Jurnal Ilmiah Sistem Informasi dan Ilmu Komputer
Volume 5, Nomor 2, Juli 2025
p-ISSN: 2827-8135; e-ISSN: 2827-7953, Hal. 405-423
DOI: https://doi.org/10.55606/juisik.v5i2.1386
Available Online at: https://journal.sinov.id/index.php/juisik

Deteksi Pose *Semaphore* Berbasis *Deep Learning* Menggunakan Metode *Multi-Layer Perceptron* dan *MediaPipe*

Milton Dapamudang[1*], Fajar Hariadi[2], Raynesta Mikaela Indri Malo[3]
[1,2,3] Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Kristen Wira Wacana Sumba, Indonesia
miltondapamudang@gmail.com [1*], *fajar@unkriswina.ac.id* [2], *raynesta@unkriswina.ac.id* [3]

Alamat: Jl. R. Suprapto No.35, Prailiu, Kec. Kota Waingapu, Kabupaten Sumba Timur, Nusa Tenggara Timur.
Korespondensi penulis: miltondapamudang@email.com

Abstract*. Semaphore is a communication method used to send and receive messages by utilizing flags, paddles, sticks, bare hands, or gloves. Semaphore signaling is commonly used in scouting activities (Pramuka). However, novice scouts often face difficulties in accurately recognizing semaphore gestures. Although guidebooks are available, errors may still occur, especially when learning independently. To address this issue, a gesture detection system using the Multi-Layer Perceptron (MLP) method and MediaPipe framework is proposed, which aims to evaluate the accuracy of MLP and MediaPipe in detecting semaphore poses. To assess the performance of the MLP model, a Confusion Matrix is used to analyze the number of correct and incorrect predictions and generate metrics such as accuracy, precision, recall, and F1-score. The evaluation results show that the system successfully recognized 122 poses correctly, achieving an accuracy of 93.84%, precision of 93.84%, recall of 100%, and an F1-score of 96.80%. Based on these results, the system is considered sufficiently accurate and can be used as an interactive learning tool to recognize semaphore codes, particularly in scouting activities*.

Keywords: *Confusion Matrix, MediaPipe, Multi-Layer Perceptron, Pose Detection, Semaphore*

Abstrak. *Semaphore* merupakan metode komunikasi untuk mengirim dan menerima pesan dengan memanfaatkan bendera, dayung, batang, tangan kosong, atau sarung tangan. Komunikasi menggunakan sandi *Semaphore* sering digunakan dalam Pramuka. Namun, masalah yang dihadapi anggota Pramuka yang baru belajar adalah kesulitan dalam mengenali gerakan sandi *semaphore* secara akurat. Meskipun terdapat buku panduan (Manual Book), namun kesalahan dapat terjadi apalagi jika belajar secara mandiri. Untuk menjawab masalah tersebut akan dikembangkan sistem deteksi gerakan *semaphore* menggunakan metode *Multi-layer Perceptron* (MLP) dan *MediaPipe*. Penelitian ini bertujuan untuk menguji seberapa besar akurasi MLP dan *MediaPipe* dalam mendeteksi pose *Semaphore*. Untuk menguji akurasi model MLP (*Multilayer Perceptron*), digunakan metode *Confusion Matrix* yang membantu mengevaluasi kinerja model berdasarkan jumlah prediksi yang benar dan salah, serta menghasilkan metrik seperti akurasi, presisi, *recall*, dan F1-*score*. Hasil pengujian menunjukkan bahwa sistem mampu mengenali 122 pose dengan benar, menghasilkan akurasi sebesar 93,84%, *precision* 93,84%, *recall* 100%, dan F1-score 96,80%. Berdasarkan hasil tersebut, sistem dinilai cukup akurat dan dapat digunakan sebagai alat bantu pembelajaran interaktif dalam mengenali sandi *semaphore*, khususnya pada kegiatan Pramuka.

Kata kunci: Matriks Kebingungan, MediaPipe, Perseptron Multi-Lapisan, Deteksi Pose, Semafor

1. LATAR BELAKANG

Semaphore merupakan metode komunikasi visual menggunakan simbol atau gerakan tangan yang digunakan saat komunikasi verbal tidak memungkinkan, misalnya karena jarak jauh atau keterbatasan alat komunikasi. Dalam kegiatan Pramuka, *semaphore* menjadi keterampilan dasar yang wajib dikuasai oleh anggota, terutama pemula Dwika dkk (2024). Namun, mengenali gerakan pose *semaphore* secara tepat masih menjadi tantangan, karena gerakan antar huruf sering kali mirip dan belajar mandiri tanpa pengawasan pelatih dapat menyebabkan kesalahan interpretasi. Kesalahan ini dapat terjadi baik dari sisi pengirim yang

Received Juni 09, 2025; Revised Juni 25, 2025; Accepted Juli 09, 2025; Published Juli 11, 2025

salah membuat gerakan, maupun dari sisi penerima yang salah dalam menafsirkan pose, sehingga menyebabkan informasi tidak tersampaikan dengan akurat.

Untuk menjawab permasalahan tersebut, diperlukan sistem yang mampu mengenali pose *semaphore* secara otomatis dan *real-time*. Teknologi *deep learning*, khususnya dengan metode *Multi-Layer Perceptron* (MLP), dipilih karena kemampuannya dalam mengenali pola data *nonlinier* dengan tingkat akurasi tinggi. Ojiako dan Farahi (2023) menunjukkan bahwa model pengenalan aktivitas manusia berbasis Multi-Layer Perceptron (MLP) yang dikembangkan mampu mencapai akurasi tinggi pada berbagai *dataset benchmark*. Misalnya, model tersebut mencapai akurasi 97,1% pada *dataset* PAMAP2, yang mencakup berbagai aktivitas fisik seperti berjalan, berlari, duduk, dan berdiri. Hasil ini mengindikasikan bahwa MLP cukup efektif dalam menangkap pola-pola sinyal sensor dari aktivitas tubuh manusia. Namun, ketika diuji pada *dataset* lain seperti *Opportunity Gestures* yang melibatkan gerakan tangan kompleks (misalnya membuka pintu atau mengambil benda), akurasi turun menjadi 68%. Ini menandakan bahwa meskipun MLP efektif untuk aktivitas kasar (*gross* motor *skills*), performanya menurun ketika dihadapkan pada aktivitas halus (*fine* motor *skills*) yang lebih kompleks secara spasial dan temporal. Hal ini menunjukkan bahwa model MLP masih memiliki keterbatasan dalam menangani aktivitas yang memiliki variasi gerakan kecil namun signifikan. Sarakon dkk (2025) menyatakan bahwa penerapan MLP klasik dalam pengenalan aktivitas inti seperti jalan, lari, dan naik/turun tangga menunjukkan performa konsisten dengan rata-rata akurasi 90%. Menariknya, model ini tetap efektif meskipun menggunakan fitur statistik sederhana dari data sensor, menandakan efisiensi MLP dalam mengolah data numerik tanpa memerlukan arsitektur kompleks seperti CNN atau RNN.

