TUGAS BESAR PENGOLAHAN CITRA DIGITAL

Disusun untuk memenuhi penugasan mata kuliah Pengolahan Citra Digital Praktikum

Disusun oleh:

Azka Darajat	231511036
Hanif Ahmad Naufal	231511048
Radja Restu Arsita	231511061

PROGRAM STUDI DIII TEKNIK INFORMATIKA

JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA

POLITEKNIK NEGERI BANDUNG

2025

Daftar Isi

1. Pendahuluan

Deteksi gerakan yang kami lakukan bertujuan untuk mengklasifikasikan tiga jenisgerakan tinju utama, yaitu jab, hook, dan uppercut. Ketiga gerakan ini dipilih karena merupakan elemen dasar dalam teknik boxing yang memiliki pola pergerakan tubuh yang khas dan dapat dikenali melalui analisis pose. Proses klasifikasi ini dirancang sebagai fondasi utama untuk menjadi input interaktif dalam sebuah game boxing yang sedang kami kembangkan. Dengan mengenali gerakan pemain secara real-time menggunakan kamera dan memprosesnya melalui model klasifikasi pose, sistem dapat merespons aksi pemain secara akurat, sehingga menciptakan pengalaman bermain yang lebih imersif dan responsif. Pendekatan ini memungkinkan pemain untuk mengontrol permainan melalui gerakan tubuh alami.

2. Studi Pustaka dan Referensi

2.1. MediaPipe Holistic

MediaPipe Holistic adalah sebuah solusi komprehensif berbasis AI dari Google yang menggabungkan pelacakan wajah, tubuh, dan tangan secara real-time dalam satu pipeline terpadu. Dengan menggunakan pendekatan *multi-modal*, MediaPipe Holistic dapat mengintegrasikan **Face Mesh**, **Pose Landmark**, dan **Hand Landmark** secara bersamaan untuk mendeteksi dan melacak berbagai titik-titik penting (landmark) pada tubuh manusia.

Fitur utama MediaPipe Holistic:

- Menghasilkan hingga **543 landmark**: 468 pada wajah, 33 pada tubuh, dan 21 pada masing-masing tangan.
- *Performance-optimized*: Menggunakan Graph-based API yang memungkinkan efisiensi tinggi dalam pengolahan real-time di perangkat lokal (CPU/GPU).
- Cocok untuk aplikasi kompleks seperti analisis postur tubuh, gesture detection, dan augmented reality (AR).

2.2. MediaPipe Pose

MediaPipe Pose adalah solusi khusus untuk pelacakan pose tubuh manusia secara keseluruhan dengan 33 landmark 3D. Solusi ini dirancang untuk memahami posisi bagian tubuh seperti kepala, leher, bahu, siku, pinggul, lutut, dan pergelangan kaki dalam ruang 3D dari input video atau gambar.

Arsitektur MediaPipe Pose:

- **Stage 1 – BlazePose Detector**: Menemukan *Region of Interest* (ROI) yaitu keberadaan tubuh manusia dari frame input.
- **Stage 2 – Landmark Model**: Mengestimasi posisi 3D dari 33 titik tubuh utama dengan akurasi tinggi.
- Pose model dibangun menggunakan neural network ringan (seperti *MobileNetV3*), cocok untuk real-time di perangkat edge.

Fitur utama MediaPipe Pose:

- Akurasi tinggi dalam pelacakan full-body pose.
- Mendukung estimasi 3D (x, y, z) dari tiap titik.
- Sangat cocok untuk aplikasi seperti pelatihan olahraga, game interaktif, dan klasifikasi pose seperti dalam sistem klasifikasi pose tinju ini.
- Mendeteksi landmark penting seperti **shoulder**, **elbow**, **wrist**, dan **hip** yang digunakan dalam sistem ini untuk mendeteksi gerakan tinju.

