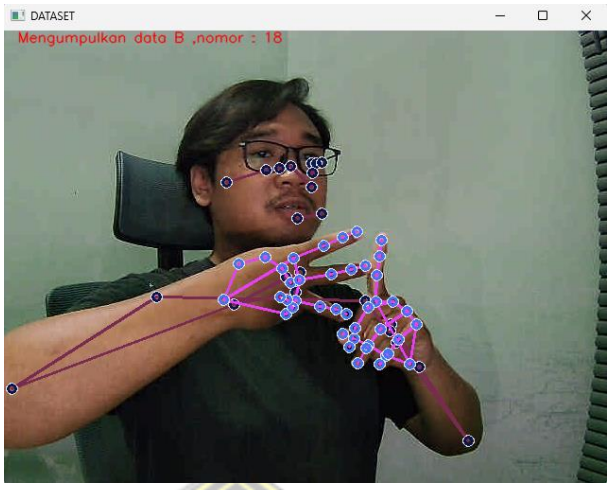

Gambar 3. 5 Pengambilan data alfabet A (1)



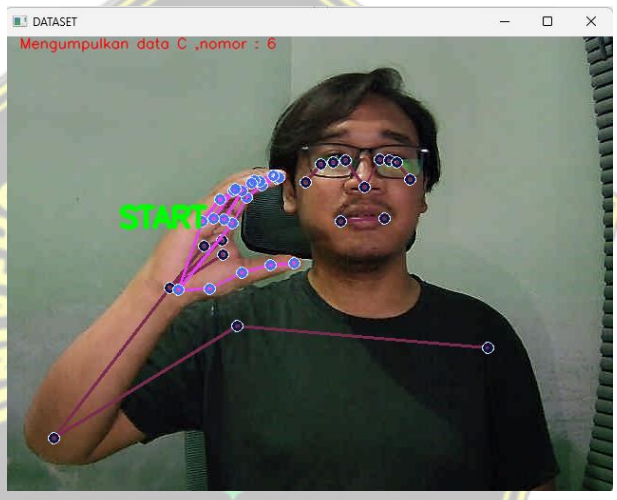

Gambar 3. 6 Pengambilan data alfabet A (2)



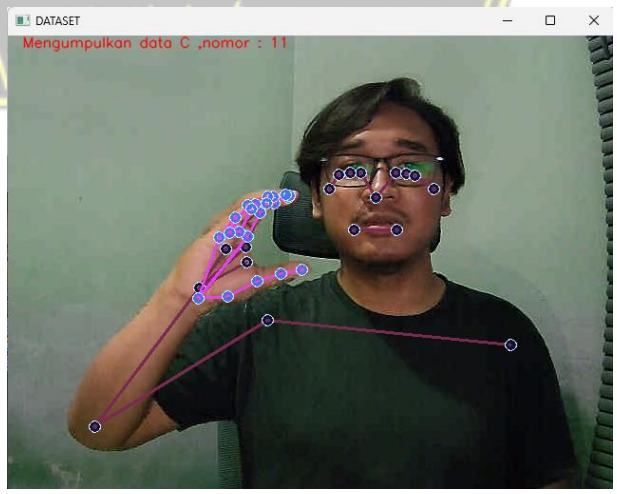

Gambar 3. 7 Pengambilan data alfabet B (1)




Gambar 3. 8 Pengambilan data alfabet B (2)




Gambar 3. 9 Pengambilan data alfabet C (1)




Gambar 3. 10 Pengambilan data alfabet C (2)

Pada gambar 3.5 dan gambar 3.6 menampilkan demonstrasi pengambilan dataset untuk alfabet A, sedangakan pada gambar 3.7 dan gambar 3.8 menampilkan contoh pengambilan dataset untuk alfabet B, dan yang terakhir pada gambar 3.9 dan gambar 3.10 merupalan contoh demonstrasi untuk pengambilan dataset alfabet C. Pada proses demonstrasi pengambilan data tersebut peneliti memberikan contoh dua skenario pada setiap pengambilan data, hal ini untuk menunjukkan bahwa dataset yang diambil oleh peneliti bukan merupakan data yang sama dalam hal ini makin banyak variasi dataset yang ada maka akan meningkatkan referensi model ketika melakukan pelatihan.

### 3.1.3 Labeling Dataset

Setelah dataset terkumpul sesuai dengan jumlah dan kriteria yang diinginkan, Langkah berikutnya adalah melakukan labelling pada dataset. Labeling dataset berfungsi untuk mempermudah ketika dataset diterima oleh model untuk melakukan pelatihan, ketika dataset sudah melewati tahap labelling maka ketika dataset diterima oleh model sebagai bahan untuk melakukan pelatihan maka model akan langsung membaca dataset sesuai dengan label yang tertera padanya dan bukan data mentah seperti apabila dataset belum melalui proses labelling. Pada penelitian ini labelling dilakukan dengan cara mendefinisikan dahulu isi dari dataset yang diberi label, sebagai contoh apabila dataset yang diinputkan merupakan data alfabet maka sebelum melakukan labelling Langkah yang harus dilakukan adalah mendefinisikan alfabet dari mulai A hingga Z. Setelah isi dari dataset didefinisikan dan disimpan dalam satu variabel Langkah berikutnya adalah melakukan inisiasi variabel untuk menyimpan data yang sudah diberi label, sebagai contoh didalam dataset menandung huruf A, B, dan C maka dengan memberikan label kepada dataset model akan membaca dataset sebagai A = 0, B = 1, dan C = 2 dan seterusnya hingga semua data sudah berhasil diberi label nomor untuk memudahkan dalam pelatihan model.

### 3.1.4 Pelatihan Model

Begitu semua dataset sudah diberikan label Langkah berikutnya adalah membangun model untuk dilatih menggunakan dataset yang ada. Sebelum melanjutkan kepada Pembangunan model dataset harus dibagi terlebih dahulu menjadi dua bagian yaitu untuk *training* dan *testing*.

```
Shape of X_train: (3200, 30, 258)
Shape of y_train: (3200,)
Shape of X_test: (800, 30, 258)
Shape of y_test: (800,)
```

Gambar 3. 11 Pembagian dataset

Gambar 3.11 menunjukkan bahwa dataset sudah berhasil dibagi dengan perbandingan 80% dataset untuk training dan 20% dataset untuk melakukan testing, pada tahap ini untuk membagi dataset menjadi dua bagian penulis menggunakan bantuan dari fungsi train_test_split yang didapat dari *library* sklearn, dengan menggunakan fugnsi train_test_split data set dapat dibagi menjadi dua bagian dengan proporsi sesuai dengan kebutuhan penelitian.

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 30, 64)            442112

lstm_1 (LSTM)                (None, 32)                12416

dense (Dense)                (None, 16)                528

dense_1 (Dense)              (None, 26)                442

=================================================================
Total params: 455498 (1.74 MB)
Trainable params: 455498 (1.74 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Gambar 3. 12 Ringkasan perencanaan model

Pada gambar 3.12 menampilkan ringkasan arsitektur model jaringan LSTM berlapis, yang terdiri dari dua lapisan LSTM berurutan dengan lapisan pertama memiliki 64 unit dengan *return_sequences=True* untuk mempertahankan urutan data, diikuti oleh lapisan kedua dengan 32 unit tanpa return_sequences. Setelahnya, terdapat dua lapisan Dense, yakni lapisan dengan 16 unit dan aktivasi ReLU untuk ekstraksi fitur, serta lapisan output dengan aktivasi softmax yang digunakan untuk klasifikasi multi-kelas sesuai dengan jumlah kelas yang ada didalam dataset.

Pada tahap training ini penulis akan mencoba melatih model dengan lima scenario kelas dataset berbeda yaitu dengan menggunakan 3 kelas, 10 kelas, 20 kelas, 30 kelas, dan 40 kelas. Hal ini dilakukan untuk melihat apakah akurasi yang dihasilkan ketika proses *training* dan *testing* dipengaruhi oleh banyaknya kelas data atau tidak.

Tabel 3. 1 Perencanaan lima skenario pelatihan

| No | Model | Kelas dataset | Learning Rate |
|---|---|---|---|
| 1 | LSTM | 3 | 0.001 |
| 2 | LSTM | 10 | 0.001 |
| 3 | LSTM | 20 | 0.0005 |
| 4 | LSTM | 30 | 0.0005 |
| 5 | LSTM | 40 | 0.0005 |

Pada tabel 3.1 merupakan lima skenario yang akan dicoba untuk pelatihan model ini, dengan menerapkan kelima skenario ini penulis dapat melihat apakah dengan makin banyaknya kelas hasil pelatihan dan pengujian akan mengalami penurunan atau tetap bertahan diangka yang sama.

### 3.1.5 Evaluasi

Evaluasi model bertujuan untuk mengidentifikasi kombinasi model terbaik dalam mengklasifikasikan bahasa isyarat dengan tingkat akurasi yang tinggi. Untuk itu, metrik akurasi dan Confusion Matrix digunakan dalam membandingkan performa model yang telah dilatih. Akurasi mengukur proporsi prediksi yang benar terhadap seluruh data yang digunakan dalam pelatihan. Sementara itu, Confusion Matrix merupakan alat yang sering digunakan dalam machine learning untuk menilai kinerja model dalam tugas klasifikasi, baik biner maupun multikelas. Dalam Confusion Matrix, kolom merepresentasikan kelas yang diprediksi, sedangkan baris menunjukkan kelas sebenarnya, sehingga memungkinkan perhitungan berbagai kemungkinan dalam masalah klasifikasi. Tiga metrik utama yang diperoleh dari Confusion Matrix adalah precision, recall, dan F1-score.