Sementara itu, *MediaPipe* biasa digunakan untuk pemrosesan secara *real-time* baik untuk pengenalan wajah, deteksi pose tubuh, pengenalan objek, dan masih banyak lagi. *MediaPipe* dapat digunakan untuk membangun sistem alternatif dalam belajar kebugaran tanpa harus ke GYM (*Gymnasium*), sistem tersebut mendeteksi fase *squat* (turun & naik) divalidasi secara *real-time* dan memiliki akurasi 100% Ragil dkk (2025). Penelitian lain yang memanfaatkan *mediapipe* dalam bidang kesehatan untuk memantau secara *real-time* pasien fisioterapi dalam melakukan latihan rehabilitasi tanpa pengawasan pelatih profesional *fisioterapis* juga dikembangkan. Sistem ini mampu secara *real-time* mendeteksi postur secara akurat dilihat dari 3 jenis latihan yang dilakukan, *push-up* 96.5%, *bridge* 88.0%, *shoulder roll* 85,98% Dudekula dkk (2024). Ojiako dan Farrahi (2023) menyatakan bahwa model MLP-*Mixer* mampu menggantikan arsitektur CNN dan RNN dalam klasifikasi aktivitas berbasis sensor. Sarakon

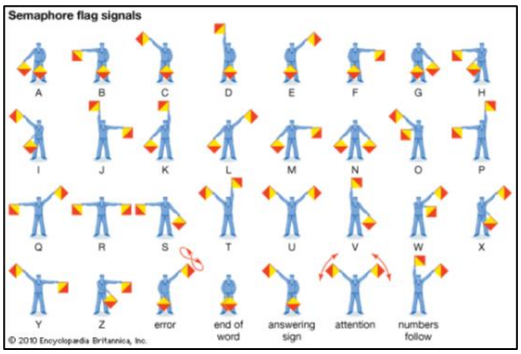
dkk (2025) menyatakan bahwa penelitian yang dilakukan berhasil menerapkan MLP klasik untuk mengklasifikasikan data dari berbagai sumber menggunakan fitur statistik.

Berbeda dari penelitian sebelumnya yang fokus pada aktivitas umum, penelitian ini mengembangkan sistem deteksi pose *semaphore* dengan menggabungkan *MediaPipe* untuk ekstraksi pose dan MLP sebagai model klasifikasi. Tujuannya adalah menghasilkan sistem bantu belajar interaktif bagi anggota Pramuka dalam mengenali sandi *semaphore* secara akurat dan efisien dan pengguna juga akan menyusun kata yang telah ditentukan oleh sistem. Fokus penelitian ini adalah menerapkan metode MLP dalam pengembangan sistem deteksi pose *semaphore*.

2. KAJIAN TEORITIS

Semaphore

Semaphore merupakan salah satu bentuk komunikasi nonverbal yang digunakan untuk menyampaikan informasi melalui gerakan isyarat, khususnya menggunakan alat bantu sederhana seperti bendera atau posisi tubuh tertentu. Metode ini tidak menggunakan kata-kata, melainkan memanfaatkan posisi atau gerakan tangan dalam pola tertentu untuk mewakili simbol atau huruf tertentu dalam alfabet. Sistem sandi *semaphore* secara umum terdiri atas 26 huruf alfabet dan satu kode khusus tambahan, sehingga memungkinkan komunikasi yang cukup kompleks dalam kondisi di mana komunikasi verbal tidak memungkinkan Alfarezi (2024). Metode komunikasi menggunakan dua bendera berwarna sebagai alat bantu. Pesan disampaikan melalui posisi tangan yang membentuk pola tertentu, di mana setiap posisi mewakili huruf atau simbol Putra dkk (2024).



Gambar 1. Pose Huruf - Huruf Pada *Semaphore*

Keterangan: Pose Huruf Semaphore dari Huruf A - Z.

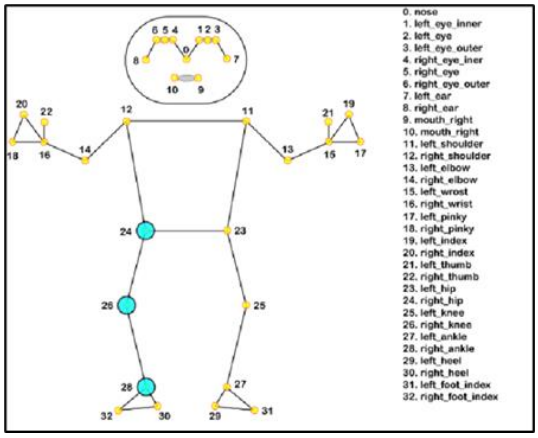
Sumber: *www.researchgate.net*.

a. *Deep Learning*

Deep Learning adalah cabang dari *Machine Learning* (ML) yang meniru cara kerja otak manusia dalam memproses data dan membuat keputusan. Ini dilakukan dengan menggunakan *Artificial Neural Networks* (ANN) yang memiliki banyak lapisan (*deep*), yang memungkinkan komputer belajar langsung dari data tanpa aturan yang jelas Peryanto dkk (2020). *Deep Learning* adalah pengembangan dari Jaringan Saraf Tiruan (JST) atau *Artificial Neural Networks* (ANN). Oleh karena itu, *Deep Learning* secara langsung menerapkan model berbasis JST tetapi dengan lebih banyak lapisan atau *deep* Handayanto dan Herlawati (2020).

b. *MediaPipe*

MediaPipe dirancang untuk menerapkan kecerdasan buatan pada aplikasi yang akan dibangun. Alat analisis visual dalam waktu nyata seperti deteksi pose tubuh manusia, deteksi wajah, dan pelacakan tangan tersedia di *library* ini. Pengembang dapat dengan mudah memasukkan analisis visual yang kompleks ke dalam sistem dengan *MediaPipe* tanpa harus membangun semua komponennya dari awal Alfarezi (2024). *MediaPipe* berfungsi mengidentifikasi gerakan tubuh manusia secara *real-time* serta memberikan hasil dan akurasi Dwika dkk (2024).