2.3. MediaPipe Face

MediaPipe Face Mesh merupakan solusi untuk pelacakan wajah yang menghasilkan 468 titik landmark 3D dengan akurasi tinggi. Dapat digunakan untuk:

- Pelacakan ekspresi wajah
- AR filter (misalnya masker wajah virtual)
- Estimasi pose kepala (head pose estimation)

Fitur utama MediaPipe Face:

- Landmark halus untuk seluruh wajah termasuk kontur bibir, mata, alis, dan hidung.
- Real-time tracking yang efisien.

2.4. **Fokus dalam Proyek Ini – MediaPipe Poe**

Pada sistem klasifikasi pose tinju ini, MediaPipe Pose digunakan secara utama karena:

- Diperlukan pelacakan bagian tubuh atas (upper body), khususnya **bahu**, **siku**, dan **pergelangan tangan**.
- MediaPipe Pose dapat mengestimasi koordinat dari keenam landmark penting ini: LEFT_SHOULDER, RIGHT_SHOULDER, LEFT_ELBOW, RIGHT_ELBOW, LEFT_WRIST, RIGHT_WRIST.
- Data landmark ini selanjutnya digunakan sebagai fitur dalam model klasifikasi gerakan tinju, seperti jab, hook, dan uppercut.

2.5. Sumber Referensi yang Digunakan

Perkuliahan Pengolahan Citra Digital Praktik Jurusan Teknik Komputer dan Informatika Program Studi D3-Teknik Informatika Politeknik Negeri Bandung Tugas Mediapipe tanggal 20 Mei.

3. Kebutuhan Sistem

3.1. Hardware

- Kamera (webcam)

3.2. Requirements

Dalam pengembangan game ini memerlukan sejumlah library pendukung yang digunakan untuk memenuhi kebutuhan pengembangan mulai dari pengambilan dataset, identifikasi pose tubuh, serta pembuatan interface frontend dan menjalankan backend. Berikut adalah daftar pustaka dan tools yang digunakan:

- **Python 3.10.0**

  Python digunakan sebagai bahasa pemrograman utama. Versi 3.10.0 memberikan stabilitas yang diperlukan untuk kompatibilitas dengan pustaka-pustaka terbaru.
- **Visual Studio Code**

  Visual Studio Code digunakan sebagai editor utama karena ringan dan sudah familiar serta sering kami gunakan untuk coding.
- **OpenCV**

  OpenCV merupakan libarary utama untuk pengolahan citra dan video. OpenCV digunakan untuk menangkap citra dari webcam, serta menampilkan output video yang telah dianotasi dengan hasil pelacakan pose. OpenCV juga mendukung berbagai fungsi dasar seperti thresholding, drawing, dan transformasi citra.
- **MediaPipe**

  Merupakan library inti yang digunakan untuk pengembangan game ini. Library ini mampu melacak gerakan tubuh seperti tangan yang akan diperlukan untuk pengembangan game ini. Dalam game ini, MediaPipe digunakan untuk mendeteksi pose tubuh pemain dan mengidentifikasi koordinat titik-titik penting (keypoints) seperti bahu, siku, dan tangan. Informasi ini menjadi dasar dalam mendeteksi gerakan tinju yang dilakukan pemain.
- **NumPy**

Library penting untuk melakukan perhitungan matematika. Ini digunakan untuk merepresentasikan citra dalam bentuk array atau matriks serta melakukan berbagai operasi matematika terhadap data citra. Dalam konteks game ini yaitu pelacakan gerakan, NumPy digunakan untuk menghitung jarak antar titik landmark tubuh.

- **Scikit-Learn**

  Scikit-Learn digunakan untuk mengklasifikasikan gerakan. Library ini bisa mempermudah dalam proses pelatihan dan pengujian model klasifikasi untuk membedakan jenis gerakan boxing. Ini juga digunakan untuk menyediakan fungsi preprocessing seperti normalisasi data dan pembentukan fitur.

- **Matplotlib**

  Digunakan untuk keperluan visualisasi data seperti menampilkan grafik, hasil klasifikasi, ataupun dalam kasus ini untuk evaluasi performa model.