### 3.1.6 Deploy

Pada proses *deployment* ini model akan diletakkan pada lingkungan jupyter notebook yang kemudian model akan dibaca dan memberikan output hail deteksi melalui opencv. Untuk meningkatkan akurasi dari output yang diberikan oleh model Langkah deployment ini akan dilakukan dengan lebih selektif yaitu dengan cara memastikan bahawa output yang akan ditampilkan pada *layer* sudah mendapatkan deteksi benar terhadap satu kelas sebanyak satu kali dan maximal ada lima hasil prediksi yang dimunculkan dilayar yang kemudian apabila sudah mencapai batas tersebut maka hasil prediksi sebelumya akan dihapus. Hal ini dilakukan untuk menghindari bias dari *output* yang diberikan oleh model karena model melakukan prediksi setiap mili detik sehingga akan banyak bias apabila hasil deteksi dari model ditampilkan secara langsung tanpa menerapkan filter seperti yang dijelaskan diatas.

## 3.2 Analisis Kebutuhan

Pada tahap ini peneliti akan memaparkan tools apa saja yang digunakan untuk merealisasikan penelitian ini, daftar tools yang digunakan pada penelitian ini dijabarkan pada tabel 3.2

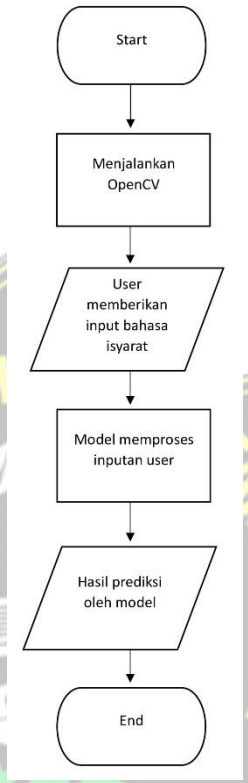
Tabel 3. 2 Analisis kebutuhan

| No | Tools | Version |
|---|---|---|
| 1 | Laptop Lenovo IdeaPad 3 14ALC6 | Windows 11 |
| 2 | Python | 3.8.10 & 3.9.11 |
| 3 | Mediapipe | 0.10.14 |
| 4 | Scikit-learn | 1.4.2 |
| 5 | matplotlib | 3.8.4 |
| 6 | OpenCV | 4.10.0 |
| 7 | Tensorflow | 2.13.0 |

## 3.3 Analisis Sistem

Pada tahap ini akan dijabarkan bagaimana perencanaan penggunaan sistem ketika model sudah mencapai tahap *deployment*. Tahap pertama adalah membuka jendela opencv dengan cara menjalankan blok kode yang ada pada

jupyter notebook kemudia user memberikan input berupa bahaa isyarat lalu model akan menterjemahkanya secara otomatis dan memberikan output pada layer sesuai dengan prediksi yang diberikan oleh model berdasarkan pelatihan yang sudah dilakukan.




Gambar 3. 13 *Flowchart* sistem

Pada 3.13 merupakan *flowchart* dari sistem yang akan dibangun, yang mana dari *flowchart* tersebut dapat dilihat bahwa alur penggunaan sistem cukup sederhana dimulai dari user menjalankan kode blok pada jupyter notebook untuk membuka popup opencv lalu user memberikan *input* kepada model melalui opencv berupa pose atau Gerakan bahasa isyarat yang kemudian secara otomatis model akan memberikan hasil prediksi dari pose atau Gerakan bahasa isyarat yang diberikan oleh user.

# BAB IV
# HASIL DAN ANALISIS PENELITIAN

## 4.1 Hasil dan Analisis

Pada tahap ini akan menampilkan hasil dan implementasi dari metode penelitian yang tercantum pada BAB 3 sebagai berikut :

### 4.1.1 Pelatihan dengan 3 kelas dataset

Pada skenario pertama ini peneliti mencoba melakukan pelatihan dan pengetesan model dengan hanya menggunakan 3 kelas dataset dari keseluruhan 40 dataset yang ada. 3 kelas tersebuat akalah huruf A, B, dan C, hal ini dilakukan untuk melihat performa model jika dihadapkan dengan skenario yang berbeda dan apakah model tetap relevan ketika diterapkan untuk melakukan deteksi nantinya.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| lstm (LSTM) | (None, 30, 64) | 82688 |
| lstm_1 (LSTM) | (None, 32) | 12416 |
| dense (Dense) | (None, 16) | 528 |
| dense_1 (Dense) | (None, 3) | 51 |

Total params: 95683 (373.76 KB)
Trainable params: 95683 (373.76 KB)
Non-trainable params: 0 (0.00 Byte)

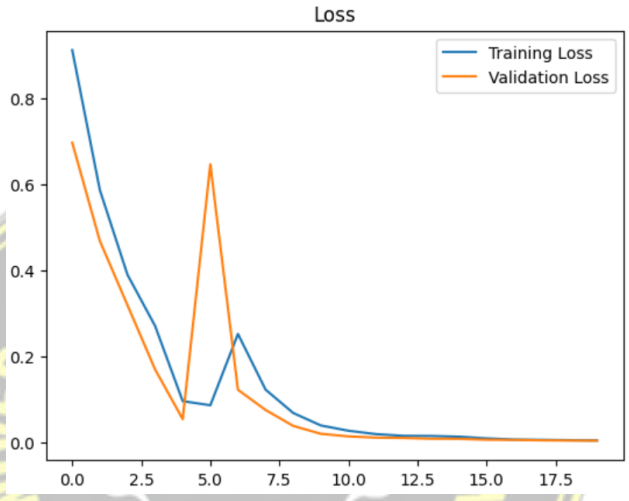
Gambar 4. 1 Rangkuman Model

Pada gambar 4.1 merupakan rangkuman model yang digunakan untuk pelatihan 3 kelas dataset, berdasarkan pada rangkuman model yang ada dapat dilihat bahwa susunan model terdiri dari 4 layer utama yaitu 2 layer LSTM dan 2 layer dense, layer LSTM pertama berfungsi untuk mempertahankan output sekuensial dari data, lalu diteruskan pada layer LSTM kedua yang berfungsi sebagai layer LSTM terakhir yang mengambil input berdasarkan output yang diberikan oleh lapisan pertama. Setelah melalui 2 layer LSTM data kemudian akan diterima oleh 2 layer dense yang ada didepan, fungsi utama dari kedua layer ini adalah untuk menentukan hasil akhir dari prediksi yang nantinya akan diberikan oleh model.

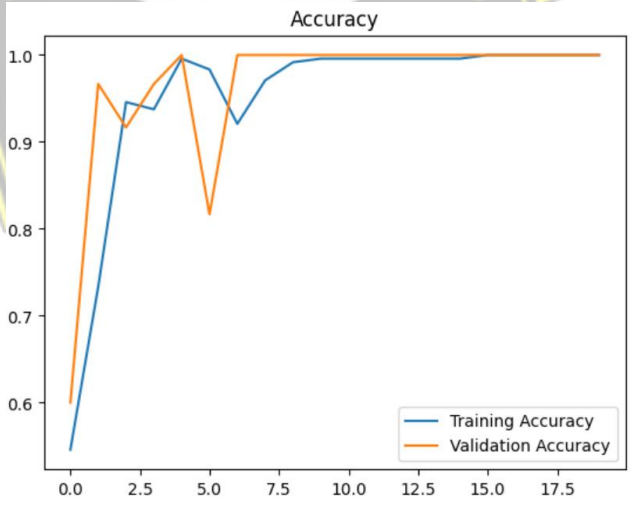
```
# Melatih model
history = model.fit(X_train, y_train, epochs=20, validation_data=(X_test, y_test))
```

Gambar 4. 2 Epoch Pelatihan

Pada gambar 4.2 menampilkan jumlah epoch yang digunakan untuk melatih model, model dengan 3 kelas dataset ini dilatih dengan epoch sebanyak 20x dan hasil dari pelatihan model akan disimpan pada variabel history yang nantinya variabel history ini akan digunakan untuk melakukan visualisasi ketika model menjalani pelatihan.



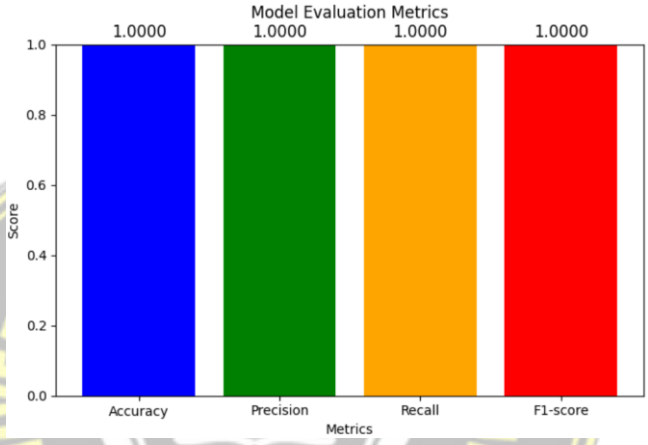

Gambar 4. 3 Grafik Loss




Gambar 4. 4 Grafik Akurasi

Pada gambar 4.3 menampilkan grafik loss ketika model menjalani pelatihan, dapat dilihat berdasarkan grafik tersebut angka loss baik pada pelatihan maupun validasi menurun secara drastis, meskipun terdapat sedikit

*spike* ditengah epoch akan tetapi model berhasil menyesuaikan Kembali sehingga mendapatkan hasil akhir yang maksimal, sedangkan pada gambar 4.4 merupakan grafik akurasi baik untuk pelatihan maupun validasi, dapat dilihat berdasarkan grafik tersebut Tingkat akurasi model naik secara perlahan dan meskipun sempat sedikit menurun ditengah pelatihan akan tetapi Tingkat akurasi model berhasil mencapai angka yang cukup tinggi yang mana itu menandakan bahwa model percaya diri dengan hasil prediksinya bedasarkan data yang diberikan.