Gambar 2. *Keypoint Mediapipe*

Keterangan: Titik Koordinat Tubuh pada *Mediapipe*.

Sumber: www.researchgate.net.

c. *Multi Layer Perceptron* (MLP)

Multi-Layer Perceptron (MLP) merupakan salah satu bentuk arsitektur dari jaringan saraf tiruan (*Artificial Neural Network*) yang memiliki sejumlah lapisan neuron, termasuk minimal satu *hidden layer* di antara *input layer* dan *output layer*. MLP dikategorikan sebagai jaringan saraf *feedforward*, yang berarti aliran data hanya bergerak satu arah dari *input* menuju *output* tanpa adanya umpan balik. Proses kerjanya melibatkan penghitungan bobot dari data

masukan, kemudian diterapkan fungsi *aktivasi* untuk menghasilkan keluaran Gulo dkk (2024). MLP merupakan salah satu metode dasar dalam pengenalan pola yang tersusun dari sejumlah neuron yang dikelompokkan dalam beberapa lapisan. Lapisan awal disebut sebagai input layer, sedangkan lapisan akhir dinamakan output layer, dengan satu atau lebih hidden layer di antaranya yang berfungsi untuk memproses informasi secara bertahap Resha dkk (2022).

d. *Open Source Computer Vision (Open CV)*

OpenCV *(Open Source Computer Vision Library)* adalah pustaka *open-source* berbasis *C++* yang dirancang untuk pemrosesan citra dan visi komputer secara *real-time* Suradi dkk (2023). *OpenCV* dapat digunakan pada berbagai *platform*, sehingga dapat diterapkan secara luas pada gambar atau video seperti *face recognition, face detection, object tracking, road tracking* dan lain-lain. *OpenCV* menggunakan metode *Computer Vision,* yang memungkinkan komputer melihat objek dengan cara yang mirip dengan manusia, yang memungkinkan mereka untuk mengambil keputusan dan melakukan tindakan berdasarkan apa yang dideteksi Sejati dkk (2021).

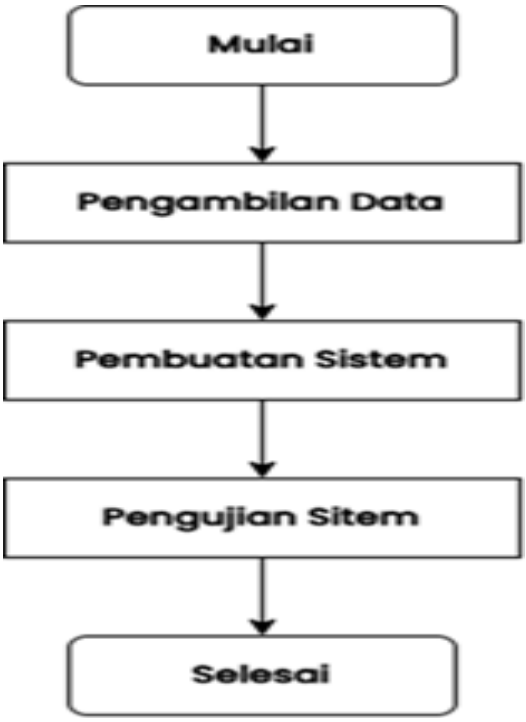
e. *TensorFlow*

Tensorflow adalah sebuah *library* open source untuk pembelajaran mesin dan deep learning Sitorus (2022). *Library* ini banyak digunakan untuk membangun dan melatih berbagai model kecerdasan buatan, termasuk sistem pengenalan objek secara *real-time* Mutianniza (2023).

f. *Python*

Python merupakan bahasa pemrograman tingkat tinggi yang dikembangkan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991. Bahasa ini dikenal karena kesederhanaan sintaksisnya, bersifat multifungsi, serta didukung oleh berbagai *library* dan komunitas *open-source* yang besar Alfarizi dkk (2023). *Python* merupakan bahasa pemrograman interpretatif yang bersifat *multiplatform*, mudah dipelajari, dan memiliki sintaksis yang jelas serta terbaca Wati dkk (2021).

3. METODE PENELITIAN

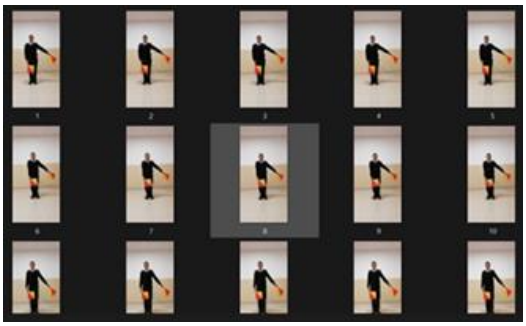


Gambar 3. Alur Penelitian

Penelitian ini dilakukan mulai dari tahap pertama adalah pengambilan data yang dilakukan dengan cara mengumpulkan *dataset*. Tahap kedua adalah pembuatan sistem, pembuatan sistem dibagi menjadi dua bagian yaitu pembuatan model dan pengujian model. Tahap yang terakhir yaitu pengujian, yang bertujuan untuk mengukur akurasi model dalam memprediksi pose *semaphore*.

a. Pengambilan Data

Pengambilan data dilakukan dengan cara mengumpulkan *dataset* pose *semaphore* yang diperoleh dari situs penyedia *dataset* yaitu https://data.mendeley.com. *Dataset* ini terdiri dari 26 folder yang setiap folder menyimpan pose huruf. Satu folder mewakili satu huruf yang di dalamnya terdapat 20 data gambar pose huruf yang sama. Jadi, terdapat 26 kelas dengan masing-masing memiliki 20 data.