- **Pillow**

  Digunakan untuk menyimpan hasil tangkapan layar, konversi format gambar, cropping, dan operasi dasar lainnya. Dalam game ini, Pillow digunakan untuk menyimpan foto gerakan.

- **FastAPI**

  Untuk pengembangan backend, disini menggunakan framework FastAPI. FastAPI memungkinkan pengembangan backend menggunakan REST API secara cepat dan efisien menggunakan Python. Backend ini bertugas untuk menerima data dari client atau frontend, memproses gerakan, dan memberikan response secara real-time.

- **Uvicorn**

  Agar FastAPI dapat berjalan dengan baik, digunakan Uvicorn sebagai ASGI server. Uvicorn menjalankan aplikasi backend dan menangani komunikasi HTTP dengan client. Performa Uvicorn yang tinggi dan penggunaan resource yang rendah menjadikannya ideal untuk aplikasi berbasis real-time.

- **Jinja2**

  Untuk menghasilkan tampilan interface berbasis web yang dinamis, digunakan **Jinja2** sebagai template engine. Jinja2 memungkinkan penyisipan data dari backend ke dalam HTML secara efisien, untuk menampilkan hasil.

- **python-multipart**

Library **python-multipart** digunakan dalam pengelolaan form data yang mengandung file, khususnya jika aplikasi mendukung pengunggahan gambar atau video sebagai input untuk pelatihan atau evaluasi sistem.

3.3. Dataset

Pengumpulan dataset kami lakukan dengan membuat satu file khusus untuk mendapatkan dataset. Dengan cara menggunakan kamera webcam dengan menggunakan interface yang dibuat menggunakan library Tkinter. Game ini juga memanfaatkan MediaPipe Pose untuk mendeteksi dan mengekstrak koordinat landmark tubuh yang relevan dengan gerakan tinju yang sudah dijelaskan sebelumnya.

Langkah-langkah Pengumpulan Data:

1. Persiapan Folder Penyimpanan:

   Sebelum proses pengumpulan dimulai, program secara otomatis membuat folder bernama boxing_poses yang digunakan untuk menyimpan gambar dan data landmark hasil tangkapan. Selain itu, file JSON (boxing_poses.json) digunakan untuk menyimpan metadata dari setiap pose yang berhasil dikumpulkan, termasuk nama pose, koordinat landmark, dan path gambar.

2. Tampilan Antarmuka Pengguna:

   Aplikasi menampilkan antarmuka sederhana yang terdiri dari:

   - Tampilan video real-time dari webcam.
   - Dropdown untuk memilih jenis gerakan tinju yang akan direkam (JAB, HOOK, UPPERCUT).
   - Tombol “Capture Pose” untuk menyimpan satu pose.
   - Tombol “Capture 10x” untuk menangkap 10 pose sekaligus secara otomatis.

3. Pelacakan Pose dengan MediaPipe:

   Sistem menggunakan MediaPipe Pose untuk mendeteksi tubuh manusia secara real-time. Namun, hanya bagian tubuh atas yang digunakan, yaitu:

   - Bahu kiri dan kanan (LEFT_SHOULDER, RIGHT_SHOULDER)
   - Siku kiri dan kanan (LEFT_ELBOW, RIGHT_ELBOW)
   - Pergelangan tangan kiri dan kanan (LEFT_WRIST, RIGHT_WRIST)

   Koordinat 3D dari keenam titik tersebut (x, y, z) akan diambil jika pose berhasil terdeteksi pada frame.

4. Penyimpanan Data Pose:

   Ketika pengguna menekan tombol *Capture Pose*:

   - Sistem menyimpan gambar yang sedang ditampilkan (berisi posisi tubuh pengguna).
   - Koordinat landmark tubuh bagian atas disimpan ke dalam format JSON, dikaitkan dengan nama gerakan yang dipilih.
   - Semua informasi ini ditambahkan ke boxing_poses.json sebagai satu entri data.