Gambar 4. 5 Grafik *convolution metrics*

Pada gambar 4.5 menampilkan visualisasi dari *evaluation metrics* yang diterapkan untuk menguji seberapa akurat model yang telah dibuat, setelah model dilatih model perlu diverifikasi ulang apakah model benar-benar dapat memberikan output prediksi dengan benar atau tidak, selain itu hal ini berfungsi untuk melihat apakah model mengalami *overvitting* ataupun *undervitting* karena apabila model mengalami indikasi diatas maka model bisa dikategorikan tidak akurat dan tidak sesuai harapan yang mana hal ini bisa diperbaiki dengan melakukan *hyperparameter tuning* ataupun mengubah komposisi data pelatihan.

### 4.1.2 Pelatihan dengan 10 kelas dataset

Pada skenario kedua ini peneliti mencoba untuk melatih model menggunakan 10 kelas dataset, pada pelatihan kedua ini peneliti mencoba menggabungkan isi dari 10 kelas dataset yang akan dilatih yaitu mengkombinasikan antara alfabet dan kata sederhana, 10 kelas dataset

tersebut antara lain A, B, C, D, E, F, G, HALO, NAMA, SAYA. Pengkombinasian dataset ini dilakukan untuk melihat apakah model tetap responsif dengan kelas data yang makin banyak mengandung variasi.

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 30, 64)            82688

lstm_1 (LSTM)                (None, 32)                12416

dense (Dense)                (None, 16)                528

dense_1 (Dense)              (None, 10)                170

=================================================================
Total params: 95802 (374.23 KB)
Trainable params: 95802 (374.23 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

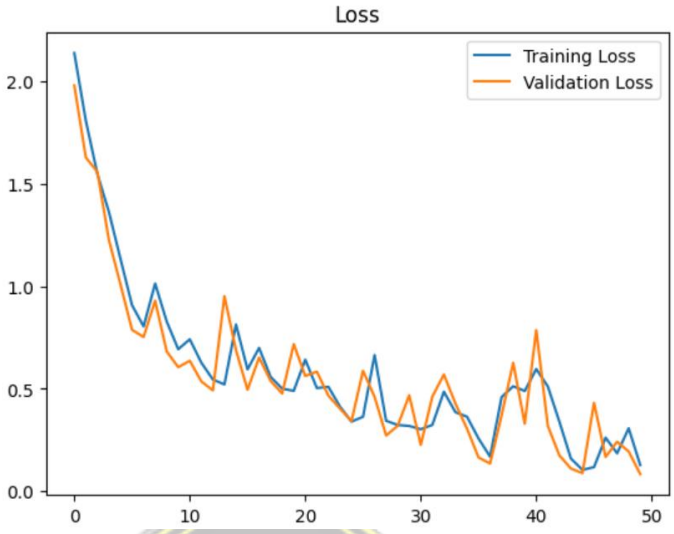
Gambar 4. 6 Rangkuman model

Pada gambar 4.6 merupakan ringkasan dari model yang akan digunakan untuk melatih 10 kelas dataset pada skenario pelatihan kedua ini. Secara garis besar model yang digunakan masih sama seperti pada skenario pertama, akan tetapi yang membedakan model ini dengan model sebelumnya adalah pada layer dense terakhir yang mana layer tersebut menunjukkan output akhir dari model ini yaitu 10 sesuai dengan skenario kedua ini yang juga melatih model dengan 10 kelas dataset yang berbeda.

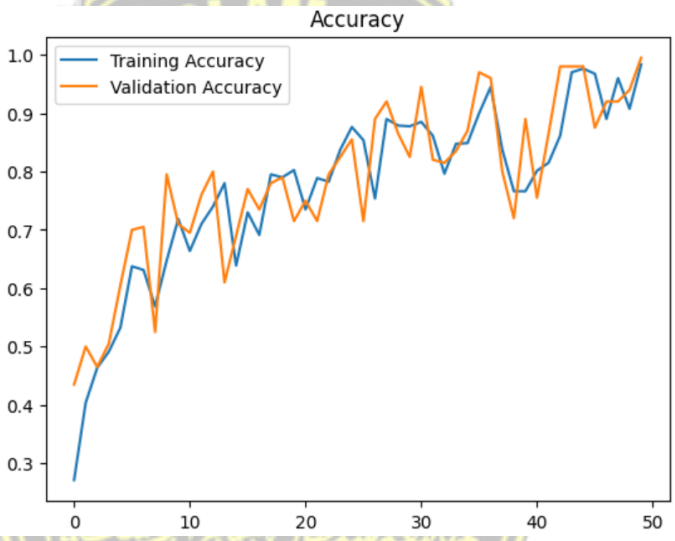
```
# Melatih model
history = model.fit(X_train, y_train, epochs=50, validation_data=(X_test, y_test))
```

Gambar 4. 7 Epoch pelatihan

Pada gambar 4.7 menampilkan jumlah epoch yang diterapkan dalam pelatihan model di skenario kedua ini. Dapat dilihat bahwa untuk pelatihan ini menerapkan 50 epoch yang mana hal ini berbeda dengan skenario pertama yang hanya menerapkan 20 epoch untuk pelatihan. Dengan menggunakan epoch yang lebih banyak hal ini meningkatkan kesempatan model untuk mempelajari dataset, yang mana pada skenario normal dengan menambah epoch ini dapat meningkatkan akurasi dan menurunkan loss yang dimiliki oleh model.

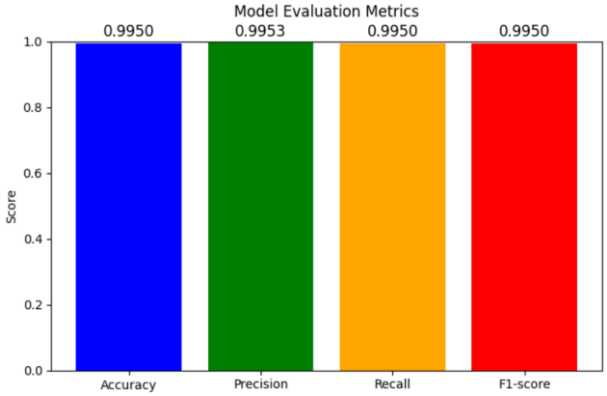



Gambar 4. 8 Grafik Loss




Gambar 4. 9 Grafik akurasi

Pada gambar 4.8 menampilkan grafik loss dari pelatihan skenario kedua ini, dapat dilihat berdasarkan grafik tersebut angka loss pada 5 epoch awal dapat dibilang cukup tinggi, akan tetapi setelah memasuki epoch 10 sampai dengan 50 loss yang ada mulai berkurang secara bertahap. Sedangkan pada gambar 4.9 menunjukkan grafik akurasi model selama menjalani pelatihan, pada grafik akurasi tersebut dapat diambil Kesimpulan bahwa model belajar dengan cukup baik selama 50 epoch karena dari awal epoch baik akurasi pelatihan ataupun validasi naik secara perlahan dan mencapai hasil akhir pada epoch 50 dengan hasil yang baik.




Gambar 4. 10 *Evaluation metrics*

Pada gambar 4.10 menampilkan hasil dari metrik evaluasi yang digunakan untuk melakukan pengujian pada model yang sudah dilatih. Dapat dilihat berdasarkan grafik batang dari metrik evaluasi tersebut hasil pengujian menunjukkan bahwa model sangat percaya diri dalam menentukan kelas prediksinya berdasarkan pelatihan yang sudah dilakukan.

### 4.1.3 Pelatihan dengan 20 kelas dataset

Pada skenario ketiga ini peneliti mencoba untuk melatih model dengan menggunakan 20 kelas dataset, pada tahap ini peneliti melakukan sedikit penyesuaian pada *hyperparameter* yaitu mengubah *learning rate default* dari optimizer adam yang semula 0.001 menjadi lebih kecil di angka 0.0005, hal ini dilakukan untuk memperlambat proses model dalam belajar yang mana hal ini dapat berdampak positif untuk model selama melakukan pelatihan. 20 kelas dataset yang digunakan untuk melatih model ini antara lain adalah A, B, C, D, E, F, G, H, I, J, HALO, NAMA, SAYA, KENAPA, TERIMAKASIH, TOLONG, KAPAN, PERKENALKAN, BERAPA, MAAF.

```
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 30, 64)            82688

 lstm_1 (LSTM)               (None, 32)                12416

 dense (Dense)               (None, 16)                528

 dense_1 (Dense)             (None, 20)                340

=================================================================
Total params: 95972 (374.89 KB)
Trainable params: 95972 (374.89 KB)
Non-trainable params: 0 (0.00 Byte)
```

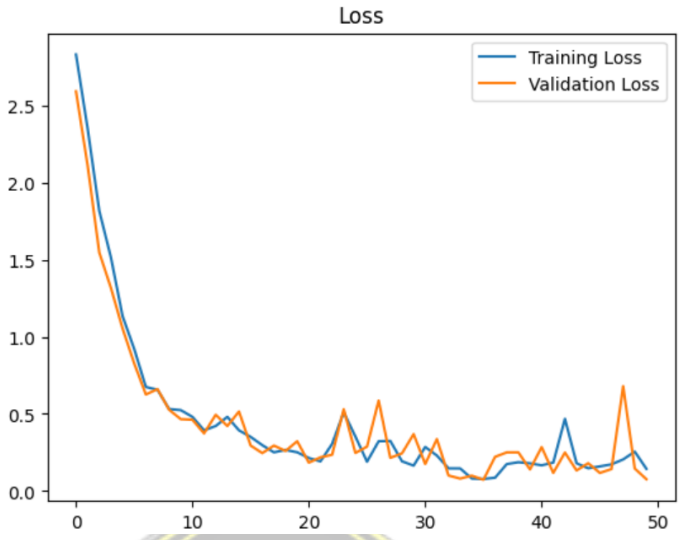
Gambar 4. 11 Ringkasan model

Pada gambar 4.11 menampilkan ringkasan model yang akan digunakan pada pelatihan skenario ketiga ini, untuk model yang akan digunakan masih sama seperti dua skenario sebelumnya yaitu menggunakan 2 layer LSTM dan 2 layer dense, yang membedakan model skenario ketiga ini adalah hasil output dari layer dense terakhir yang mana hasil output ini disesuaikan dengan jumlah kelas pelatihan pada skenario ketiga ini yaitu 20 kelas.

