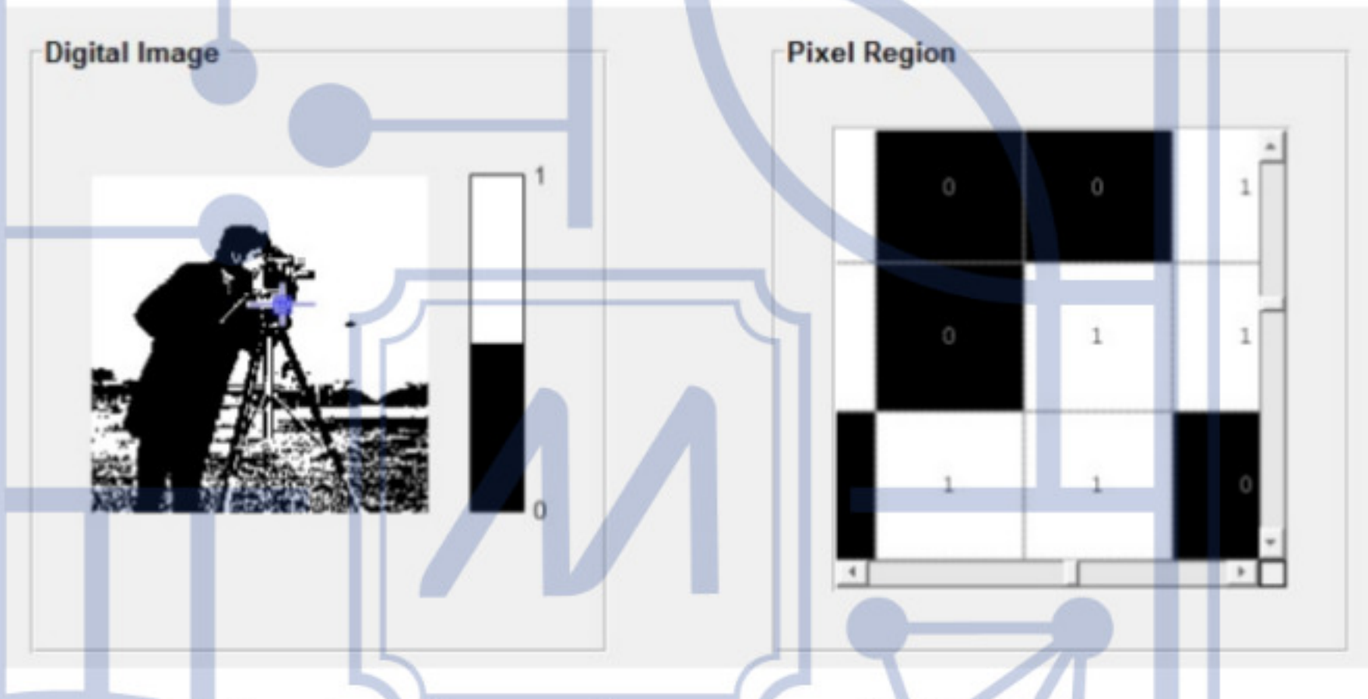
BAB II
KAJIAN LITERATUR

2.1 Citra Digital

Citra digital adalah representasi numerik dari informasi visual pada bidang dua dimensi yang diperoleh melalui proses digitalisasi citra analog (sampling dan kuantisasi). Setiap citra digital direpresentasikan sebagai matriks M×N, di mana setiap elemen matriks (piksel) pada koordinat (x,y) menyimpan nilai intensitas atau nilai komponen warna. [23] [24].



Gambar 2. 1 Citra analog vs Citra digital [25]

Untuk citra *grayscale* nilai intensitas biasanya direpresentasikan dalam rentang 0–255 (8-bit), sedangkan citra warna umum memakai model RGB dengan tiga kanal masing-masing 8-bit, sehingga setiap piksel berisi tiga nilai (R,G,B). Proses pengolahan citra memanfaatkan representasi matriks ini untuk melakukan peningkatan kualitas (mis. *filtering*, histogram *equalization*), restorasi, segmentasi, dan ekstraksi fitur menggunakan teknik spasial maupun frekuensial. Dalam konteks sistem absensi berbasis pengenalan wajah, wajah yang ditangkap kamera direpresentasikan sebagai matriks angka [24] [23].

