Deteksi Pose dan Penjumlahan Jari dari Gerakan Tangan

R. B. Wishnumurti and 13519203 (*Author*)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519203@std.stei.itb.ac.id

***Abstract*—Gerakan tangan manusia memiliki banyak karakterisitik. Setiap gerakan jari bisa memiliki banyakl arti, misalnya seperti *gestures peace atau juga dapat berartikan sebuah angka*. Dengan pemrosesan citra, kita bisa mendeteksi gerakan tersebut dari citra yang diinput untuk menjadi instruksi yang interaktif misalnya pada sebuah sistem rumah pintar. Pada makalah in akan dipakai sebuah model *landmark* tangan untuk mendeteksi input video secara langsung. Aplikasi ini dapat mendeteksi beberapa *gestures* yang umum dipakai dan juga melakukan penjumlahan dasar dari jari tangan.**

Keywords—pengenalan gerakan tangan; landmark detection,

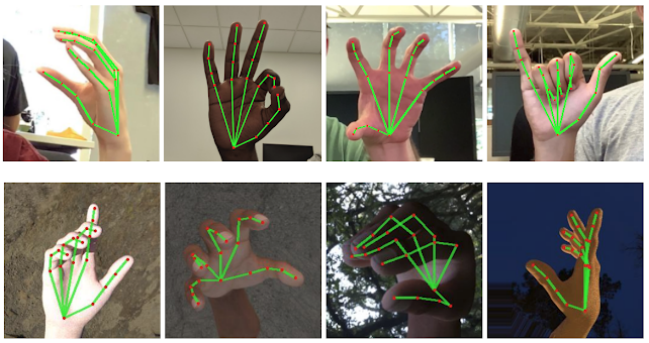
I. PENDAHULUAN

Gerakan tangan adalah gerakan atau posisi tangan dan jari-jari yang digunakan untuk mengomunikasikan pesan tertentu atau untuk menyampaikan makna tertentu. Gerakan tangan adalah sarana komunikasi yang alami dan intuitif yang dapat digunakan untuk berinteraksi dengan mesin dan perangkat, seperti smartphone, tablet, dan lingkungan realitas virtual.

Gerakan tangan dapat diklasifikasikan ke dalam kategori yang berbeda, seperti gerakan statis dan gerakan dinamis. Gestur statis adalah posisi tangan yang dipegang untuk jangka waktu tertentu, sedangkan gestur dinamis melibatkan gerakan atau perubahan posisi tangan dari waktu ke waktu. Gerakan tangan juga dapat diklasifikasikan berdasarkan jumlah jari yang digunakan, seperti gerakan satu jari, gerakan dua jari, dan gerakan banyak jari.

Pengenalan gerakan tangan adalah topik yang semakin penting dalam bidang visi komputer dan interaksi manusia-komputer. Dengan berkembangnya smartphone dan perangkat berbasis sentuhan lainnya, kemampuan untuk menginterpretasikan dan merespons gerakan tangan telah menjadi sangat penting untuk desain antarmuka yang ramah pengguna dan intuitif.

Teknologi deteksi gerakan tangan memiliki aplikasi yang luas, mulai dari interpretasi bahasa isyarat dan antarmuka VR game, dan perawatan kesehatan. Telah ada kemajuan yang signifikan dalam pengembangan algoritma dan teknik untuk pengenalan gerakan tangan dalam beberapa tahun terakhir. Hal ini disebabkan oleh ketersediaan dataset yang besar dan kemajuan teknik deep learning, sehingga erakan tangan dapat dikenali berdasarkan berbagai fitur, seperti bentuk, gerakan, dan penampilannya.



Gambar I.1 Contoh Aplikasi *Hand Landmark*

Pada makalah ini, penulis mengimplementasikan pengenalan gerakan tangan dengan model *pre-trained* mediapipe *hand landmark* berbasis *convolutional neural network untuk* mengidentifikasi tengara atau titik-titik kunci pada tangan. Titik-titik kunci ini bisa mencakup ujung jari, buku-buku jari, pergelangan tangan, dan fitur khas lainnya dari tangan sehingga nantinya bisa diteksi sesusai kebuthan pengembang,