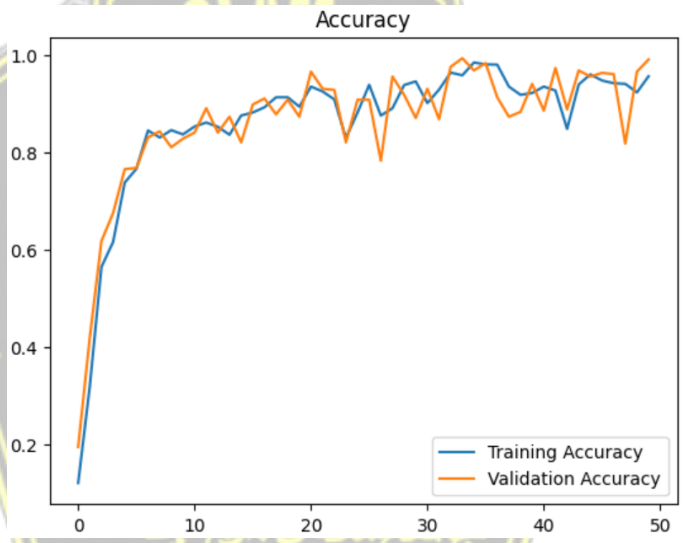
```
# Melatih model
history = model.fit(X_train, y_train, epochs=50, validation_data=(X_test, y_test))
```

Gambar 4. 12 Epoch pelatihan

Pada gambar 4.12 menampilkan jumlah epoch yang digunakan pada pelatihan model skenario ketiga, model pada skenario ini menjalani 50 epoch sama seperti pada skenario kedua, hal ini dilakukan karena meskipun untuk epoch tidak ada perubahan akan tetapi pada *learning rate* terdapat perubahan yang tadinya 0.001 menjadi 0.0005 perubahan *learning rate* ini yang menyebabkan emskipun model dilatih di epoch yang sama dengan jumlah dataset yang berbeda dua kali lipat akan tetapi model tetap menunjukkan hasil pelatihan yang maksimal.



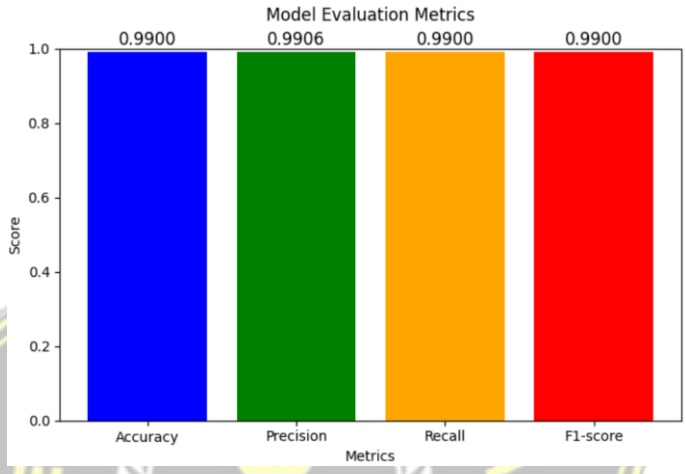

Gambar 4. 13 Grafik loss




Gambar 4. 14 Grafik akurasi

Pada gambar 4.13 menunjukkan grafik loss selama model menjalani pelatihan dapat dilihat berdasarkan grafik loss tersebut loss perlahan menurun seiring dengan berjalanya epoch, meskipun skenario ketiga ini dilatih menggunakan dataset yang lebih banyak akan tetapi tetap menggunakan epoch yang sama yaitu sejumlah 50 epoch, grafik loss tetap berkurang selama pelatihan dikarenakan untuk learning rate yang digunakan pada skenario ketiga ini adalah 0.0005 yang lebih kecil dibandingkan dengan dua skenario sebelumnya.

Sedangkan untuk gambar 4.14 menampilkan grafik akurasi selama momdel menjalani pelatihan, berdasarkan grafik tersebut Tingkat akurasi model perlahan meningkat selama pelatihan, dapat dilihat juga pada epoch awal Tingkat akurasi model meningkat dengan cukup pesat dan pada epoch akhir model memberikan hasil yang memuasskan dengan Tingkat akurasi diatas 90 persen.




Gambar 4. 15 *Evaluation metrics*

Pada gambar 4.15 hasil dari metrik evaluasi dalam bentuk grafik batang, dapat dilihat berdasarkan grafik tersebut untuk hasil metrik evaluasi ini cukup bagus yang menandakan bahwa hasil pelatihan model berjalan dengan baik dan memberikan hasil akhir yang memuaskan. Dengan melihat pada grafik tersebut dapat ditarik Kesimpulan bahwa model cukup percaya diri dalam memberikan hasil prediksi berdasarkan dataset selama pelatihan.

### 4.1.4 Pelatihan dengan 30 kelas dataset

Pada skenario keempat ini peneliti mencoba melatih model menggunakan 30 kelas dataset antara lain : A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, HALO, PERKENALKAN, NAMA, SAYA KAPAN, KENAPA, TERIMAKASIH, TOLONG, BERAPA, MAAF. Pada skenario keempat ini jumlah kelas dataset yang digunakan sudah mendekati skenario akhir yaitu 40 kelas dataset. Struktut model yang digunakan masih sama seperti tig akelas sebelumnya yaitu dengan menggunakan 2 *layer* LSTM dan