Gambar 4. *Dataset* Pose *Semaphore*

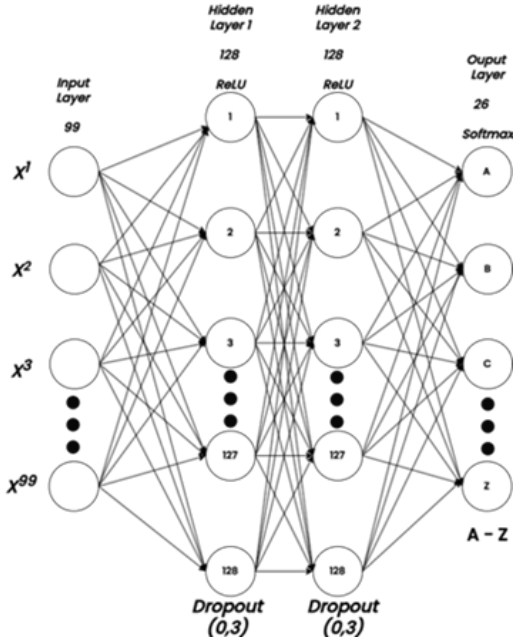
b. Pembuatan Sistem

Pembuatan sistem dibagi menjadi dua bagian utama, yaitu pembuatan model dan pengujian model. Pembuatan model dilakukan melalui dua tahap, *preprocessing* dan *training*. Pada tahap *Preprocessing*, *dataset* yang digunakan terdiri dari 26 folder yang masing-masing merepresentasikan huruf A hingga Z, dan diolah menggunakan bahasa pemrograman *Python* dengan bantuan pustaka *OpenCV*, *NumPy*, dan *MediaPipe*. *OpenCV* digunakan untuk membaca gambar dan mengonversinya dari format BGR ke RGB. Selanjutnya, *MediaPipe* mengekstraksi 33 titik pose (*landmark*) dari setiap gambar, di mana setiap titik memiliki tiga koordinat (x, y, z), sehingga menghasilkan total 99 fitur numerik per gambar. Hasil ekstraksi tersebut disimpan dalam *array NumPy*, dengan variabel X berisi vektor 99 fitur dan Y berisi label huruf yang sesuai ('A' hingga 'Z').

Tabel 1. Data Titik Koordinat Setiap Pose

No	x0	y0	z0	…	x32	y32	z32	Label
0	0.587010	0.222340	0.579298	…	0.528466	0.726324	0.726324	A
1	0.562313	0.257021	0.257021	…	0.599563	0.599563	0.236515	B
..	…	…	…	…	…	…	…	..
519	0.236515	0.236515	0.236515	…	0.236515	0.236515	0.236515	Y
520	0.558681	0.558681	0.558681	…	0.558681	0.558681	0.558681	Z

Kemudian pada tahap *Training*, model akan dilatih menggunakan arsitektur di bawah ini:



Gambar 5. Arsitektur MLP

Gambar 5 adalah arsitektur model MLP yang akan digunakan pada saat *training*. Arsitektur di atas menggambarkan bagaimana model bekerja dan melakukan prediksi mulai dari *Input Layer*, *Hidden Layer 1*, *Hidden Layer 2*, *Output Layer*. Dalam membangun arsitektur MLP, struktur dua *hidden layer* masing-masing berukuran 128 neuron dipilih berdasarkan hasil eksplorasi Przybyła-Kasperek dan Marfo (2024) menyatakan bahwa penggunaan dua *hidden layer* memberikan performa klasifikasi optimal.

a. *Input Layer*

Bertujuan untuk menyampaikan informasi data awal (fitur pose *semaphore*) ke *hidden layer*. Di mana fitur pose diperoleh dari hasil ekstraksi dari *MediaPipe*. *MediaPipe* menghasilkan 33 titik pose yang setiap titiknya memiliki 3 nilai yaitu x (horizontal), y (vertikal), dan z (kedalaman). Sehingga, 33 titik x 3 = 99 fitur nilai. Ini berarti pada *input layer* terdapat 99 *neuron*. Dinotasikan menjadi:

$$Input = X = [x_1, x_2, x_3 \dots, x_{99}]$$

b. *Hidden Layer* 1

Hidden Layer 1 terdapat 128 neuron yang berfungsi menerima semua nilai dari *input layer*. Pada bagian *hidden layer* 1 ini, setiap neuron melakukan perhitungan untuk mengekstrak pola kompleks dari fitur-fitur pose. Setiap neuron melakukan perhitungan menggunakan rumus *Linear Combination* berikut.

$$z = w_1 x_1 + w_2 x_2 + \cdots + w_{99}\, x_{99} + b$$

Atau disederhanakan menjadi,

$$z_j^{(1)} = \sum_{i=1}^{99} x_i \,.\, w_{ij}^{(1)} \; + \; b_j^{(1)}$$

Keterangan:

x_i = *input* ke-i dari nilai koordinat pose

$w_{ij}^{(1)}$ = bobot dari *input* ke-i ke *neuron* ke-j di *layer* 1

$b_j^{(1)}$ = bias untuk *neuron* ke-j

$z_j^{(1)}$ = *output* sebelum aktivasi untuk *neuron* ke-j.

Setelah mendapat nilai $z_j^{(1)}$, akan di aktivasi menggunakan fungsi *ReLu* (*Rectified Linear Unit*). Ini berfungsi jika nilai $z_j^{(1)}$ merupakan bilangan positif maka akan diloloskan ke *layer* berikutnya dan jika bilangannya *negative* maka dijadikan 0. Fungsi *ReLU* membuat jaringan hanya meneruskan jaringan positif untuk mempermudah pelatihan. Berikut rumusnya:

$$ReLU(z) = max(0, z)$$

Keterangan:

z = nilai dari $z_j^{(1)}$

$max(0, z)$ = mengambil nilai antara 0 dan z

$ReLU(z)$ = *output* setelah fungsi diterapkan.

c. *Dropout*

Dalam *hidden layer* 1 yang terdapat 128 neuron akan dilakukan *dropout* sebesar 30% pada saat *training* sekitar 38 *neuron*. Artinya akan dimatikan secara acak tujuannya adalah mencegah *overfitting* (model terlalu bagus dalam menghafal data latih, tapi jelek dalam memprediksi data baru). Ini membuat model tidak hanya menghafal pose tertentu tapi lebih *general*. *Dropout* pada kisaran 20%–30% terbukti memberikan efek regularisasi yang baik Cai dkk (2019).

d. *Hidden Layer 2*

Hidden Layer 2, Pada *hidden* layer 2 ini, terdapat *neuron* sebanyak 128. *Neuron-neuron* ini juga melakukan operasi yang sama seperti pada layer sebelumnya, setelah nilai di dapati di aktivasi lagi menggunakan fungsi *ReLU*. Berikut rumusnya:

$$z_k^{(2)} = \sum_{j=1}^{128} a_j^{(1)} . w_{jk}^{(2)} + b_k^{(2)}$$

Keterangan:

$a_j^{(1)}$ = *output* dari *Layer* 1 (setelah *dropout*)

$w_{jk}^{(2)}$ = bobot dari *neuron* ke-j ke *neuron* ke-k

$b_k^{(2)}$ = bias *neuron* ke-k

$z_k^{(2)}$ = *output* sebelum aktivasi

Fungsi Aktivasi:

$$a_k^{(2)} = ReLU(z_k^{(2)})$$

e. *Dropout*

Seperti sebelumnya, sebesar 30% dari *output neuron* pada *layer* 2 dimatikan secara acak pada saat *training*. Tujuannya agar model terlalu bergantung pada *neuron* tertentu.