II. LANDASAN TEORI

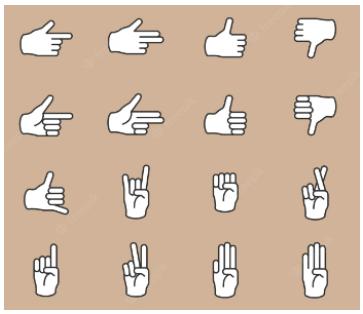
A. Gerakan Tangan

Gerakan tangan adalah bentuk umum dari komunikasi nonverbal yang dapat menyampaikan berbagai macam makna dan pesan. Gerakan tangan dapat digunakan untuk mengekspresikan emosi, menyampaikan penekanan atau makna, dan bahkan dapat digunakan untuk menyampaikan konsep yang lebih abstrak. Gerakan tangan sering digunakan bersamaan dengan bahasa lisan, dan gerakan tangan dapat membantu meningkatkan efektivitas komunikasi.

Beberapa penelitian telah berfokus pada penggunaan gerakan tangan dalam konteks budaya yang berbeda, sementara penelitian lain telah meneliti cara gerakan tangan digunakan dalam berbagai jenis komunikasi, seperti berbicara di depan umum atau interaksi interpersonal.

Gerakan tangan sangat menarik karena dapat mengirimkan makna ketika tidak disertai dengan kata-kata lisan. Misalnya, seseorang bisa membuat gerakan tangan untuk menyampaikan bahwa mereka ingin segala sesuatunya berhenti, meskipun mereka tidak berbicara pada saat itu. Hal ini menunjukkan

bahwa gerakan tangan mungkin memiliki makna global yang dipahami oleh individu dari semua budaya.



Gambar II.1 Contoh Gerakan Tangan

Gerakan tangan juga merupakan bagian penting dari banyak bahasa isyarat, yaitu bahasa yang menggunakan gerakan tangan dan bentuk komunikasi nonverbal lainnya untuk menyampaikan makna. Bahasa isyarat digunakan oleh orang-orang yang tuli atau sulit mendengar, dan bahasa-bahasa ini adalah bahasa yang lengkap dengan tata bahasa dan sintaksisnya yang unik.

B. Convolutional Neural Network

Gambar II.2 Arsitektur CNN

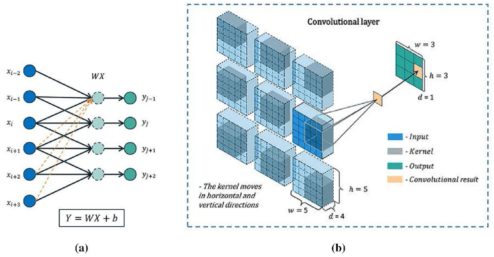
Convolutional neural network adalah jenis *artificial neural network* tiruan yang dirancang khusus untuk klasifikasi gambar dan tugas pemrosesan. Secara umum CNN tidak jauh berbeda dengan neural network umumnya, CNN setidaknya memiliki lapisan konvolusi yang dapat digunakan untuk m engurangi jumlah unit dalam jaringan, sehingga lebih sedikit parameter untuk dipelajari dan mengurangi kemungkinan overfitting. Lapisan CNN disusun dalam volume tiga dimensi yang berupa lebar, tinggi, dan kedalaman (di mana kedalaman mengacu pada dimensi ketiga dari volume, seperti jumlah saluran dalam gambar atau jumlah filter di dalam *layer*). CNN sangat berguna untuk menemukan pola dalam data dengan fitur yang sangat besar seperti dalam deteksi gambar.

Berikut akan dijelaskan setiap bagian dari arsitektur *Convolutional Neural Network :*

1. Input Layer

Lapisan ini menerima data input, yang bisa berupa gambar, video, atau urutan kata yang umumnya diwakilkan dalam bentuk matriks.

2. *Convolutional Layer*



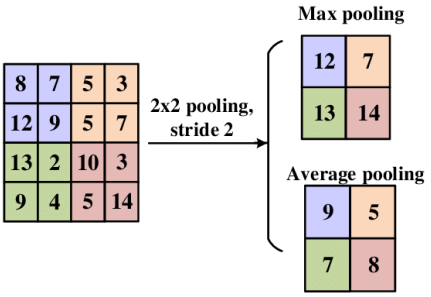
Gambar II.3 *Convolutional Layer*

Konvolusi adalah operasi matematika, di mana suatu fungsi "diterapkan" dengan cara tertentu ke fungsi lain dan hasilnya berupa sebagai "campuran" dari kedua fungsi tersebut. Konvolusi diwakili oleh tanda bintang (*), yang mungkin membingungkan dengan operator * yang umumnya digunakan untuk perkalian dalam banyak bahasa pemrograman.