2 *layer* dense dengan output akhir pada *layer* dense adalah 30 sesuai dengan kelas dataset yang digunakan untuk pelatihan.

```
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 30, 64)            82688

 lstm_1 (LSTM)               (None, 32)                12416

 dense (Dense)               (None, 16)                528

 dense_1 (Dense)             (None, 30)                510

=================================================================
Total params: 96142 (375.55 KB)
Trainable params: 96142 (375.55 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

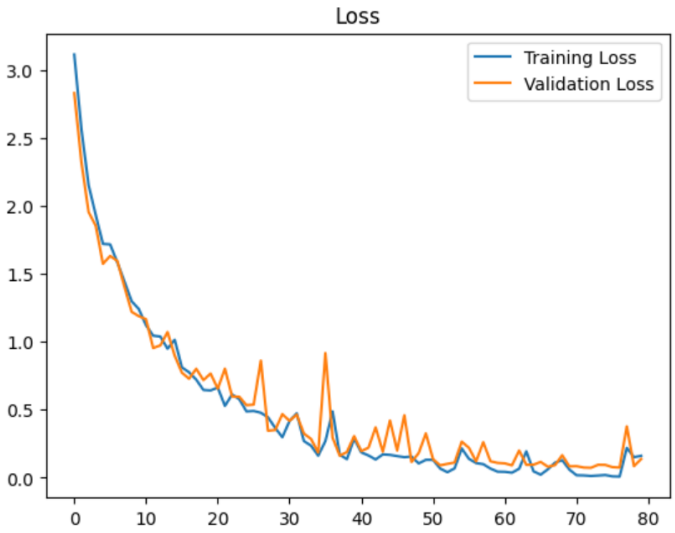
Gambar 4. 16 Ringkasan model

Pada gambar 4.16 menampilkan ringkasan model yang digunakan dalam skenario keempat ini, dapat dilihat untuk perbedaanya dengan skenario ketiga terdapat pada *output layer* dense terakhir yang mana untuk output ini sesuai dengan jumlah kelas yang digunakan untuk pelatihan yaitu 30 kelas dataset.

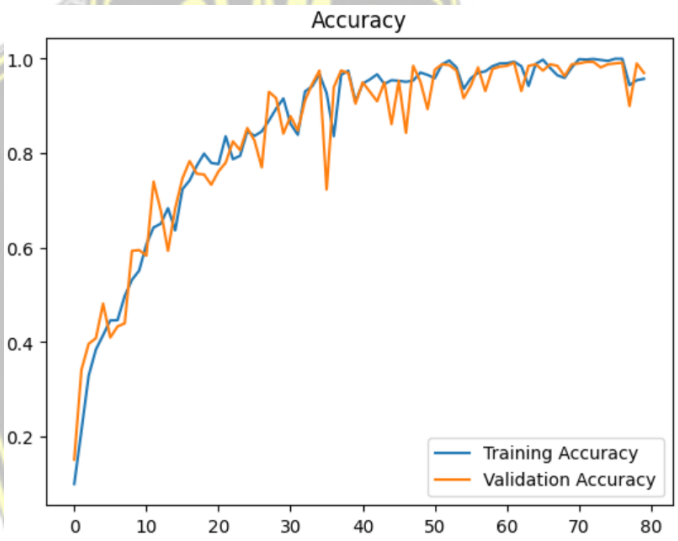
```
# Melatih model
history = model.fit(X_train, y_train, epochs=80, validation_data=(X_test, y_test))
```

Gambar 4. 17 Epoch pelatihan

Pada gambar 4.17 menampilkan jumlah epoch yang digunakan untuk melatih model pada skenario keempat ini, dapat dilihat bahwa pada skenario keempat model dilatih sebanyak 80 epoch yang mana penambahan epoch ini cukup signifikan jika dibandingkan dengan epoch sebelumnya, untuk learning rate yang digunakan masih sama diangka 0.0005.



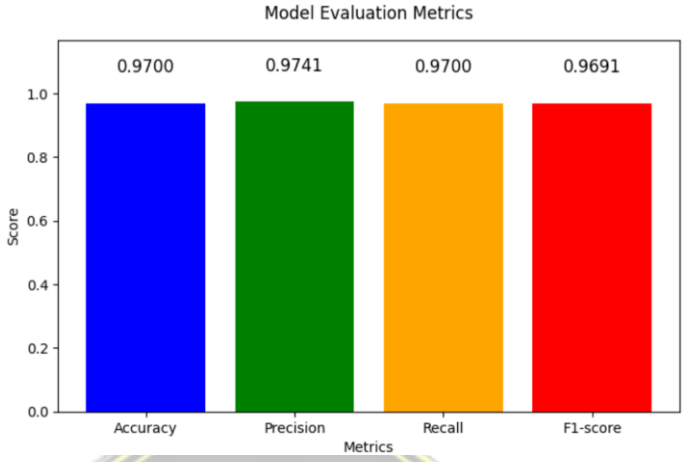

Gambar 4. 18 Grafik loss




Gambar 4. 19 Grafik akurasi

Pada gambar 4.18 menampilkan grafik loss dari pelatihan model pada skenario keempat, dapat dilihat bahwa Tingkat loss model turun secara perlahan mulai dari epoch pertama hingga epoch terakhir, hal ini menandakan bahwa model berhasil untuk mempelajari pola dataset dengan baik, hal ini juga diperkuat pada gambar 4.19 yang mana menunjukkan Tingkat akurasi model ketika pelatihan, dapat dilihat bahwa pada grafik ini akurasi model terus meningkat, meskipun ada spike penurunan pada grafik akurasi akan tetapi model langsung Kembali pada grafik ideal yang mana terus naik yang

menandakan bahwa model belajar dengan baik pada skenario 30 kelas dataset.




Gambar 4. 20 *Evaluation metrics*

Pada gambar 4.20 menampilkan *evaluation metrics* dari pelatihan model pada skenario keempat, dapat dilhat pada grafik batang tersebut meskipun hasil evaluasi metrik tidak setinggi ketiga skenario lain akan tetapi hasil ini tetap bisa dibilang tinggi dengan indikassi hasil evaluasi masih diatas 90 persen yang menandakan model masih cukup percaya diri dalam memberikan prediksi berdasarkan pelatihan yang sudah dilalui.

### 4.1.5 Pelatihan dengan 40 kelas dataset

Pada skenario 40 kelas dataset sekaligus skenario terakhir ini peneliti akan melatih model menggunakan keseluruhan data yang sudah dikumpulkan pada tahap pengumpulan dataset, 40 kelas dataset tersebut antara lain: 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, BERAPA, HALO, KAPAN, KENAPA, MAAF, MAU, NAMA, PERKENALKAN, SAMA-SAMA, SAYA, TERIMAKASIH, TIDAK MAU, TOLONG, YA.

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 30, 64)            82688

lstm_1 (LSTM)                (None, 32)                12416

dense (Dense)                (None, 16)                528

dense_1 (Dense)              (None, 40)                680

=================================================================
Total params: 96312 (376.22 KB)
Trainable params: 96312 (376.22 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

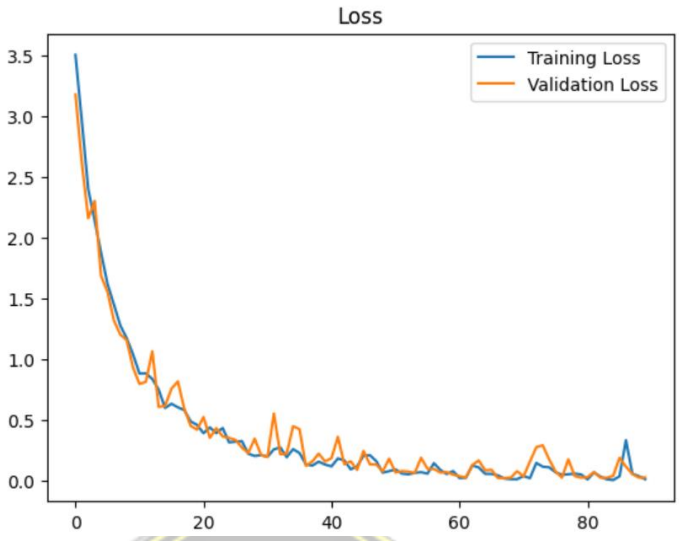
Gambar 4. 21 Ringkasan model

Pada gambar 4.21 menampilkan ringkasan dari model yang akan digunakan pada pelatihan skenario kelima ini, dapat dilihat bahwa model akan dilatih dengan empat *layer* yaitu dua *layer* LSTM dan dua *layer* dense sebagai *layer* yang akan memberikan output akhir model sesuai dengan jumlah kelass yang digunakaan saat pelatihan yaitu 40 kelas.

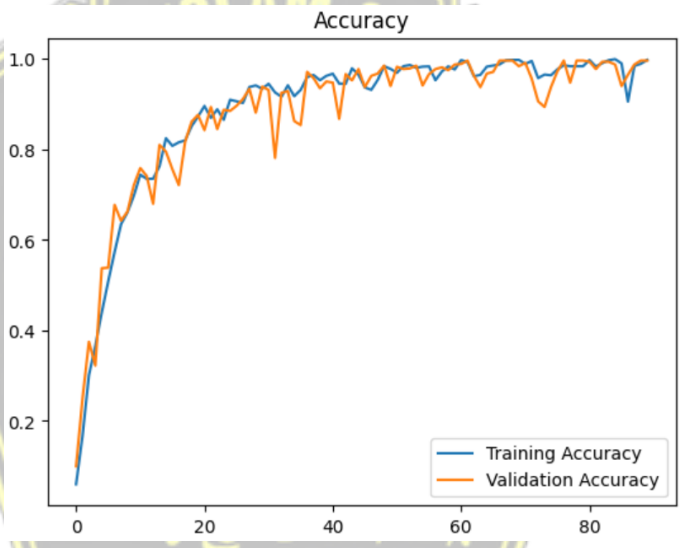
```
# Melatih model
history = model.fit(X_train, y_train, epochs=90, validation_data=(X_test, y_test))
```

Gambar 4. 22 Epoch pelatihan

Gambar 4.22 menampilkan jumlah epoch yang akan digunakan dalam pelatihan skenario kelima ini, dapat dilihat bahwa model akan dilatih sebanyak 90 epoch, jumalh epoch ini adalah epoch terbanyak jika dibandingkan dengan keempat skenario pelatihan sebelumnya, hal ini dikarenakan pada skenario kelima ini juga merupakan skenario dengan kelas data terbanayk yaitu sebanyak 40 kelas dataset.

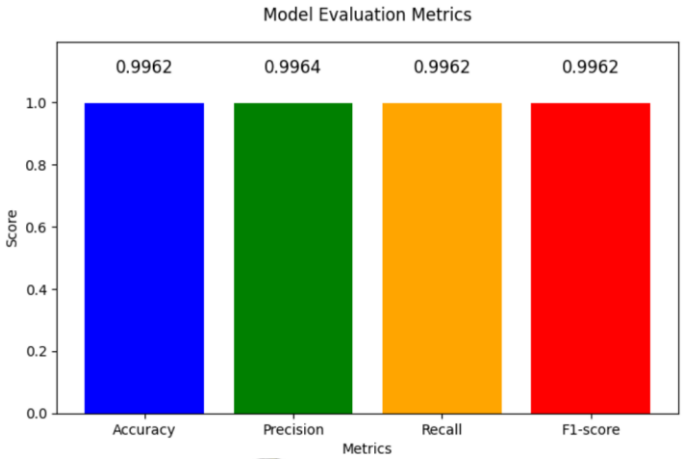



Gambar 4. 23 Grafik loss




Gambar 4. 24 Grafik akurasi

Pada gambar 4.23 menampilkan grafik akurasi dari hasil pelatihan model pada skenario kelima ini, dapat dilihat berdasarkan grafik loss tersebut dapat ditarik Kesimpulan bahwa model belajar dengan cukup baik hal ini diindikasikan dengan makin menurunya angka loss hingga menentuh angka dibawah satu selain juga dapat dibuktikan pada gambar 4.24 yang mana tingakat akurasi dari model perlahan naik hingga menyentuh angka akurasi 99% yang mana hal ini menandakan bahwa model cukup percaya diri dalam memberikan hasil prediksi dari 40 kelas data pelatihan yang ada.




Gambar 4. 25 *Evaluation metrics*

Pada gambar 4.25 menampilkan hasil dari *evaluation metrics* yang dilakukan untuk menguji apakah model benar-benar dapat memberikan hassil prediksi yang bagus atau tidak, dari hasil ini dapat ditarik Kesimpulan bahwa model sangat percaya diri terhadap output klasifikasi yang diberikan. Hasil dari metrik evalusi ini terbilang lebih tinggi jika dibandingkan dengan skenario sebelumnya, pada skenario kelima ini dengan pelatihan model menggunakan 40 kelas dataset model masih dapat memberikan hasil evaluation metrics yang tinggi.

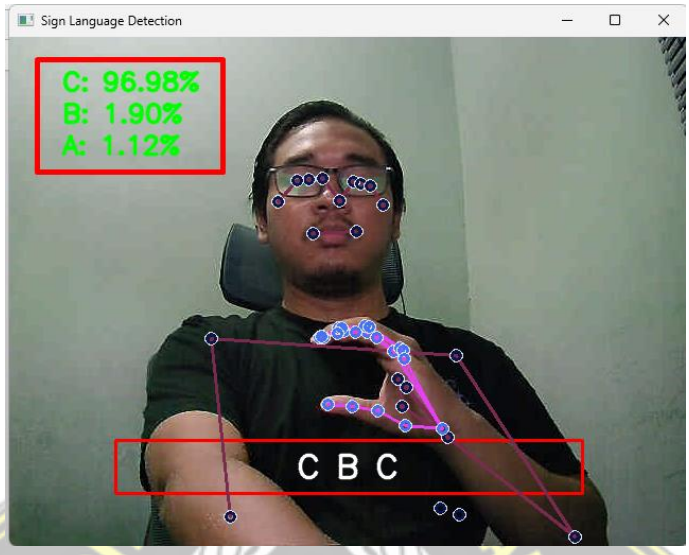
## *4.2 Deployment*

Pada tahap ini penulis akan mendeploy kelima model yang sudah dilatih dengan menggunakan OpenCV. Setiap model yang sudah dilatih akan dideploy satu persatu untun melihat performa model apabila sudah diterapkan langsung dengan skenario livestream dan juga untuk melihat apakah nilai akurasi yang diberikan kepada model sejalan dengan hasil prediksi livestream yang diberikan.

### 4.2.1 *Deployment* model 3 kelas dataset

Pada tahap ini peneliti akan mencoba untuk melakukan *deployment* terhadap model yang sudah dihasilkan dari skenario pelatihan pertama, pada skenario pertama model dilatih dengan menggunakan tiga kelas dataset. Bedasarkan pada grafik akurasi dan loss model menunjukkan bahwa model berlatih dengan sangat baik, selain itu mengacu pada hasil *evaluation metrics*

model juga berhasil memberikan skor evaluasi yang tinggi yang mana dengan skor evaluasi ini menandakan bahwa model tidak hanya baik ketika pelatihan akan tetapi juga baik dalam memberikan hasil prediksi multi kelas.