f. *Output Layer*

Bagian terakhir dari arsitektur ini adalah *ouput layer*, akan digunakan fungsi aktivasi *softmax* untuk mengubah skor menjadi probabilitas. Pada *layer* ini terdapat 26 *neuron*, diaman

neuron ini menerima *input* hasil aktivasi dari setiap *neuron* pada *layer* 2. Karena ada label untuk setiap *input* pose sebanyak 26 kelas, maka model perlu 26 neuron pada *ouput layer*. Perhitungan nilai pada neuro nya tetap sama tapi kali ini tidak lagi menggunakan fungsi ReLU melainkan fungsi *Softmax* untuk melakukan prediksi

$$z_h^{(3)} = \sum_{k=1}^{128} a_k^{(2)} . w_{kh}^{(3)} + b_h^{(3)}$$

Keterangan:

$a_k^{(2)}$ = *output* dari *layer* 2

$w_{kh}^{(3)}$ = bobot dari *neuron* ke-k ke *output* ke-h (huruf A–Z)

$b_h^{(3)}$ = bias untuk huruf ke-h

Kemudian digunakan fungsi Aktivasi *Softmax* untuk mengubah skor menjadi probabilitas yang bernilai antara 0 – 1.

$$softmax(z_h) = \frac{e^{z_h}}{\sum_{m=1}^{26} e^{z_m}}$$

Keterangan:

z_h = *logit* (*output mentah*) dari kelas ke-h (misalnya *output* untuk huruf "C").

e^{z_h} = menaikkan skala ke eksponensial, membuat perbedaan skor lebih mencolok.

$\sum_{m=1}^{26} e^{z_m}$ = jumlah dari eksponensial semua *logit* digunakan untuk normalisasi agar hasilnya adalah probabilitas (0–1) dan totalnya 1.

g. *Backpropagation*

Setelah dilakukan *forward pass* dan memperoleh *output* berupa probabilitas dari fungsi aktivasi *softmax*, maka langkah selanjutnya dalam proses pembelajaran adalah melakukan *update* terhadap bobot dan bias menggunakan metode *backpropagation*. Tujuan utama proses ini adalah meminimalkan nilai *loss* (kesalahan prediksi) agar model semakin akurat. Pada saat proses *traning* digunakan *learning rate*:

$$\eta = 0.01$$

Learning rate adalah parameter yang mengatur seberapa besar langkah pembaruan bobot yang dilakukan model MLP selama proses pelatihan.

Langkah pertama dalam *backpropagation* adalah menghitung Kesalahan (*error*) pada lapisan *output* menggunakan rumus:

$$\delta^{output} = \hat{y} - y$$

Dalam persamaan ini, $\hat{y}$ merupakan keluaran prediksi dari model yang biasanya telah melalui fungsi aktivasi *softmax*, sedangkan y adalah label sebenarnya dalam format *one-hot encoding*. Nilai δ^{output} mencerminkan seberapa jauh prediksi model dari nilai yang diharapkan dan menjadi dasar untuk perhitungan berikutnya.

Kesalahan ini digunakan untuk menghitung gradien dari bobot pada lapisan *output*, dengan rumus:

$$W^{(3)} = a^{(2)} \times \delta^{(output)}$$

Nilai $a^{(2)}$ adalah nilai aktivasi dari *hidden layer* 2, yang menjadi masukan bagi lapisan *output*. Gradien $W^{(3)}$ menunjukkan perubahan yang perlu dilakukan terhadap bobot untuk mengurangi kesalahan keluaran.

Pembaruan bobot dilakukan menggunakan rumus *gradient descent*:

$$W^{(baru)} = W^{(lama)} - \eta \times W$$

Secara paralel, bias pada lapisan *output* juga diperbarui dengan rumus:

$$b^{(baru)} = b^{(lama)} - \eta \times \delta$$

Proses perhitungan *error*, *update* bobot dan bias ini di *propagation* dan dilakukan hingga ke *hidden layer* 1 selama putaran *training* (*epoch*).

Pengujian Akurasi Sistem

Pengujian sistem dilakukan untuk mengukur akurasi model dalam memprediksi pose *semaphore*. Model diuji menggunakan Anggota pramuka yang telah mahir pada pose *semaphore.* Pengujian ini untuk mengukur seberapa besar akurasi sistem dalam melakukan prediksi pose huruf *semaphore* dari huruf A-Z. Menggunakan metode evaluasi yaitu *confusion matrix* dan metrik seperti *Accuracy*, *Precision*, *Recall*, dan F1-*Score.* Berikut rumus setiap metrik pengujian:

a. *Accuracy*

Akurasi mengukur seberapa banyak huruf yang dikenali dengan benar saat pengujian.

$$Accuracy = \frac{TP}{TP + FP + FN} \times 100\ \%$$

b. *Precision*

Precision mengukur keakuratan model dalam memprediksi huruf yang dikenali.

$$Precision = \frac{TP}{TP + FP} \times 100\ \%$$

c. *Recall*

Recall mengukur kemampuan model dalam mengenali huruf yang seharusnya dikenali selama proses pengujian.

$$Recall = \frac{TP}{TP + FN} \times 100\ \%$$

d. Keseluruhan Keseimbangan Model (*F1-Score).*

F1-Score menggabungkan *Precision* dan *Recall* untuk mengukur keseimbangan kemampuan model dalam mengenali huruf dengan benar serta menghindari kesalahan.

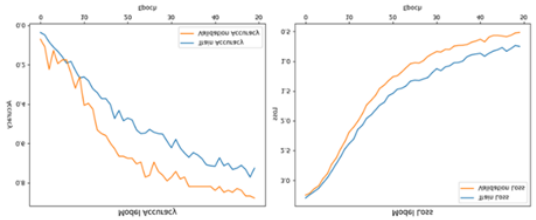
$$F1 = \ 2 \times \frac{Precision\ x\ Recall}{Precision + Recall}$$

Keterarangan:

1) TP = *True Positive* (prediksi dan aktual sama)
2) FP = *False Positive* (prediksi salah)
3) FN = *False Negative* (aktual benar tapi prediksi salah)

4. HASIL DAN PEMBAHASAN

Sistem deteksi pose *semaphore* telah berhasil dikembangkan dan diuji sebanyak 130 kali pose oleh anggota pramuka, dan masing-masing huruf diuji sebanyak 5 kali pose. Berdasarkan pengujian yang dilakukan, sistem dapat mengenali sebanyak 122 pose dengan baik dan kesalahan mengenali pose sebanyak 8 kali kesalahan. Namun, model tetap mampu menunjukkan generalisasi yang kuat karena *dropout* sebesar 30% pada masing-masing *hidden* layer untuk mengurangi *overfritting*. Hal ini juga didukung dengan Grafik Performa Model selama *traning* di bawah ini.