Konvolusi sangat bagus dalam mendeteksi struktur sederhana dalam sebuah gambar, dan kemudian menyatukan fitur-fitur sederhana tersebut untuk membangun fitur yang lebih kompleks. Dalam jaringan konvolusi, proses ini terjadi pada serangkaian banyak lapisan, yang masing-masing melakukan konvolusi pada output dari lapisan sebelumnya.

Lapisan ini memproses data yang masuk melalui kumpulan filter yang dapat dipelajari, menghasilkan sekumpulan peta fitur. Setiap filter adalah matriks kecil yang meluncur di atas data input, mengeksekusi perkalian dan penjumlahan elemen untuk membangun peta fitur baru. Lapisan konvolusional dapat belajar mengenali karakteristik yang berbeda pada skala spasial yang berbeda dengan menerapkan beberapa filter ke data input.

3. *Pooling Layer*



Gambar II.4 *Pooling Layer*

Pooling layer, juga dikenal sebagai *downsampling*, melakukan reduksi dimensionalitas, mengurangi jumlah parameter dalam input. Mirip dengan lapisan konvolusi, operasi pooling menyapu filter di seluruh input, tetapi perbedaannya adalah bahwa filter ini tidak memiliki bobot apa pun. Sebagai gantinya, kernel menerapkan fungsi agregasi ke nilai-nilai dalam bidang reseptif yang akan mengisi array output. Terdapat dua jenis pooling, yaitu:

- *Max Pooling*

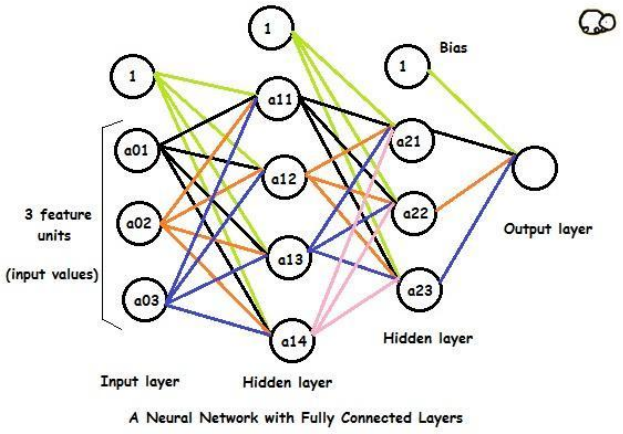
  Saat filter bergerak melintasi input, filter memilih piksel dengan nilai maksimum untuk dikirim ke array output. Sebagai tambahan, pendekatan ini cenderung lebih sering digunakan dibandingkan dengan average pooling.

- *Average Pooling*

  Sewaktu filter bergerak melintasi input, filter menghitung nilai rata-rata dalam bidang reseptif untuk dikirim ke array output.

Meskipun banyak informasi yang hilang dalam lapisan penyatuan, ini juga memiliki sejumlah manfaat bagi CNN. Mereka membantu mengurangi kompleksitas, meningkatkan efisiensi, dan mengurangi risiko overfitting.

4. *Fully Connected Layer*

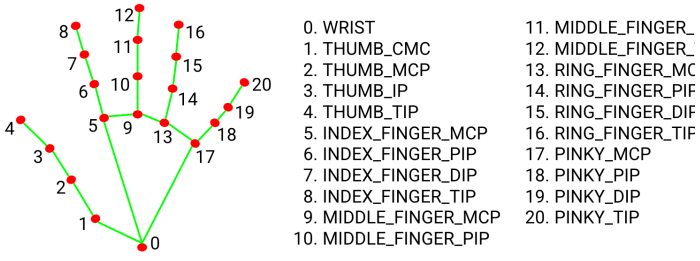


Gambar II.5 Fully Connected Layer

Pada lapisan ini, nilai piksel dari gambar input tidak secara langsung terhubung ke lapisan output. Namun, pada lapisan yang *fully connected layer*, setiap node pada lapisan output terhubung ke node pada lapisan sebelumnya secara langsung. Lapisan ini melakukan tugas kategorisasi berdasarkan karakteristik yang dikumpulkan oleh lapisan sebelumnya dan filternya yang bervariasi. Sementara convolutional dan pooling layer sering menggunakan fungsi ReLu untuk mengkategorikan input, *fully connected layer* biasanya menggunakan fungsi aktivasi softmax untuk memberikan probabilitas mulai dari 0 sampai 1.