2.2 *Pre-processing* Citra Wajah

Pre-processing citra wajah merupakan tahap awal yang bertujuan untuk meningkatkan kualitas citra dan menstandarkan representasi wajah sebelum dilakukan ekstraksi fitur atau klasifikasi [26]. Dalam implementasi berbasis SCRFD, citra masukan terlebih dahulu dideteksi untuk menemukan *area* wajah yang valid. Sampel tanpa wajah yang valid dieliminasi, sedangkan wajah yang terdeteksi dipotong pada *area bounding boxes* dan disejajarkan agar posisi wajah menjadi lebih konsisten [27]. Pada beberapa

implementasi, SCRFD juga digunakan untuk memperoleh *landmark* wajah yang selanjutnya dipakai dalam proses *alignment* menggunakan *similarity transformation*. Setelah itu, citra wajah hasil *pre-processing* diubah ke ukuran input 112 × 112 piksel [15], [27], agar sesuai dengan kebutuhan model. Tahapan ini membantu mengurangi gangguan latar belakang, menyeragamkan orientasi wajah, dan meningkatkan kualitas masukan untuk proses pengenalan wajah atau anti-*spoofing* [26], [27], [28].

2.2.1 *Resize*

Resize digunakan untuk mengubah ukuran dimensi citra atau mengatur proporsi *area* wajah terhadap keseluruhan gambar agar *input* menjadi lebih seragam dan sesuai dengan kebutuhan model pengenalan wajah. Dalam penelitian *Yu Xing, resize* dilakukan dengan memotong *area non*-wajah (*cropping*) atau memperluas *area* latar belakang, sehingga proporsi wajah di dalam citra berubah. Hasil penelitian tersebut menunjukkan bahwa ketika wajah menempati bagian yang lebih besar pada citra, performa pengenalan meningkat karena model dapat mengekstraksi fitur wajah dengan lebih efisien. Sebaliknya, penambahan latar belakang yang berlebihan dapat menurunkan performa karena perhatian model terpecah oleh informasi *non*-wajah [29].

Selain itu, *resize* juga digunakan untuk menyeragamkan ukuran *input* citra sebelum diproses lebih lanjut oleh model. Mengubah ukuran citra wajah ke dimensi tetap, 112 × 112 piksel, setelah tahap deteksi, *cropping*, dan *alignment*. Penyeragaman ukuran ini bertujuan agar proses ekstraksi fitur berlangsung lebih konsisten serta sesuai dengan arsitektur jaringan yang digunakan [30], [31]. Dengan mempertahankan *aspect ratio* menggunakan faktor skala minimum:

$$r = \min\left(\frac{W_t}{W}, \frac{H_t}{H}\right)$$
$$W_r = \lfloor W \cdot r \rfloor, H_r = \lfloor H \cdot r \rfloor \quad (1)$$

Di mana:

W, H = lebar dan tinggi citra asli

W_t, H_t = ukuran target model SCRFD,

W_r, H_r = ukuran citra setelah *resize* [32]

2.2.2 *Padding*

Padding digunakan untuk menambahkan piksel pada tepi citra agar ukuran input sesuai dengan ukuran target model tanpa mengubah rasio aspek citra. Teknik ini umumnya

diterapkan setelah *resize* proporsional, sehingga wajah tidak mengalami distorsi bentuk. Seperti pada tahap *pre-processing* untuk model deteksi wajah pada penelitian ini yang menggunakan algoritma SCRFD yang mengharuskan citra di-*resize* ke ukuran 640 × 640 piksel dengan mempertahankan rasio aspek, kemudian dimensi yang lebih pendek diisi menggunakan *letterbox padding* dengan nilai abu-abu konstan. Setelah proses tersebut, koordinat *bounding boxes* dan *landmark* juga disesuaikan agar tetap akurat. Pendekatan ini membantu menghasilkan data masukan yang seragam dan konsisten untuk proses deteksi serta pengenalan wajah [33] dengan persamaan rumus berikut [34].