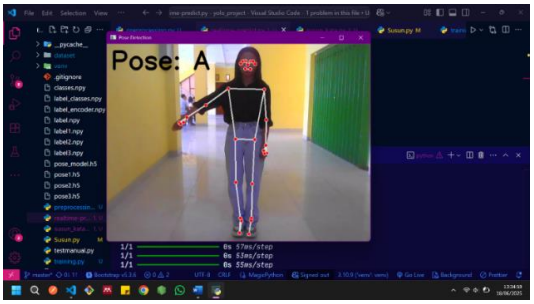
Gambar 6. Grafik Performa Model

Gambar di atas merupakan grafik kurva pembelajaran (*learning curve*) dari hasil *training* model. Grafik ini terdiri dari dua bagian, yaitu grafik akurasi (*Accuracy*) di sebelah kiri dan grafik kesalahan (*Loss*) di sebelah kanan. Grafik ini membantu menunjukkan perkembangan model selama proses pelatihan sebanyak 50 *epoch* atau putaran. Pada grafik akurasi, sumbu mendatar (X) menunjukkan jumlah *epoch*, yaitu dari 1 sampai 50, sedangkan sumbu vertikal (Y) menunjukkan nilai akurasi, dengan rentang dari 0 (0%) hingga 1 (100%). Garis biru

menunjukkan *Train Accuracy*, yaitu tingkat keberhasilan model saat mempelajari data latih, sedangkan garis oranye menunjukkan *Validation Accuracy*, yaitu tingkat keberhasilan model saat diuji dengan data yang belum pernah dilihat sebelumnya (data uji). Misalnya, jika pada *epoch* ke-40 garis oranye berada di angka 0.85, itu berarti pada putaran ke-40, model berhasil mengenali pose dengan benar sebanyak 85% dari total data validasi. Berdasarkan grafik tersebut, terlihat bahwa akurasi meningkat secara konsisten seiring bertambahnya *epoch*. Pada awal pelatihan (sekitar *epoch* 1–10), akurasi masih rendah, yaitu di bawah 30%, karena model masih dalam tahap awal belajar. Namun setelah melalui proses pelatihan, akurasi terus naik dan pada *epoch* ke-50, *Train Accuracy* mencapai sekitar 72%, dan *Validation Accuracy* mencapai sekitar 87%, yang menandakan model bekerja dengan baik pada data baru dan tidak hanya menghafal data pelatihan.

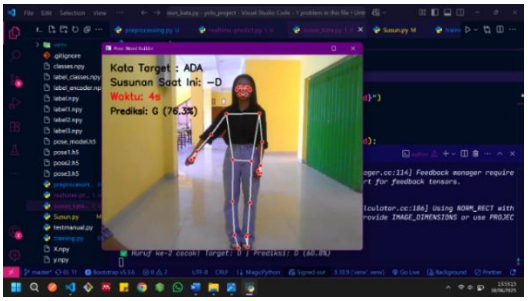
Sementara itu, pada grafik *loss*, sumbu mendatar (X) juga menunjukkan jumlah *epoch*, dan sumbu vertikal (Y) menunjukkan nilai *loss* atau kesalahan model. Nilai *loss* menunjukkan seberapa jauh prediksi model dari label yang sebenarnya semakin kecil nilainya, semakin baik. Garis biru menunjukkan *Train Loss*, sedangkan garis oranye menunjukkan *Validation Loss*. Pada awal pelatihan, nilai *loss* masih sangat tinggi (sekitar 3.2), yang berarti prediksi model masih banyak salah. Namun seiring waktu, nilai *loss* turun secara signifikan, hingga pada akhir pelatihan (*epoch* ke-50), *Validation Loss* mencapai nilai di bawah 0.5, artinya model sudah sangat jarang melakukan kesalahan. Secara keseluruhan, grafik ini menunjukkan bahwa model berhasil belajar dengan baik: akurasi meningkat dan *loss* menurun, baik pada data latih maupun data uji. Tidak terlihat gejala *overfitting*, karena garis validasi selalu mengikuti atau bahkan lebih baik dari garis *training*. Model tidak hanya menghafal data latih, tetapi juga mampu mengenali pose baru dengan baik. Hal ini menjadikan model sangat layak untuk digunakan dalam implementasi deteksi pose *semaphore* secara *real-time*.

Proses pengenalan huruf pada sistem ini dimulai dari pengambilan *input* pose melalui kamera *webcam*, yang berfungsi untuk menangkap gerakan tubuh pengguna secara *real-time*. Selanjutnya, dilakukan ekstraksi titik pose menggunakan pustaka *MediaPipe*, yang mendeteksi 33 titik pose tubuh dan menghasilkan tiga koordinat (X, Y, Z) untuk masing-masing titik, sehingga diperoleh total 99 fitur dari satu pose. Fitur-fitur ini kemudian dikirim ke model *Multi-Layer Perceptron* (MLP) untuk dilakukan proses klasifikasi guna memprediksi huruf dari A hingga Z. Terakhir, huruf yang berhasil dikenali akan ditampilkan satu per satu pada antarmuka sistem, dan dapat digunakan untuk menyusun kata dasar yang telah ditentukan sebelumnya.



Gambar 7. Tampilan Deteksi Pose

Gambar 7 adalah tampilan sistem yang digunakan untuk deteksi pose huruf *semaphore* di mana sistem akan secara *real-time* memprediksi pose yang dilakukan pengguna.



Gambar 8. Tampilan Menyusun Kata

Gambar 8 adalah tampilan sistem pada saat pengguna menyusun kata dari pose huruf. Kata yang muncul merupakan kata yang telah ditentukan pada saat pengembangan sistem.