5. Pengambilan Data (10 Sampel Sekaligus):

   Pengguna juga dapat menekan tombol *Capture 10x* untuk secara otomatis merekam sepuluh pose dari gerakan yang sama. Sistem akan mencoba mendeteksi pose pada tiap iterasi:

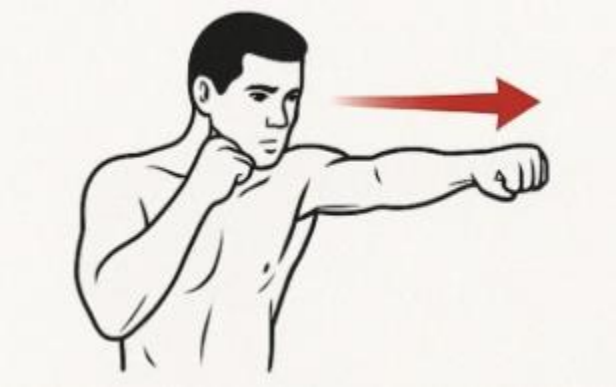
   - Jika pose tidak terdeteksi, akan dilakukan retry beberapa kali sebelum dilanjutkan ke sampel berikutnya.
   - Hal ini membantu memastikan kualitas data yang dikumpulkan dan menghindari penyimpanan data yang tidak valid

4. Langkah-langkah Pengembangan

4.1. Eksplorasi dan Preprocessing Data

Setelah dilakukan analisis mengenai kebutuhan telah kami rancang dataset yang akan kami kumpulkan yaitu dataset berupa gambar untuk 3 jenis pose boxing yaitu :

1. *Jab*



2. *Hook*

3. *Uppercut*

Setiap gambar yang diambil akan di ekstrasi landmark dari kedua lengannya dan nilai nya akan disimpan sebagai dataset untuk melakukan trainning

4.2. Pemilihan Arsitektur

Model yang digunakan dalam proyek ini adalah MLPClassifier (Multilayer Perceptron) dari pustaka *scikit-learn*, dan model ini sangat cocok digunakan karena karakteristik data dan tujuan sistem yang spesifik. Pertama, data input yang digunakan berupa 18 fitur numerik hasil ekstraksi dari 6 titik landmark tubuh bagian atas, masing-masing memiliki koordinat x, y, dan z. Bentuk data yang sangat terstruktur dan berdimensi rendah seperti ini memang ideal untuk diproses oleh model jaringan syaraf tiruan sederhana seperti MLP, tanpa perlu menggunakan model kompleks seperti CNN atau LSTM yang lebih cocok untuk data gambar atau sekuens.

Selain itu, tugas klasifikasi yang dihadapi oleh sistem ini tergolong sederhana, yaitu hanya membedakan tiga jenis pose tinju seperti yang sudah disebutkan diatas. Dengan jumlah kelas yang sedikit dan data yang cukup bersih karena berasal dari MediaPipe, MLP mampu mempelajari pola dari fitur dengan sangat baik tanpa membutuhkan arsitektur yang dalam atau waktu pelatihan yang lama.

Keunggulan lain dari penggunaan MLP adalah kecepatan dan kemudahannya dalam di-deploy. Model ini dapat dilatih dalam waktu singkat, disimpan menggunakan joblib, dan dimuat kembali dengan cepat di dalam backend FastAPI. Ini sangat sesuai untuk sistem berbasis real-time seperti game, yang membutuhkan prediksi instan untuk menjaga responsivitas dan pengalaman pengguna. Kesederhanaan dan efisiensi MLP

membuatnya sangat cocok dengan arsitektur sistem secara keseluruhan, dari proses ekstraksi pose hingga klasifikasi gerakan dalam game.

4.3. Pelatihan Model

Dalam proyek ini, model klasifikasi pose menggunakan MLPClassifier dari framework scikit-learn, yang merupakan salah satu pustaka machine learning paling populer dan mudah digunakan di Python untuk model berbasis data terstruktur. Pemilihan scikit-learn sebagai framework didasarkan pada kesederhanaan implementasi, efisiensi pelatihan, dan kemudahan integrasi dengan pipeline aplikasi yang lebih luas, khususnya dalam konteks proyek ini yang membutuhkan sistem prediksi ringan dan cepat untuk aplikasi real-time.