Gambar 4. 26 *Deployment* 3 kelas dataset

Pada gambar 4.26 menampilkan tampilan model setelah berhasil melakukan *deployment* dengan menggunakan opencv, dapat dilihat pada gambar 4.26 peneliti amsih mempertahankan *landmark* mediapipe, *landmark* ini hanya berfungsi sebagai tampilan saja yang dapat memudahkan peneliti untuk emmastikan apakah posisi tangan dan landmark mediapipe sudah sesuai dengan arah yang diharapkan, *landmark* yang ditampilkan tidak akan mempengaruhi prediksi karena hanya berfungsi sebagai tampilan.

Pada *deployment* ini peneliti menampilkan dua output dengan memberikan kotak merah agar lebih mudah untuk identifikasi, kotak pertama adalah kotak merah pada bagian atas yang mana kotak ini berfungsi untuk menunjukkan tiga prediksi teratas yang dihasilkan oleh model dari skor argmax, dan untuk kotak merah kedua yang ada pada bagian bawah merupakan kotak hasil prediksi akhir model yang mana sebelum menampilkan hasil prediksi pada kotak ini model harus melakukan prediksi benar sebanyak sepuluh kali.

Dapat ditarik Kesimpulan bahwa pada skenario pertama ini model masih cukup baik dalam memberikan hasil prediksi yang mana dapat dilihat dari

persentase pada kotak merah diatas yang menampilkan huruf C dengan Tingkat percaya diri sebesar 96%.

Tabel 4. 1 Uji hasil deployment

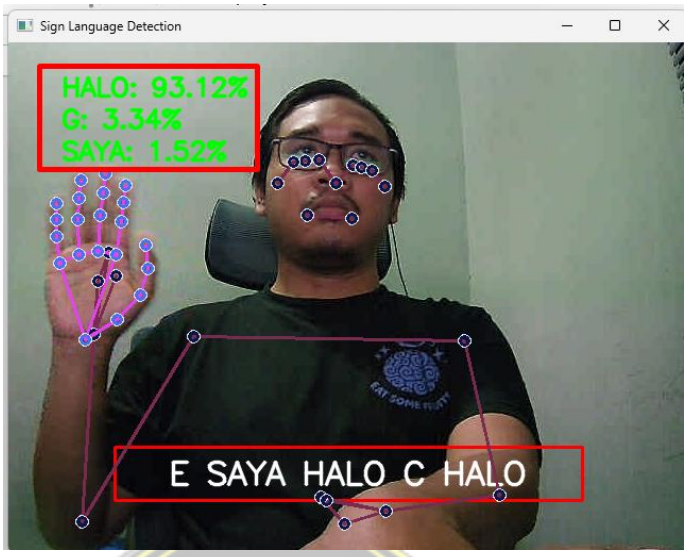
| Label | Uji 1 | Uji 2 | Uji 3 | Uji 4 | Persentase |
|---|---|---|---|---|---|
| A | B | B | B | B | 100% |
| B | B | B | B | B | 100% |
| C | B | B | B | B | 100% |

Pada tabel 4.1 menampilkan hasil dari uji coba deteksi pada model ayng sudah dideploy, untuk skenario pengujian peneliti mencoba untuk memberikan input sesuai dengan label kelas yang ada dan mencoba memberikan input sebanyak tiga kali lalu melhat apakah model berhasil memberikan prediksi sesuai dengan input yang diberikan.

Pada model skenario pertama ini model berhasil memberikan prediksi terhadap semua kelas yang ada yaitu A, B, dan C yang mana dengan pembuktian ini menegaskan bahwa model pada skenario pertama ini memiliki Tingkat keakuratan 100% dalam memberikan prediksi sesuai dengan inputan yang diberikan.

### 4.2.2 *Deployment* model 10 kelas dataset

Pada skenario kedua ini peneliti mencoba melakukan *deployment* dari model yang sudah dilatih dengan 10 kelas dataset, teknis *deployment* masih sama seperti tahap sebelumnya yang mana sebelum model memberikan hasil prediksi, model hasrus berhasil melakukan prediksi dibalik layar sebanyak 10 kali dan kemudian hasil akhir dari prediksi model akan ditampilkan pada kotak merah dibagian bawah.




Gambar 4. 27 *Deployment* 10 kelas dataset

Pada gambar 4.27 menampilkan hasil *deployment* dari mode yang sudah dilatih pada skenario kedua dengan menggunakan 10 kelas dataset, dapat dilihat bahwa pada skenario kedua ini model masih cukup percaya diri dalam memberikan hasil prediksi setelah melakukan *deployment*, hasil prediksi yang diberikan oleh model tidak setinggi skenario pertama dalam hal presentase yang mana apabila pada skenario pertama presentase akhir dari prediksi mencapai 96% maka pada skenario kedua ini hasil persentase prediksi akhir yang tertangkap layer ada di angka 93%.

Hal ini menunjukkan bahwa semakin banyak kelas maka model akan semakin sulit dalam memberikan prediksi dengan persentase yang tinggi, meskipun hal ini bisa dibilang cukup bagus yang mana model pada skenario kedua ini memiliki kelas dataset yang 3 kali lebih banyak dibandingkan dengan skenario kedua akan tetapi model tetap berhasil memberikan prediksi dengan persentase yang tinggi.

Pada gambar 4.27 terdapat dua kotak merah pada bagian atas dan bawah, untuk kotak merah pada bagian atas menampilkan tiga prediski dengan Tingkat persentase tertinggi lalu pada kotak kedua yang ada dibagian bawah menampilkan hasil akhir prediksi. Sebelum model menampilkan prediksi pada kotak merah dibawah model terlebi dahulu harus memenuhi syarat untuk mengurangi bias dalam prediksi, yaitu model harus berhasil memprediksi

benar sebanyak 10 kali sebelum memberikan output prediksi akhir pada kotak merah dibagian bawah.

Tabel 4. 2 Uji hasil *deployment*

| Label | Uji 1 | Uji 2 | Uji 3 | Uji 4 | Persentase |
|---|---|---|---|---|---|
| A | B | B | B | B | 100% |
| B | S | S | B | S | 25% |
| C | B | B | B | B | 100% |
| D | S | B | S | S | 25% |
| E | B | B | B | B | 100% |
| F | S | S | S | S | 0% |
| G | B | B | B | B | 100% |
| HALO | B | B | B | B | 100% |
| NAMA | B | B | B | B | 100% |
| SAYA | S | S | B | S | 25% |

Pada tabel 4.2 menampilkan hasil uji coba model pada skenario kedua ini, dapat dilihat bahwa untuk beberapa kelas model sangat percaya diri dalam menentukan prediksi namun untuk beberapa kelas lainya model mengalami kesulitan dalam prediksi dan bahkan untuk kelas alfabet F model gagal memprediksi pada keempat kesempatan yang diberikan.

Berdasarakan pada tabel 4.2 dapat ditarik Kesimpulan bahwa pada skenario *deployment* kedua ini model memiliki nilai keakuratan dalam memprediksi sebesar 77,5% yang mana hasil ini mengalami penurunan jika dibandingkan dengan saat model menjalani pelatihan dan evaluasi menggunakan *evaluation metrics*.

### 4.2.3 *Deployment* model 20 kelas dataset

Pada skenario ketiga ini peneliti akan mencoba mendeploy model yang sudah dilatih dengan menggunakan 20 kelas dataset. Jumlah kelas pada skenario ini dua kali lebih banyak jika dibandingkan skenario sebelumnya, untuk teknis *deployment* masih sama seperti skenario sebelumnya yaitu model harus berhasil memerbikan 10 kali prediksi benar dahulu sebelum hasil prediksi dapat ditampilkan sebagai hasil prediksi akhir.


Gambar 4. 28 *Deployment* 20 kelas dataset

Pada gambar 4.28 menampilkan hasil *deployment* dari model yang sudah dilatihi pada skenario ketiga sebelumnya, model yang dideploy pada tahap ini sudah dilatih menggunakan 20 kelas dataset, dapat dilihat pada gambar meskipun persentase keakuratan yang ditampilkan tidak setinggi dua skenario sebelumnya akan tetapi model masih dapat memberikan hasil prediksi yang cukup akurat, akan tetapi pada skenario *deployment* ini peneliti menemukan sedikit kendala yaitu model mengalami kesulitan dalam membaca pose yang diam seperti pose alfabet dan model lebih mudah dan juga akurat ketika memprediksi pose yang bergerak seperti pose kata-kata seperti yang tertera pada gambar 4.28

Pada deployment skenario ketiga ini penulis menghilangkan kotak merah pada bagian atas karena untuk kotas merah menghalangi tampilan dan menjadi tidak efektif apabila tetap dipertahankan, akan tetapi output persentase tetap peneliti tampilkan sebgai panduan untuk melihat apa yang sedang diprediksi oleh model. Sedangkan untuk kotak merah pada bagian bawah amsih peneliti pertahankan untuk memberikan Batasan hasil prediksi agar terlihat lebih rapi dan mudah untuk dibaca.