Pengujian dilakukan secara langsung (*realtime*) menggunakan *webcam* untuk mendeteksi pose tubuh dari pengguna yang memperagakan sandi *semaphore.* Hasil pengujian dicatat secara manual selama proses berlangsung, kemudian disusun ke dalam tabel *confusion matrix.* Berikut tabel *confusion matrix* hasil pengujian:

Tabel 2. *Confusion Matrix* Hasil Pengujian

Actual\ Predicted	**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**
A	**5**																									
B		**3**	**2**																							
C			**5**																							
D				**5**																						
E					**4**																					
F						**3**						**2**														
G							**4**																			
H								**5**																		
I									**5**																	
J										**5**																
K											**5**															
L												**5**														
M													**5**													

N			**2**										**3**												
O														**5**											
P															**5**										
Q																**4**			**1**						
R																	**5**								
S																		**5**							
T																			**5**						
U																				**5**					
V																					**5**				
W																						**5**			
X																							**5**		
Y																					**1**			**4**	
Z																									**5**

Berdasarkan hasil pengujian , diperoleh:

- *True Positive* (pose yang berhasil dikenali dengan benar) = 122
- *False Positive* (pose yang salah dikenali sebagai huruf lain) = 8
- *False Negative* (pose yang gagal dikenali/tidak terbaca sistem) = 0

Sehingga dihitung setiap metrik berikut ini:

a. *Accuracy*

$$Accuracy = \frac{TP}{Total\ Pengujian} = \frac{122}{130} \times 100\% = 0.93 = 93.84\%$$

Terdapat 122 pose huruf yang dikenali dengan benar dari 130 pose yang diuji. Ini berarti sekitar 93.84% dari seluruh pengujian berhasil diklasifikasikan dengan tepat oleh sistem. Angka ini menunjukkan bahwa secara keseluruhan, sistem memiliki tingkat akurasi yang sangat baik.

b. *Precision*

$$Precision = \frac{TP}{TP + FP} = \frac{122}{122 + 8} = \frac{122}{130} \times 100\% = 0.93 = 93.84\%$$

Dari 130 prediksi sistem, sebanyak 122 benar (TP) dan 8 salah (FP). Dengan hasil *precision* sebesar 93.84%, berarti sebagian besar prediksi sistem adalah benar, dan hanya sebagian kecil yang salah diklasifikasikan sebagai huruf yang keliru.

c. *Recall*

$$Recall = \frac{TP}{TP + FN} = \frac{122}{122 + 0} = \frac{122}{122} \times 100\% = 100\%$$

Karena tidak ada pose huruf yang gagal dikenali (FN = 0), maka recall mencapai 100%. Ini berarti semua huruf yang diperagakan oleh pengguna berhasil terbaca oleh sistem, meskipun ada sebagian kecil yang diklasifikasikan secara salah (masuk ke *precision*).

d. F1-*Score*

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.9384 \times 1.0}{0.9384 + 1.0} = 2 \times 0.4840 \approx 0 = 96.80\%$$

Dengan F1-*score* 96.80%, sistem dinilai sangat seimbang dan andal, karena mampu mengenali hampir semua pose dengan benar dan hanya sedikit melakukan kesalahan klasifikasi.

Perhitungan di atas merupakan perhitungan keseluruhan *confusion matrix* dari keseluruhan model. Berikut penjelasan secara garis besar mengenai hasil evaluasi *confusion matrix* untuk tiap huruf (A-Z) pada tabel di bawah ini:

Tabel 3. *Confusion Matrix* Setiap Huruf

Huruf	TP	FP	FN	*Accuracy (%)*	*Precision (%)*	*Recall (%)*	*F1-Score (%)*
A	5	0	0	100.00	100.00	100.00	100.00
B	3	2	0	98.46	60.00	100.00	75.00
C	5	0	0	100.00	100.00	100.00	100.00
D	5	0	0	100.00	100.00	100.00	100.00
E	4	0	0	100.00	100.00	100.00	100.00
F	3	2	0	98.46	60.00	100.00	75.00
G	4	0	0	100.00	100.00	100.00	100.00
H	5	0	0	100.00	100.00	100.00	100.00
I	5	0	0	100.00	100.00	100.00	100.00
J	5	0	0	100.00	100.00	100.00	100.00
K	5	0	0	100.00	100.00	100.00	100.00
L	5	0	0	100.00	100.00	100.00	100.00
M	5	0	0	100.00	100.00	100.00	100.00
N	3	2	0	98.46	60.00	100.00	75.00
O	5	0	0	100.00	100.00	100.00	100.00
P	5	0	0	100.00	100.00	100.00	100.00
Q	4	1	0	99.23	80.00	100.00	88.89
R	5	0	0	100.00	100.00	100.00	100.00
S	5	0	0	100.00	100.00	100.00	100.00
T	5	0	0	100.00	100.00	100.00	100.00
U	5	0	0	100.00	100.00	100.00	100.00
V	5	0	0	100.00	100.00	100.00	100.00
W	5	0	0	100.00	100.00	100.00	100.00
X	5	0	0	100.00	100.00	100.00	100.00
Y	4	1	0	99.23	80.00	100.00	88.89
Z	5	0	0	100.00	100.00	100.00	100.00

Tabel 3 menunjukkan bahwa sebanyak 21 huruf (A, C, D, E, G, H, I, J, K, L, M, O, P, R, S, T, U, V, W, X, Z) memiliki performa sangat baik dengan akurasi dan *precision* mencapai ≥90%, yang berarti sistem mampu mengenali pose huruf-huruf tersebut secara akurat tanpa kesalahan. Namun, terdapat 5 huruf yang mengalami kesalahan pengenalan, yaitu B, F, N, Q,

dan Y. Huruf Q dan Y masih tergolong baik dengan *precision* 80%, sementara B, F, dan N memiliki *precision* 60% sehingga masuk dalam kategori cukup. Kesalahan ini disebabkan oleh kemiripan bentuk pose yang membuat sistem keliru mengklasifikasikannya sebagai huruf lain. Secara umum, sistem telah menunjukkan performa yang baik, namun masih perlu peningkatan khusus pada beberapa huruf yang kurang akurat agar lebih andal dalam penggunaan nyata.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian dan pengujian yang telah dilakukan dapat disimpulkan bahwa Sistem Deteksi Pose *Semaphore* Berbasis *Deep Learning* Menggunakan *Metode Multi Layer Perceptron* (MLP) dan *MediaPipe* yang dikembangkan memiliki akurasi sebesar 93.84% dari 130 pose huruf yang diuji secara *real-time*, 122 pose dikenali dengan benar. Nilai akurasi sebesar 93,84% menunjukkan bahwa sebagian besar prediksi benar, dan nilai *recall* sebesar 100% menunjukkan bahwa semua huruf terdeteksi dengan tanpa ada yang terlewat. Kemampuan sistem untuk mengenali pose dan menghindari kesalahan klasifikasi seimbang dengan F1-score 96,80%. Hasil ini menunjukkan bahwa sistem bekerja dengan konsisten dan akurat secara *real-time*. Dengan kemampuan ini, sistem ini layak digunakan sebagai alat bantu untuk mengajar atau melatih pengenalan sandi *semaphore*. Ini terutama berlaku untuk pemula yang baru belajar pose *semaphore*. Saran yang dapat diterapkan untuk penelitian selanjutnya adalah:

Penambahan variasi data pelatihan, seperti pengambilan gambar dari berbagai sudut kamera, pencahayaan yang berbeda, serta postur tubuh yang lebih variasi untuk meningkatkan ketahanan sistem terhadap kondisi nyata. Pengembangan sistem dalam aplikasi android, web, dan desktop agar dapat digunakan secara langsung dan juga untuk tampilan yang lebih menarik dan interaktif. Karena dalam penelitian ini kata yang ditampilkan telah ditentukan pada saat pengembangan ada baiknya di tambahkan fitur untuk pengenalan kata atau kalimat secara langsung. Penambahan fitur untuk dua atau lebih pengguna sekaligus dalam satu frame untuk pose berkelompok.

DAFTAR REFERENSI

Alfarezi, M. (2024). Pengenalan kode semaphore berbasis Raspberry Pi menggunakan metode CNN. *E-Proceeding of Engineering, 11*(4), 4851–4858. https://doi.org/Diakses dari repository Universitas Telkom

Alfarizi, M. R. S., Al-farish, M. Z., Taufiqurrahman, M., Ardiansah, G., & Elgar, M. (2023). Penggunaan Python sebagai bahasa pemrograman untuk machine learning dan deep learning. *Karya Ilmiah Mahasiswa Bertauhid (KARIMAH TAUHID), 2*(1), 1–6.

Cai, S., Shu, Y., Chen, G., Ooi, B. C., Wang, W., & Zhang, M. (2019). Effective and efficient dropout for deep convolutional neural networks. *arXiv*, 1–12. https://doi.org/10.48550/arXiv.1904.03392

Dudekula, K. V., Chalapathi, M. M. V., Kumar, Y. V. P., Prakash, K. P., Reddy, C. P., Gangishetty, D., Solanki, M., & Singhu, R. (2024). Physiotherapy assistance for patients using human pose estimation with Raspberry Pi. *ASEAN Journal of Scientific and Technological Reports, 27*(4). https://doi.org/10.55164/ajstr.v27i4.251096

Dwika, A. S. P., Putra, A. S., & Alkadri, S. P. A. A. (2024). Identifikasi gerakan tangan pada sandi semaphore Pramuka secara realtime menggunakan decision tree. *Jurnal Education and Teknologi, 5*(2), 401–417.

Gulo, S. H., & Lubis, A. H. (2024). Penerapan multi-layer perceptron untuk mengklasifikasi penduduk kurang mampu. *Journal of Computer Science and Information Technology, 4*(2), 51–59. https://doi.org/10.47065/explorer.v4i2.1146

Handayanto, R. T., & Herlawati, H. (2020). Prediksi kelas jamak dengan deep learning berbasis graphics processing units. *Jurnal Kajian Ilmiah, 20*(1), 67–76. https://doi.org/10.31599/jki.v20i1.71

Mahersatillah Suradi, A., Alam, S., Rasyid, M. F., Djafar, I., Dipa Makassar, U., & Perintis Kemerdekaan, J. K. (2023). Sistem deteksi kantuk pengemudi mobil berdasarkan analisis rasio mata menggunakan computer vision. *JUKI: Jurnal Komputer dan Informatika, 2*, 222–230.

Mutianniza, S. Z., & Suwardoyo, U. (2023). Aplikasi kamera cerdas untuk deteksi kendaraan menggunakan library TensorFlow. *Jurnal Sintaks Logika, 3*(3), 61–68. https://doi.org/10.31850/jsilog.v3i3.2589

Ojiako, K., & Farrahi, K. (2023). MLPs are all you need for human activity recognition. *Applied Sciences (Switzerland), 13*(20). https://doi.org/10.3390/app132011154

Peryanto, A., Yudhana, A., & Umar, R. (2020). Rancang bangun klasifikasi citra dengan teknologi deep learning berbasis metode convolutional neural network. *FORMAT: Jurnal Ilmiah Teknik Informatika, 8*(2), 138. https://doi.org/10.22441/format.2019.v8.i2.007

Przybyła-Kasperek, M., & Marfo, K. F. (2024). A multi-layer perceptron neural network for varied conditional attributes in tabular dispersed data. *PLoS ONE, 19*(12). https://doi.org/10.1371/journal.pone.0311041

Putra, L. S. A., Wigyarinto, F. T. P., Kusumawardhani, E., & Gunawan, V. A. (2024). Semaphore letter code recognition system using wavelet method and back propagation neural network. *International Journal of Advanced Technology and Engineering Exploration, 11*(112), 316–331. https://doi.org/10.19101/IJATEE.2023.10102433

Ragil, K., Dyansyah, K., Purwantoro, S. D., & Ilmi, M. (2025). Penggunaan computer vision untuk estimasi pose squat sebagai solusi alternatif latihan kebugaran di gym. *Seminar Nasional Teknologi & Sains (STAINS), 4*, 199–207.

Resha, M., Syamsu, S., Andryanto, & Bakry. (2022). Prediksi penyebaran kasus tuberkulosis dengan metode artificial neural network dan multi-layer perceptron di Kota Makassar. *JNSTA ADPERTISI Journal*, *2*(1), 16–21.

Sarakon, S., Massagram, W., & Tamee, K. (2025). Multisource data fusion using MLP for human activity recognition. *Computers, Materials and Continua*, *82*(2), 2109–2136. https://doi.org/10.32604/cmc.2025.058906

Sejati, R. H. P., & Mardhiyyah, R. (2021). Deteksi wajah berbasis facial landmark. *Jurnal Teknologi Informasi*, *5*(2), 144–148.

Sitorus, H. (2022). Implementasi deep learning mendeteksi pengguna masker berbasis framework TensorFlow dengan metode convolutional neural network. *SENTRI: Jurnal Riset Ilmiah*, *1*(3), 862–874. https://doi.org/10.55681/sentri.v1i3.298

Wati, R., & Ernawati, S. (2021). Analisis sentimen persepsi publik mengenai PPKM pada Twitter berbasis SVM menggunakan Python. *Jurnal Teknik Informatika UNIKA Santo Thomas*, *6*, 240–247. https://doi.org/10.54367/jtiust.v6i2.1465