$$pad_w = W_h - W_s$$
$$pad_h = H_h - H_s \quad (2)$$

Di mana:

W_h = lebar resolusi target

H_h = tinggi resolusi target

W_s = lebar gambar setelah *di-resize*

H_s = tinggi gambar setelah *di-resize*

pad_w = sisa ruang horizontal yang belum terisi

pad_h = sisa ruang vertikal yang belum terisi

Secara umum, *preprocessing* citra wajah berperan penting dalam meningkatkan kualitas citra, mengurangi variasi yang tidak diperlukan, dan menstandarkan representasi wajah agar proses ekstraksi fitur menjadi lebih andal. *Padding* berkontribusi pada standarisasi ukuran *input* serta menjaga proporsi wajah tetap alami selama proses penyesuaian dimensi [26].

$$I_{\text{pad}}(i,j) = \begin{cases} C, & \text{jika } (i,j) \text{ berada pada } \textit{area padding} \\ I(i - P_t, j - P_l), & \text{jika } (i,j) \text{ berada pada } \textit{area} \text{ citra asli} \end{cases} \quad (3)$$

Dimana:

- I = citra hasil *resize*
- I_pad = citra setelah *padding*
- C = nilai konstanta *padding*
- P_t, P_b, P_l, P_r = padding atas, bawah, kiri, kanan [35]

2.2.3 *Crop* Wajah

Crop wajah merupakan proses pemotongan atau pengambilan bagian wajah dari sebuah citra asli agar sistem hanya memfokuskan analisis pada *area* wajah manusia. Namun, pada citra dunia nyata, sebuah gambar dapat berisi lebih dari satu orang, posisi wajah yang berbeda, pencahayaan yang tidak stabil, serta kemungkinan adanya sebagian wajah yang tertutup. Oleh karena itu, proses *crop* wajah perlu dilakukan secara otomatis dan akurat agar wajah aktor utama dapat dipisahkan dari citra asli sebelum diproses lebih lanjut oleh model. berikut menggunakan persamaan rumus [36].

$$B_j = [x_j, y_j, w_j, h_j]$$

Maka:

$$w_j = x_{2,j} - x_1 \quad (4)$$

$$h_j = y_{2,j} - y_1$$

Di mana:

x_j = koordinat horizontal titik kiri atas *bounding box*es,
y_j = koordinat vertikal titik kiri atas *bounding box*es,
w_j = lebar *bounding box*es,
h_j = tinggi *bounding box*es.

2.2.4 *Face Alignment*

Face alignment dijelaskan sebagai Langkah *pre-processing* yang menyelaraskan posisi wajah berdasarkan *landmark* (mis. mata, hidung) sehingga orientasi dan skala wajah menjadi konsisten sebelum ekstraksi fitur oleh jaringan, sehingga *embedding* yang dihasilkan lebih stabil [37]. Variasi *pose* (wajah miring/berputar) dan kondisi pencahayaan menurunkan performa pengenalan, oleh sebab itu normalisasi dan *alignment* diperlukan untuk mengurangi efek *pose* sehingga model *deep-learning* dapat membandingkan *embedding* secara lebih andal.

Penggunaan *facial landmarks* (titik-titik kunci seperti kedua mata, ujung hidung, dan mulut) disebutkan sebagai dasar untuk melakukan *cropping* dan *alignment landmark* ini dipakai untuk *crop/rotate* sehingga fitur penting berada pada posisi yang seragam [37]. Berikut persamaan rumus *face alignment* [38]

$$Q_i = (u_i, v_i)$$

$$dx = x_{\text{mata kanan}} - x_{\text{mata kiri}} \quad (5)$$

$$dy = y_{\text{mata kanan}} - y_{\text{mata kiri}}$$

Di mana:

P_i = titik *landmark* wajah ke-i pada citra sumber atau citra hasil *crop*.