Dapat ditarik Kesimpulan sementara bahwa pada *deployment* skenario ketiga ini model sudah mulai mengalami kesulitan dalam memberikan prediksi terutama pada pose yang diam, sedangkan model lebih sensitive dan

mudah apabila membaca pose yang bergerak, hal ini tidak selaras apabila dibandingkan dengan hasil model ketika peltihan yang mana model pada skenario ketiga ini berdasarkan metrik evaluasi tetap baik dalam memprediksi semua kelas yang ada.

Tabel 4. 3 Uji hasil *deployment*

| Label | Uji 1 | Uji 2 | Uji 3 | Uji 4 | Persentase |
|---|---|---|---|---|---|
| A | S | B | S | S | 25% |
| B | B | S | B | B | 75% |
| C | S | S | B | S | 25% |
| D | B | B | S | B | 75% |
| E | S | S | S | B | 25% |
| F | B | S | B | B | 75% |
| G | B | S | B | B | 75% |
| H | S | S | S | S | 0% |
| I | S | S | S | S | 0% |
| J | S | S | S | S | 0% |
| HALO | S | B | B | S | 50% |
| NAMA | B | B | B | B | 100% |
| SAYA | B | S | S | S | 25% |
| KENAPA | S | S | S | S | 0% |
| TERIMAKASIH | S | S | B | S | 25% |
| TOLONG | S | S | S | S | 0% |
| KAPAN | S | B | B | B | 75% |
| PERKENALKAN | B | B | B | B | 100% |
| BERAPA | S | S | S | S | 0% |
| MAAF | S | S | S | S | 0% |

Pada tabel 4.3 menampilkan hasil uji coba *deployment* untuk model skenario ketiga, dapat dilihat pada skenario ketiga ini model mengalami penurunan performa ketika melakukan deployment, pada tabel 4.3 terdapat persentase keberhasilan deteksi dari masing-masing label yang dilakukan sebanyak empat kali. Pada uji hasil ini ada beberapa label yang sangat baik

ketika diprediksi yaitu label NAMA dan PERKENALKAN yang memiliki Tingkat akurasi mencapai 100% akan tetapi sebaliknya ada beberapa label yang gagal diprediksi oleh model selama uji coba sebanyak empat kali antara lain label H, I, J KENAPA, BERAPA, dan MAAF. Hal ini mugkin disebabkan oleh makin banyaknya label yang harus diprediksi oleh model

Pada skeanrio ketika ini berdasarkan tabel 4.3 tingkat akurasi model dalam memberikan prediksi benar adalah sebanyak 42.5% yang mana angka ini terbilang sangat kecil apabila dibandingka dengan hasil evaluasi model ketika model melakukan pelatihan.

### 4.2.4 *Deployment* model 30 kelas dataset

Pada skenario keempat ini peneliti akan mendeploy model yang sudah dilatih menggunakan 30 kelas dataset, teknis *deployment* dapa skenario ini disamakan seperti skenario ketiga yang mana peneliti akan menghilangkan kotak merah pada bagian atas dan tetap mempertehankan kotak merah dibagina bawah sebagai pemabatas untuk model memberikan hasil prediksi.


Gambar 4. 29 *Deployment* 30 kelas dataset

Pada gambar 4.29 merupakan hasil *deployment* dari skenario keempat ini dengan model yang sudah dilatih dengan 40 kelas dataset, dapat dilihat berdasarkan gambar 4.29 model melakukan kesalahan dalam memberikan prediksi akhir yang mana untuk pose yang peneliti lakukan merupakan pose bergerak untuk kata terimakasih, akan tetapi model memberikan output

prediksi berupa huruf R yang mana untuk dua pose antara R dan terimakasih terbilang cukup jauh.

Akan tetapi dapat dilhat pada presentase yang ada dibagian atas kata terimakasih tetap muncul dengan presentase yang cukup kecil, yang mana hal ini menandakan meskipun pada prediksi akhir model memberikan hasil yang salah model tetap mengidentifikasi adanya kemungkinan untuk pose terimakasih, hal ini mungkin disebabkan oleh banyaknya kelas dataset yang digunakan sehingga model mengalami bias yang besar dalam menentukan hasil prediksi akhir.

Tabel 4. 4 Uji hasil *deployment*

| Label | Uji 1 | Uji 2 | Uji 3 | Uji 4 | Persentase |
|---|---|---|---|---|---|
| A | B | S | B | B | 75% |
| B | S | S | B | S | 25% |
| C | S | S | S | S | 0% |
| D | S | S | S | S | 0% |
| E | S | S | S | S | 0% |
| F | S | S | S | S | 0% |
| G | B | B | B | B | 100% |
| H | B | S | B | B | 75% |
| I | S | S | S | S | 0% |
| J | S | S | S | S | 0% |
| K | S | S | S | S | 0% |
| L | S | B | S | S | 25% |
| M | S | S | S | S | 0% |
| N | S | S | S | S | 0% |
| O | S | B | B | B | 75% |
| P | S | S | S | S | 0% |
| Q | B | B | B | B | 100% |
| R | S | B | S | S | 25% |
| S | S | S | S | B | 25% |

| | | | | | |
|---|---|---|---|---|---|
| T | S | B | B | B | 75% |
| HALO | S | S | S | S | 0% |
| PERKENALKAN | B | B | B | B | 100% |
| NAMA | B | B | B | B | 100% |
| SAYA | B | B | S | B | 75% |
| KAPAN | B | S | B | B | 75% |
| KENAPA | S | S | S | S | 0% |
| TERIMAKASIH | B | S | S | S | 25% |
| TOLONG | S | S | S | S | 0% |
| BERAPA | S | S | S | S | 0% |
| MAAF | S | S | S | S | 0% |

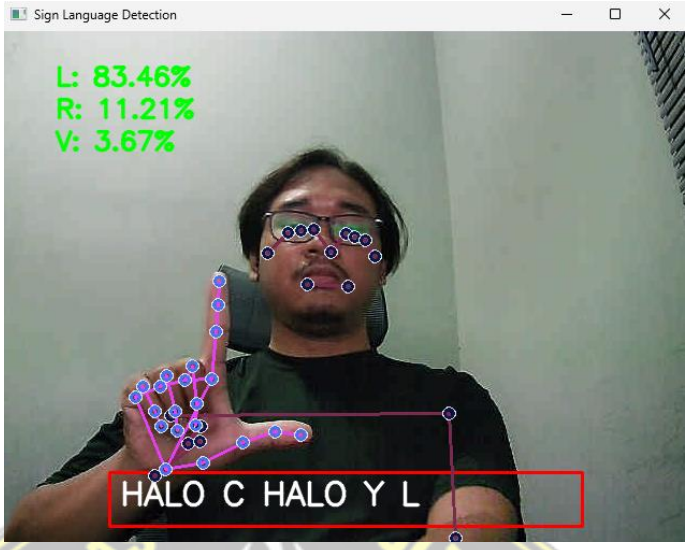
Pada tabel 4.4 menampilkan hasil uji coba prediksi pada skenario deployment model keempat, dapat dilihat bedasarkan tabel 4.4 model Kembali mengalami penurunan performa ketika tahap *deployment*, secara keseluruhan model hanya memiliki persentase keberhasilan sebesar 35.83% yang mana hasil ini sangat jauh berkurang apabila dibandingkan dengan ketika model menjalani pelatihan dan melakukan evaluasi/

Pada *deployment* skenario keempat ini ada beberapa label yang berhasil diprediksi model dengan sangat percaya diri antara lain G, Q PERKENALKAN, dan NAMA. Dari keempat label tersebut semua memberikan persentase tertinggi mencapai 100% yang mana ini berarti model tidak pernal salah ketika diberikan input sesuai dengan label tersebut. Sedangkan sebaliknya ada beberapa label yang gagal diprediksi oleh model bahkan pada semua kesempatan uji coba, tiga diantara label yang gagal diprediksi oleh model tersebut adalah C, D, dan E yang mana ketiga label tersebut termasuk kedalam pose alfabet yang tidak bergerak

### 4.2.5 *Deployment* model 40 kelas dataset

Pada skenario terakhir ini penulis akan mencoba untuk melakukan deployment terhadap model yang sudah dilatih dengan menggunakan 40 kelas dataset, untuk teknis *deployment* masih sama seperti skenario keempat

dan kelima yang mana penulis akan menghilangkan kotak merah pada bagian atas dan etap menampilkan kotak merah pada bagian bawah.




Gambar 4. 30 *Deployment* 40 kelas dataset

Pada gambar 4.30 menampilkan hasil *deployment* dari model yang dilatih menggunakan 40 kelas dataset, pada skenario ini hasil dari *deployment* menunjukkan perilaku yang sama dengan skenario sebelumnya yaitu skenario ketiga, pada skenario ini model mengalami kesulitan dalam memberikan prediksi yang akurat sesuai dengan input pose yang diberikan oleh *user*, akan tetapi apabila pada skenario ketiga model mengalami kesulitan dalam memprediksi pose yang diam amka pada skenario kelima ini model lebih mudah dalam memprediksi skenario diam dan mengalami kesulitan dalam skenario yang bergerak.