Setelah melewati seluruh layer tadi, *output layer* akan menghasilkan prediksi akhir dari CNN, berdasarkan distribusi probabilitas yang dihasilkan oleh lapisan yang terhubung sepenuhnya. Beberapa arsitektur CNN jugahand l termasuk lapisan tambahan, seperti lapisan dropout untuk mencegah overfitting dan lapisan normalisasi batch untuk mempercepat pelatihan. Selain itu, beberapa CNN menggunakan koneksi lompatan atau blok residu untuk memungkinkan jaringan untuk mempelajari hubungan yang lebih kompleks antara input dan output.

C. Hand Landmarking



Gambar II.6 Hand Landmark

Hand landmark adalah titik atau fitur spesifik pada tangan yang bisa digunakan sebagai referensi dalam berbagai aplikasi, seperti computer vision, robotika, atau biometrik. *Hand landmark* mencakup ujung jari, buku-buku jari, pangkal ibu jari, atau bagian tengah telapak tangan yang nantinya digunakan untuk melacak posisi dan orientasi tangan di ruang dan untuk mengidentifikasi gerakan atau gerakan tertentu.

Data *ground truth* untuk model ini terdiri dari ~30 ribu gambar dunia nyata yang dianotasi secara manual dengan 21 koordinat 3D dan dengan nilai Z dari peta kedalaman gambar, jika ada per koordinat yang sesuai. Untuk lebih mencakup kemungkinan pose tangan dan memberikan supervisi tambahan pada sifat geometri tangan, mereka juga membuat model tangan sintetis berkualitas tinggi di atas berbagai latar belakang dan memetakannya ke koordinat 3D yang sesuai.

III. IMPLEMENTASI

A. Implementasi Program

Pengimplemntasian dilakukan pada program python dengan memanfaatkan kakas opencv dan mediapipe. Program menggunakan model *hand landmark* yang berbasis *convolutional neural network.* Berikut komponen dalam kode program:

1. Menginisialisasi model *hand lanndmark* dan kakas opencv

Kode di bawah mengintruksikan pengaturan untuk model *hand landmark* yang dipakai serta pengaturan dari window perekaman video webcam dari opencv. Juga dilakukan set citra menjjadi tidak *writable* untuk meningkatkan performansi.

```
import cv2
import mediapipe as mp
mp_drawing =
mp.solutions.drawing_utils
mp_drawing_styles =
mp.solutions.drawing_styles
mp_hands = mp.solutions.hands

# For webcam input:
cap = cv2.VideoCapture(0)
with mp_hands.Hands(
    model_complexity=0,
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5) as
hands:
  while cap.isOpened():
    success, image = cap.read()
    if not success:
```

```
      print("Ignoring empty camera
frame.")
      # If loading a video, use
'break' instead of 'continue'.
      continue

    # To improve performance,
optionally mark the image as not
writeable to
    # pass by reference.
    image.flags.writeable = False
    image = cv2.cvtColor(image,
cv2.COLOR_BGR2RGB)
    results = hands.process(image)

    # Draw the hand annotations on the
image.
    image.flags.writeable = True
    image = cv2.cvtColor(image,
cv2.COLOR_RGB2BGR)
```

2. Pengaturan *hand landmark*

Pada kode di bawah dilakukan set nilai inisial untuk jumlah jari dan juga indeks dari setiap jenis jari. Dilakukan juga pengecekan apakah tangan kanan atau kiri dari indeks tangan dan mengisi variabel list dengan posisi x dan y dari setiap *landmark.*

```
    # Initially set finger count to 0
for each cap
    fingerCount = 0

    #set index for each type of
fingers
    idx_0 = 0
    idx_1 = 0
    idx_2 = 0
    idx_3 = 0
    idx_4 = 0
    finger = ""
```

```
    if results.multi_hand_landmarks:

      for hand_landmarks in
results.multi_hand_landmarks:
        # Get hand index to check
label (left or right)
        handIndex =
results.multi_hand_landmarks.index(han
d_landmarks)
        handLabel =
results.multi_handedness[handIndex].cl
assification[0].label

       # Atur variabel untuk menjaga
posisi landmark (x dan y)
        handLandmarks = []

        # Isi daftar dengan posisi x
dan y dari setiap landmark
        for landmarks in
hand_landmarks.landmark:

handLandmarks.append([landmarks.x,
landmarks.y])
```