Parameter pelatihan yang digunakan pada model MLPClassifier mencakup max_iter=1000, yang setara dengan jumlah maksimum epoch dalam pelatihan. Nilai 1000 dipilih untuk memberi cukup kesempatan bagi model untuk mencapai konvergensi, terutama karena data pose yang digunakan memiliki dimensi rendah (18 fitur) dan modelnya tidak terlalu kompleks. Meskipun 1000 tampak besar, pelatihan akan berhenti lebih awal jika konvergensi sudah tercapai. Dengan demikian, nilai ini bersifat preventif agar model memiliki ruang cukup untuk belajar tanpa memaksakan pelatihan terlalu pendek.

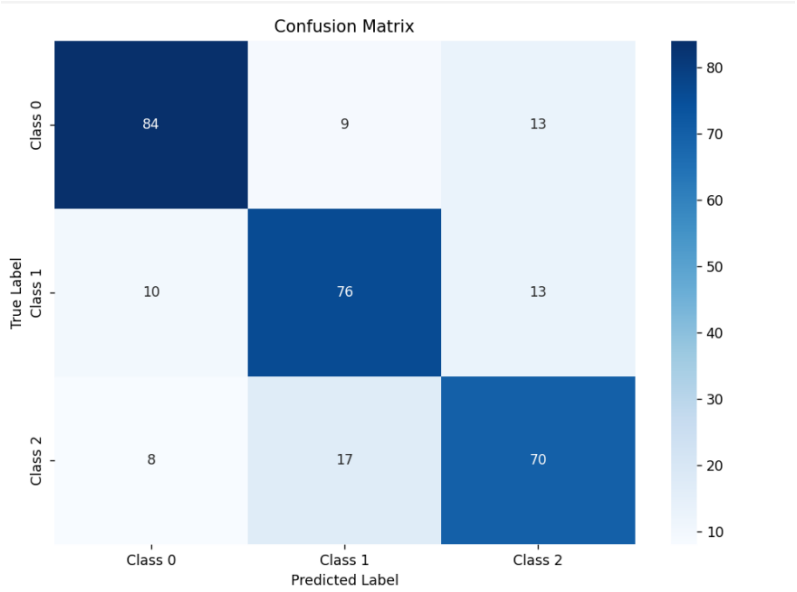
Selain itu, model menggunakan learning rate sebesar 0.001, yang merupakan nilai default pada MLPClassifier. Nilai ini sudah umum digunakan dalam banyak arsitektur neural network karena menawarkan keseimbangan antara kecepatan pelatihan dan stabilitas. Learning rate ini cukup kecil untuk menghindari lonjakan atau ketidakteraturan dalam proses pembaruan bobot, namun tetap cukup besar agar proses pelatihan tidak terlalu lambat.

Untuk proses optimisasi, digunakan optimizer Adam (Adaptive Moment Estimation), yang juga merupakan default dari MLPClassifier. Adam menggabungkan kelebihan dari momentum dan RMSProp, sehingga mampu menyesuaikan laju pembelajaran untuk masing-masing parameter secara dinamis. Ini sangat efektif untuk pelatihan pada dataset kecil hingga menengah seperti pada proyek ini, dan dapat mempercepat konvergensi tanpa terlalu banyak penyesuaian parameter.

Dengan demikian, penggunaan scikit-learn sebagai framework, learning rate 0.001, optimizer Adam, dan maksimal 1000 epoch memberikan kombinasi yang tepat antara kesederhanaan, performa, dan stabilitas dalam membangun sistem klasifikasi pose yang efisien dan akurat.

4.4. Evaluasi Model

Hasil evaluasi model klasifikasi pose menunjukkan performa yang cukup baik dengan akurasi sebesar 76,67% dan F1-score sebesar 76,70%. Angka ini mencerminkan bahwa model mampu mengenali tiga jenis gerakan tinju dengan tingkat ketepatan yang cukup tinggi, meskipun masih terdapat ruang untuk peningkatan, terutama dalam mengurangi prediksi yang salah. F1-score yang mendekati nilai akurasi juga menunjukkan keseimbangan yang baik antara precision dan recall, yang penting dalam tugas klasifikasi multi-kelas seperti ini.