Q_i = titik *landmark* wajah ke-i pada citra target atau *template* referensi.

x_i = koordinat horizontal *landmark* ke-i pada citra sumber.

y_i = koordinat vertikal *landmark* ke-i pada citra sumber.

u_i = koordinat horizontal *landmark* ke-i pada citra target.

v_i = koordinat vertikal *landmark* ke-i pada citra target.

dx = selisih koordinat horizontal antara mata kanan dan mata kiri.

dy = selisih koordinat vertikal antara mata kanan dan mata kiri.

$x_{\text{mata kanan}}$ = koordinat horizontal *landmark* mata kanan.

$x_{\text{mata kiri}}$ = koordinat horizontal *landmark* mata kiri.

$y_{\text{mata kanan}}$ = koordinat vertikal *landmark* mata kanan.

$y_{\text{mata kiri}}$ = koordinat vertikal *landmark* mata kiri.

2.2.5 Transformasi *Landmark* ke *Domain Crop*

Transformasi *landmark* ke domain *crop* adalah proses memindahkan koordinat *landmark* dari domain citra asli ke domain *area* hasil *crop*. Artinya, *landmark* yang semula dinyatakan terhadap seluruh citra diubah menjadi koordinat yang dinyatakan relatif terhadap titik kiri atas *bounding boxes* atau *area crop*. Dengan cara ini, posisi *landmark* menjadi sesuai dengan citra wajah hasil *crop* yang akan diproses pada tahap berikutnya.

Koordinat titik diubah berdasarkan titik kiri atas *bounding box*es melalui persamaan: [39]

$$x_{crop} = x_{original} - x1_{clip} \quad (6)$$
$$y_{crop} = y_{original} - y1_{clip}$$

Di mana:

x_{crop} = koordinat horizontal *landmark* pada domain citra hasil *crop*.

y_{crop} = koordinat vertikal *landmark* pada domain citra hasil *crop*.

$x_{original}$ = koordinat horizontal *landmark* pada citra asli.

$y_{original}$ = koordinat vertikal *landmark* pada citra asli.

$x1_{clip}$ = koordinat horizontal titik kiri atas *area crop*.

$y1_{clip}$ = koordinat vertikal titik kiri atas *area crop*.

2.2.6 Normalisasi

Normalisasi pada citra wajah merupakan tahap *pre-processing* yang bertujuan menstandarkan representasi *input* agar lebih konsisten sebelum ekstraksi fitur atau klasifikasi [26]. Dalam implementasi berbasis *deep learning*, normalisasi dapat dilakukan dengan mengubah nilai piksel RGB dari rentang 0–255 menjadi *tensor* pada rentang mendekati [-1,1] [40].

$$x' = \frac{x - 127{,}5}{128} \tag{7}$$

Maknanya:

- dikurangi 127,5 untuk menggeser pusat data ke sekitar 0
- dibagi 128 untuk mengecilkan skala nilai piksel agar berada di sekitar [-1, 1]

Contoh:

- jika $x = 0$, maka

  $x' = \frac{0 - 127{,}5}{128} = -0{,}996$

- jika $x = 127{,}5$, maka

  $x' = 0$

- jika $x = 255$, maka

  $x' = \frac{255 - 127{,}5}{128} = 0{,}996$

Transformasi ini memusatkan distribusi data di sekitar nol dan mengurangi perbedaan skala antar piksel, sehingga membantu model mengekstraksi fitur wajah secara lebih stabil dan andal.