3. Set nilai dari setiap jari

Dari model *hand landmark*, akan dilakukan set nilai apabila cocok dengan input yang ditangkap dari video webcam. Setiap model program mendeteksi adanya jari yang diangkat maka fingerCount akan bertambah satu dan begitu pula sebaliknya. Lalu jenis jari yang teridentifikasi akan diset nilai indeksnya (jempol = 0, telunjuk =1, tengah =2, manis = 3, kelingking =4). Setelah melakukan set untuk setiap jari yang tertangkap pada kamera, maka akan dilakkan deteksi berdasarkan atribut yang dimiliki sesuai aturan yang ada pada kode selanjutnya.

```
       # Uji kondisi untuk setiap
jari: Hitungan meningkat jika jari
dianggap terangkat.
        # Jempol: Posisi TIP x harus
```

```
lebih besar atau lebih rendah dari
posisi IP x,
        # deppeding pada label tangan.
        if handLabel == "Left" and
handLandmarks[4][0] >
handLandmarks[3][0]:
          fingerCount = fingerCount+1
          idx_0 = 1
        elif handLabel == "Right" and
handLandmarks[4][0] <
handLandmarks[3][0]:
          fingerCount = fingerCount+1
          idx_0 = 1

        # Jari-jari lainnya: Posisi
TIP y harus lebih rendah dari posisi
PIP y,
        # karena asal gambar berada di
sudut kiri atas.
        if handLandmarks[8][1] <
handLandmarks[6][1]:       #Index
finger
          fingerCount = fingerCount+1
          idx_1 = 1
        if handLandmarks[12][1] <
handLandmarks[10][1]:     #Middle
finger
          fingerCount = fingerCount+1
          idx_2 = 1
        if handLandmarks[16][1] <
handLandmarks[14][1]:     #Ring finger
          fingerCount = fingerCount+1
          idx_3 = 1
        if handLandmarks[20][1] <
handLandmarks[18][1]:     #Pinky
          fingerCount = fingerCount+1
          idx_4 = 1
```

4. *Hand Pose/Gestures Detection*

Berdasarkan atribut yang dimiliki, maka program akan mendeteksi apakah input masuk ke dalam kategori jenis pose yang telah ditentukan. Conothnya adalah pose *peace* yang terdiri dari jari indeks 1 (telunjuk) dan indeks 2 (jari tengah) dan seterusnya.

```
 #Hand Gestures/Pose Detection
        if idx_1 and idx_2 and
fingerCount==2:
            finger = "peace"
        if idx_0 and fingerCount==1:
            finger = "thumbs up"
        if idx_0 and idx_4 and
fingerCount==2:
            finger = "call"
        if idx_0 and idx_1 and
fingerCount==2:
            finger = "L"
        if idx_0 and idx_1 and idx_4
and fingerCount == 3:
            finger = "metal/rock"
        if idx_1 and fingerCount==1:
            finger = "point"
        if idx_4 and fingerCount==1:
            finger = "promise"
        if idx_0 and idx_1 and idx_2
and idx_3 and idx_4:
            finger = "wave"
```

5. Penggambaran *landmark* dan output display ke layar

Pada bagian ini akan melakukan penggamabaran *landmark* diatas tangan dan juga melakukan output display ke layar. Informasi yang ditampilkan adalah jumlah jari dan juga jenis pose tangan.