Analisis lebih lanjut dari confusion matrix memperlihatkan bagaimana model memprediksi masing-masing kelas. Untuk kelas pertama, model berhasil mengklasifikasikan 84 data dengan benar, tetapi juga melakukan kesalahan dengan memprediksi 9 sebagai kelas kedua dan 13 sebagai kelas ketiga. Pola serupa terjadi pada kelas kedua, dengan 76 prediksi benar dan kesalahan ke kelas lain sebanyak 23 kasus. Kelas ketiga memiliki 70 prediksi yang tepat, namun juga mengalami cukup banyak kesalahan klasifikasi, terutama ke kelas kedua sebanyak 17 kali. Ini menunjukkan bahwa

sebagian besar kesalahan klasifikasi terjadi antar kelas yang memiliki kemiripan bentuk gerakan, seperti antara HOOK dan UPPERCUT.

Secara keseluruhan, metrik evaluasi ini menunjukkan bahwa model sudah cukup andal untuk digunakan dalam aplikasi real-time seperti game pose recognition. Namun, masih memungkinkan dilakukan perbaikan lebih lanjut, misalnya dengan menambahkan data pelatihan, melakukan augmentasi, atau menerapkan teknik normalisasi untuk meningkatkan presisi klasifikasi.

5. Tantangan dan Solusi

Dalam pengembangan proyek game boxing berbasis deteksi gerakan menggunakan MediaPipe, kami menghadapi sejumlah tantangan teknis dan non-teknis yang memengaruhi performa sistem. Berikut adalah uraian lengkap mengenai tantangan yang dihadapi beserta solusi yang diterapkan untuk mengatasinya:

1. Deteksi Tangan Tidak Akurat Saat Bergerak Cepat

Tantangan: Salah satu kendala utama adalah ketidakakuratan deteksi tangan oleh MediaPipe Pose saat pemain melakukan gerakan tinju yang cepat, seperti jab atau hook. Gerakan cepat menyebabkan landmark tangan (wrist) sering kali terdeteksi dengan posisi yang salah atau bahkan hilang dari frame tertentu, sehingga mengganggu klasifikasi gerakan.

Solusi:

- **Peningkatan Kualitas Foto**: Kami meningkatkan kualitas foto menjadi beresolusi tinggi supaya terlihat jelas pada saat pengambilan gambar juga supaya landmark yang terbentuk dari dari mediapipe bisa lebih presisi dengan kualitas foto yang lebih baik.
- **Filter Gerakan (Smoothing)**: Kami menerapkan filter rata-rata bergerak (moving average filter) pada koordinat landmark tangan untuk mengurangi jitter dan menstabilkan deteksi. Filter ini menghitung rata-rata posisi landmark dari beberapa frame sebelumnya, sehingga posisi tangan tetap konsisten meskipun ada frame dengan deteksi yang buruk.
- **Thresholding Kecepatan Gerakan**: Untuk mengatasi deteksi yang salah akibat gerakan terlalu cepat, kami menetapkan batas kecepatan perubahan koordinat landmark. Jika perubahan koordinat melebihi ambang batas tertentu dalam satu frame, sistem akan mengabaikan data tersebut dan menggunakan data dari frame sebelumnya.

2. Sudut Kamera Menyebabkan Landmark Tidak Terdeteksi

Tantangan: Posisi dan sudut kamera yang tidak ideal, seperti kamera yang terlalu rendah atau terlalu miring, sering kali menyebabkan landmark tubuh bagian atas (misalnya, bahu atau siku) tidak terdeteksi dengan baik. Hal ini terutama terjadi ketika pemain berada di posisi menyamping, yang umum dalam gerakan tinju seperti hook.