Dapat dilihat pada gambar 4.30 penulis memberikan input pose huruf L yang mana model berhasil memerbikan prediksi huruf L, akan tetapi ketika penulis mencoba memberikan *input* pose bergerat terimakasih, model melakukan kesalahan dengan memberikan hasil prediksi kata halo seperti tertera pada gambar 4.30

Dapat ditarik Kesimpulan sementara bahwa pada skenario kelima ini model mengalami kesulitan dalam memberikan prediksi karena kelas yang semakin banyak dan beragam yang itu ada 26 huruf alfabet yang didominasi

oleh pose diam dan 14 kata yang didominasi oleh pose yang bergerak yang menyebabkan model mengalami bias dalam memberikan hasil prediksi.

Tabel 4. 5 Uji hasil *deployment*

| Label | Uji 1 | Uji 2 | Uji 3 | Uji 4 | Persentase |
|---|---|---|---|---|---|
| A | B | S | B | S | 50% |
| B | S | S | S | S | 0% |
| C | B | B | S | B | 75% |
| D | S | S | S | S | 0% |
| E | S | S | S | S | 0% |
| F | S | S | S | S | 0% |
| G | S | B | B | B | 75% |
| H | S | B | S | S | 25% |
| I | S | S | S | S | 0% |
| J | S | S | S | S | 0% |
| K | S | B | S | B | 50% |
| L | B | B | S | B | 25% |
| M | S | S | S | S | 0% |
| N | S | S | S | S | 0% |
| O | S | S | S | S | 0% |
| P | S | B | S | S | 25% |
| Q | S | B | B | S | 50% |
| R | B | B | S | S | 50% |
| S | S | S | S | S | 0% |
| T | S | S | S | S | 0% |
| U | S | S | S | S | 0% |
| V | B | S | B | B | 75% |
| W | S | S | S | S | 0% |
| X | S | S | S | S | 0% |
| Y | S | S | S | S | 0% |
| Z | S | S | S | S | 0% |

| | | | | | |
|---|---|---|---|---|---|
| BERAPA | S | S | S | S | 0% |
| HALO | S | S | S | S | 0% |
| KAPAN | S | S | S | S | 0% |
| KENAPA | S | S | S | S | 0% |
| MAAF | S | S | S | S | 0% |
| MAU | S | S | S | S | 0% |
| NAMA | B | B | B | B | 100% |
| PERKENALKAN | B | B | B | B | 100% |
| SAMA-SAMA | S | S | S | S | 0% |
| SAYA | S | S | S | S | 0% |
| TERIKAMASIH | S | S | S | S | 0% |
| TIDAK MAU | S | S | S | S | 0% |
| TOLONG | S | S | S | S | 0% |
| YA | S | S | S | S | 0% |

Pada tabel 4.5 menampilkan hasil dari ujicoba *deployment* pada skenario keempat ini, dapat dilihat pada tabel 4.5 hasil persentase setiap kelas yang ada pada skenario keempat ini cenderung menunjukkan performa yang rendah pada saat *deployment*, banyak label yang gagak dibaca oleh model bahkan pada keempat uji coba yang dilakukan, akan tetapi ada dua label yang tetap konsisten memberikan hasil yang sangat akurat yaitu label PERKENALKAN dan NAMA.

Berdasarkan pada tabel 4.5 persentase keseluruhan yang didapat oleh model dari skenario keempat ini adalah 17,5% yang mana persentase ini sangat rendah jika dibandingkan ketika model menjalani training dan validasi yang mana skor *evaluation matrics* saat itu mencapai lebih dari 90%.

# BAB V
# KESIMPULAN DAN SARAN

## 5.1 Kesimpulan

Berdasarkan penelitian ini dapat ditarik Kesimpulan bahwa dengan menggunakan bantuan mediapipe dan mengambil landmarknya sebagai dataset lalu menyederhanakannya dengan bantuan flatten maka data tersebut dapat digunakan untuk melatih model *Long Short Term Memory* yang dapat digunakan untuk membaca Gerakan dan pose dalam bahasa isyaraat Indonesia (BISINDO)

Hasil evaluasi saat training dan persentase saat melakukan *deployment* terdapat perbedaan akurasi yang cukup tinggi, pada skenario pertama mendapat skor *training* 99% dan *deployment* 100%, pada skenario kedua mendapat skor *training* 99% dan *deployment* 77.5%, pada skenario ketiga mendapat skor *training* 99% dan *deployment* 42.5%, pada skenario keempat mendapat skor *training* 99% dan *deployment* 35/83%, dan yang terakhir pada skenario kelima model mendapatkan skor *training* 99% dan *deployment* 17.5%. Berdasarkan skor persentase diatas dapat disimpulkan bahwa skot validasi saat training tidak berbanding lurus dengan skor ketika model sudah memasuki tahap *deployment*.

## 5.2 Saran

Bedasarkan penelitian yang sudah ada, penulis memberi saran untuk penelitian yang akan datang adalah :

1. Meningkatkan jumlah dataset pelatihan agar model dapat menerima data pelatihan lebih banyak sehingga mengurangi bias prediksi ketika model masuk tahap *deployment*.
2. Menggunakan algoritma selain *Long Short Term Memory* atau dapat mengkombinasikan dengan algoritma *deep learning* lain agar mendapat hasil prediksi yang lebih maksimal.
3. Mengoptimalkan pada tahap *deployment* adan cara membaca model agar model dapat memberikan output yang lebih akurat dan meminimalisir bias yang ada dalam memberikan hasil prediksi akhir.

# DAFTAR PUSTAKA


Aisyah Muhammad Amin, N., Pribadi, F., & Kunci, K. (2022). Urgensi Bahasa Isyarat dalam Pendidikan Formal sebagai Media Komunikasi dan Transmisi Informasi Penyandang Disabilitas Rungu dan Wicara. *Jurnal Hasil Pemikiran, Penelitian, dan Pengembangan Keilmuan Sosiologi Pendidikan*, *9*(1), 77–86.

Hasyim Nur'azizan, A., Riqza Ardiansyah, A., & Fernandis, R. (t.t.). *Implementasi Deteksi Bahasa Isyarat Tangan Menggunakan OpenCV dan MediaPipe* (Vol. 3).

Hayyu Gustsa, A., & Setyo Permadi, G. (2022). *Sistem Deteksi Bahasa Isyarat Secara Realtime Dengan Tensorflow Object Detection dan Python Menggunakan Metode Convolutional Neural Network*.

Inayatul Arifah, I., Nur Fajri, F., & Qorik Oktagalu Pratamasunu, G. (2022). Deteksi Tangan Otomatis Pada Video Percakapan Bahasa Isyarat Indonesia Menggunakan Metode YOLO Dan CNN. Dalam *Journal of Applied Informatics and Computing (JAIC)* (Vol. 6, Nomor 2). http://jurnal.polibatam.ac.id/index.php/JAIC

Mursita, R. A., Tunarungu, R., Pascasarjana, M., & Upi Bandung, P. (t.t.). *RESPON TUNARUNGU TERHADAP PENGGUNAAN SISTEM BAHASA ISYARAT INDONESA (SIBI) DAN BAHASA ISYARAT INDONESIA (BISINDO) DALAM KOMUNIKASI Rohmah Ageng Mursita*. http://www.change.org/id/petisi/

Nababan, D. R. M., & Budiarso, Z. (2023). Sistem Pendeteksi Gerakan Bahasa Isyarat Indonesia Menggunakan Webcam Dengan Metode Supervised Learning. *Jurnal Ilmiah Komputasi*, *22*(3). https://doi.org/10.32409/jikstik.22.3.3403

Nyoman Tri Anindia Putra, I., Sepdyana Kartini, K., Kristian Suyitno, Y., Made Sugiarta, I., & Kadek Era Puspita, N. (2023). *Penerapan Library Tensorflow, Cvzone, dan Numpy pada Sistem Deteksi Bahasa Isyarat Secara Real Time* (Vol. 2, Nomor 3). https://ejournal.sidyanusa.org/index.php/jkdn

Suyudi, I., Sudadio, S., & Suherman, S. (2023). Pengenalan Bahasa Isyarat Indonesia menggunakan Mediapipe dengan Model Random Forest dan Multinomial Logistic Regression. *Jurnal Ilmu Siber dan Teknologi Digital*, *1*(1), 65–80. https://doi.org/10.35912/jisted.v1i1.1899

Tri Hermanto, D., Setyanto, A., & Luthfi, E. T. (t.t.). *Algoritma LSTM-CNN untuk Sentimen Klasifikasi dengan Word2vec pada Media Online LSTM-CNN Algorithm for Sentiment Clasification with Word2vec On Online Media*.

Wen, X., & Li, W. (2023). Time Series Prediction Based on LSTM-Attention-LSTM Model. *IEEE Access*, *11*, 48322–48331. https://doi.org/10.1109/ACCESS.2023.3276628