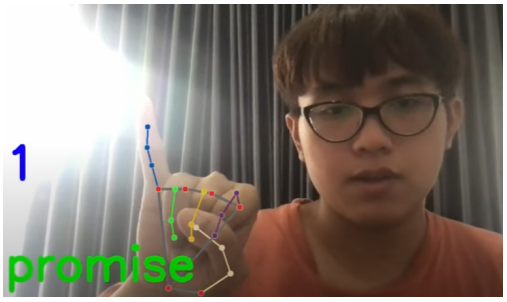
```
        # Gambar Hand Landmarks
        mp_drawing.draw_landmarks(
            image,
            hand_landmarks,
            mp_hands.HAND_CONNECTIONS,

mp_drawing_styles.get_default_hand_lan
dmarks_style(),
```

```
mp_drawing_styles.get_default_hand_con
nections_style())

    # Display finger count & type
    cv2.putText(image,
str(fingerCount), (50, 450),
cv2.FONT_HERSHEY_SIMPLEX, 3, (255, 0,
0), 10)
    cv2.putText(image, finger, (50,
650), cv2.FONT_HERSHEY_SIMPLEX, 3, (0,
255, 0), 10)

    # Display image
    cv2.imshow('MediaPipe Hands',
image)
    if cv2.waitKey(5) & 0xFF == 27:
      break
cap.release()
```

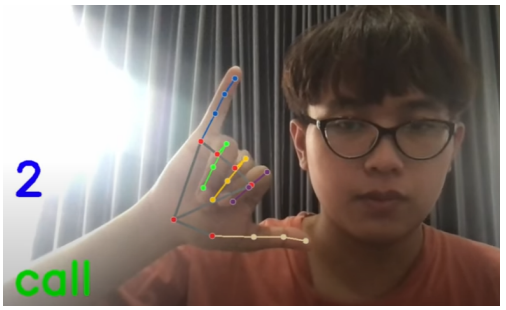
B. Hasil

Berikut beberapa contoh citra hasil dari proses deteksi tangan.

Gambar III.1 Deteksi pose peace



Gambar III.2 Deteksi pose promise



Gambar III.3 Deteksi pose call

Gambar III.4 Deteksi pose metal/rock

Gambar III.5 Deteksi pose wave dengan dua jari

Bisa dilihat dilihat dari hasil citra di atas bahwa program berhasil menentukan pose dari setiap gerakan tangan yang dilakukan oleh orang yang berada di input video. Program juga dapat mendeteksi jika terdapat dua tangan seperti yang tertera pada Gambar III.5.

IV. Kesimpulan

Pemanfaatan model *hand landmark* berbasis *convolutional neural network* berhasil mengidentifikasi input video dari gerakan tangan. Bidang ini memiliki banyak pengaplikasian yang sangat potensial, penulis berharap penulisan makalah ini dapat menjadi pembelajaran awal untuk mengembangkan aplikasi deteksi gerakan tangan yang lebih baik lagi.

Video Link Youtube

https://youtu.be/M4H-JkxvzuE

Ucapan Terima Kasih

Penulis mengucapkan terima kasih yang sebesar-besarnya kepada Tuhan yang Maha Esa karena dengan bantuannya

penulis bisa menyelesaikan makalah ini dengan tepat waktu. Penulis juga ingin mengucapkan terima kasih sebesar-besarnya kepada Bapak Rinaldi Munir dan asisten dosen IF4073 Interpretasi dan Pengolahan Citra yang telah mengajarkan konsep-konsep strategi algoritma sehingga penulis bisa membuat makalah berjudul "Deteksi Pose dan Penjumlahan Jari dari Gerakan Tangan".

References

[1] Munir, R. "Pengantar Interpretasi dan Pengolahan Citra (Bagian I),". 2022.Retrieved December 19, 2022, from https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2022-2023/01-Pengantar-Pengolahan-Citra-Bag1-2022.pdf

[2] Munir, R. "Pembentukan Citra dan Digitalisasi Citra,". 2022. Retrieved December 19, 2022, from https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2022-2023/03-Pembentukan-Citra-dan-Digitalisasi-Citra-2022.pdf

[3] [Munir, R. "24-Convolutional Neural Network,". 2022. Retrieved December 19, 2022, from https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2022-2023/24-CNN

[4] IBM, Convolutional Neural Networks,". 2022. Retrieved December 19, 2022, from What are Convolutional Neural Networks? | IBM

[5] MediaPipe, MediaPipe Hands, ,". 2022. Retrieved December 19, 2022, from Hands | mediapipe (google.github.io)

[6] MediaPipe Hands Model Card, MediaPipe Hands, ,". 2022. Retrieved December 19, 2022, from Model Card Hand Tracking (Lite_Full) with Fairness Oct 2021.pdf - Google Drive

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Desember 2022

R. B. Wishnumurti 13519203