Solusi:

- **Panduan Posisi Kamera**: Kami menyediakan panduan visual pada antarmuka pengguna (frontend) yang menunjukkan posisi ideal pemain relatif terhadap kamera, seperti menjaga tubuh tetap berada di tengah frame dan kamera pada ketinggian dada. Panduan ini ditampilkan sebagai overlay pada layar game.
- **Rotasi Landmark**: Untuk menangani sudut pandang yang tidak ideal, kami menerapkan transformasi rotasi pada koordinat landmark menggunakan NumPy untuk menormalkan posisi tubuh ke sudut pandang frontal. Transformasi ini memanfaatkan koordinat bahu sebagai acuan untuk menyesuaikan orientasi tubuh.
- **Peningkatan ROI Detection**: Kami menyesuaikan parameter BlazePose Detector pada MediaPipe untuk memperluas Region of Interest (ROI) saat mendeteksi tubuh, sehingga memungkinkan deteksi landmark meskipun tubuh pemain tidak sepenuhnya berada di tengah frame.

3. Latensi Sistem Saat Banyak Proses Berjalan

Tantangan: Sistem mengalami latensi yang signifikan saat menjalankan beberapa proses secara bersamaan, seperti pengambilan video dari webcam, pengolahan pose dengan MediaPipe, klasifikasi gerakan dengan MLPClassifier, dan pengiriman data ke frontend melalui FastAPI. Latensi ini menyebabkan respons game menjadi lambat, mengurangi pengalaman pengguna.

Solusi:

- **Optimasi Pipeline Pemrosesan**: Kami mengurangi beban komputasi dengan membatasi resolusi input video menjadi 720p (1280x720) dari sebelumnya 1080p,

yang masih cukup untuk deteksi pose akurat namun lebih ringan untuk diproses. Selain itu, kami menonaktifkan fitur MediaPipe yang tidak diperlukan, seperti pelacakan wajah dan tangan tambahan, untuk fokus pada pelacakan pose tubuh.

- **Multithreading**: Kami menerapkan multithreading menggunakan modul threading di Python untuk memisahkan proses pengambilan video, pemrosesan pose, dan komunikasi dengan backend. Hal ini memungkinkan setiap tugas berjalan secara paralel, sehingga mengurangi bottleneck.
- **Caching Hasil Klasifikasi**: Untuk mengurangi frekuensi prediksi model, kami menerapkan caching hasil klasifikasi gerakan untuk frame-frame yang mirip (berdasarkan jarak Euclidean antar landmark). Jika perubahan posisi landmark di bawah ambang batas tertentu, sistem akan menggunakan hasil klasifikasi sebelumnya, sehingga mengurangi beban komputasi.

4. Variasi Pencahayaan Memengaruhi Deteksi

Tantangan: Perubahan kondisi pencahayaan, seperti ruangan yang terlalu gelap atau terlalu terang, memengaruhi kualitas deteksi landmark oleh MediaPipe. Dalam kondisi pencahayaan buruk, BlazePose Detector sering gagal menemukan ROI tubuh, menyebabkan sistem tidak dapat mendeteksi pose.

Solusi:

- **Preprocessing Citra**: Kami menerapkan teknik preprocessing citra menggunakan OpenCV, seperti penyesuaian histogram (histogram equalization) untuk meningkatkan kontras gambar di kondisi pencahayaan rendah. Teknik ini membantu BlazePose mendeteksi tubuh dengan lebih baik.
- **Panduan Pencahayaan**: Kami menambahkan panduan di antarmuka pengguna yang menyarankan pemain untuk bermain di ruangan dengan pencahayaan yang cukup, idealnya dengan cahaya alami atau lampu yang merata.
- **Fallback Mechanism**: Jika deteksi ROI gagal selama beberapa frame berturut-turut, sistem akan menampilkan pesan peringatan kepada pengguna untuk memeriksa pencahayaan atau menyesuaikan posisi kamera.

5. Kurangnya Konsistensi Dataset

Tantangan: Dataset gerakan tinju yang dikumpulkan memiliki variasi yang tidak konsisten, misalnya perbedaan sudut pandang, kecepatan gerakan, atau postur tubuh antar sampel. Hal ini menyebabkan model klasifikasi MLPClassifier menghasilkan akurasi yang kurang optimal untuk beberapa jenis gerakan, terutama uppercut.

Solusi:

- **Augmentasi Data**: Kami memperluas dataset dengan menambahkan augmentasi data, seperti rotasi, translasi, dan scaling pada koordinat landmark menggunakan NumPy. Augmentasi ini mensimulasikan variasi sudut pandang dan posisi tubuh, sehingga model lebih robust terhadap data baru.
- **Normalisasi Data**: Kami menormalkan koordinat landmark berdasarkan jarak antar bahu untuk menghilangkan variasi skala akibat perbedaan jarak pemain dari kamera. Normalisasi ini memastikan bahwa fitur input ke model konsisten, terlepas dari ukuran tubuh atau posisi pemain.
- **Pembersihan Dataset**: Kami melakukan validasi manual terhadap dataset untuk menghapus sampel yang memiliki landmark tidak valid (misalnya, koordinat yang hilang atau tidak realistis). Selain itu, kami menyeimbangkan jumlah sampel untuk setiap kelas (jab, hook, uppercut) agar model tidak bias terhadap kelas tertentu.

6. Kendala Non-Teknis: Kurangnya Pengalaman Pengguna

Tantangan: Pengguna yang tidak terbiasa dengan game berbasis gerakan sering kali kesulit memahami cara berinteraksi dengan sistem, seperti posisi berdiri yang benar atau gerakan tinju yang sesuai dengan standar yang diharapkan model.

Solusi:

- **Tutorial Interaktif**: Kami mengembangkan tutorial interaktif di dalam game yang memandu pengguna melalui langkah-langkah dasar, seperti cara melakukan jab, hook, dan uppercut, serta posisi ideal di depan webcam. Tutorial ini dilengkapi dengan visualisasi real-time landmark untuk memberikan feedback langsung.

- **Feedback Visual dan Audio**: Kami menambahkan feedback visual berupa anotasi warna pada landmark (misalnya, hijau untuk deteksi berhasil, merah untuk gagal) dan suara konfirmasi saat gerakan terdeteksi benar. Ini membantu pengguna memahami apakah gerakan mereka dikenali oleh sistem.
- **Dokumentasi Pengguna**: Kami menyediakan dokumentasi sederhana dalam bentuk video dan teks yang menjelaskan cara bermain, yang dapat diakses melalui menu utama game.

6. Kesimpulan

Sistem deteksi gerakan tinju yang dikembangkan berhasil mengklasifikasikan tiga jenis gerakan utama jab, hook, dan uppercut dengan memanfaatkan MediaPipe Pose untuk melacak landmark tubuh bagian atas, seperti bahu, siku, dan pergelangan tangan. Menggunakan model MLPClassifier dari scikit-learn, sistem ini mencapai akurasi 76,67% dan F1-score 76,70%, menunjukkan performa yang cukup andal untuk aplikasi real-time dalam game boxing interaktif. Pendekatan ini memungkinkan pengenalan gerakan pemain secara akurat melalui kamera, menciptakan pengalaman bermain yang imersif dan responsif. Keberhasilan sistem ini didukung oleh efisiensi MediaPipe dengan neural network ringan, kesederhanaan model klasifikasi, serta solusi teknis seperti filter gerakan, normalisasi data, augmentasi dataset, dan multithreading untuk mengatasi tantangan seperti deteksi tangan yang tidak akurat, sudut kamera, latensi, dan variasi pencahayaan. Fitur tambahan seperti tutorial interaktif, feedback visual/audio, dan panduan posisi kamera meningkatkan aksesibilitas bagi pengguna. Solusi ini relevan sebagai fondasi input interaktif untuk game boxing, mendukung kontrol permainan melalui gerakan tubuh alami, dengan potensi peningkatan melalui penambahan data pelatihan dan penyempurnaan deteksi gerakan cepat untuk aplikasi yang lebih kompleks.