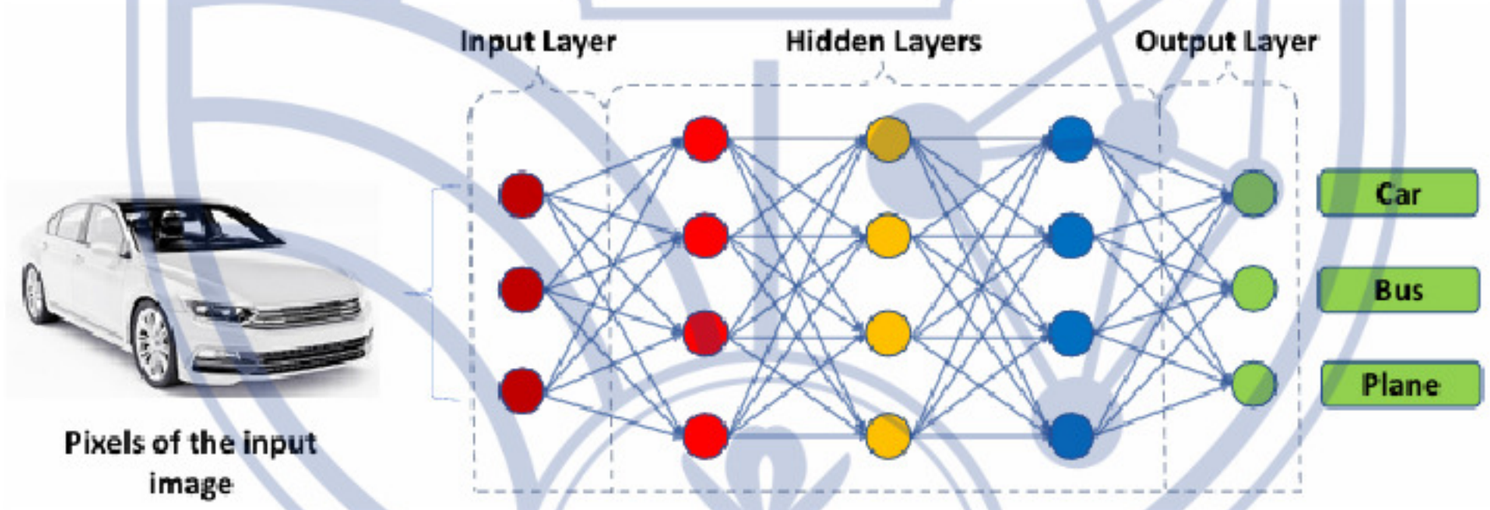
2.3 *Deep Learning*

Deep learning adalah cabang dari *machine learning* yang menggunakan jaringan saraf tiruan berlapis dalam untuk mempelajari representasi fitur secara otomatis dari data mentah. Dalam bidang pengolahan citra, arsitektur konvolusional (CNN) telah terbukti unggul karena kemampuannya mengekstraksi fitur hierarkis mulai dari *edge* dan tekstur pada lapis awal hingga pola semantik pada lapis dalam. Berbeda dengan metode tradisional seperti *Local Binary Pattern* (LBP), *Histogram of Oriented Gradient* (HOG), atau *Scale Invariant Feature Transform* (SIFT) yang membutuhkan perancangan fitur manual, pendekatan *deep learning* memungkinkan pembelajaran *end-to-end* dari piksel ke prediksi sehingga meningkatkan akurasi dan kemampuan generalisasi. Proses pelatihan melibatkan *forward pass*, perhitungan *loss*, dan *backpropagation* untuk memperbarui bobot jaringan berdasarkan

dataset berlabel, teknik *pre-training* pada *dataset* besar diikuti *fine-tuning* sering dipakai untuk mengatasi keterbatasan data tugas. Untuk aplikasi sistem absensi berbasis pengenalan wajah, *deep learning* mampu mengekstraksi *embedding* wajah secara otomatis representasi numerik berdimensi tetap yang kemudian digunakan untuk verifikasi atau identifikasi, sehingga mengeliminasi kebutuhan fitur manual dan meningkatkan ketahanan terhadap variasi pencahayaan, pose, dan skala [41].

2.4 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan salah satu jenis jaringan saraf tiruan yang banyak digunakan dalam bidang pengolahan citra dan pengenalan pola. CNN dirancang untuk mengekstraksi fitur secara otomatis dari data masukan, khususnya citra, melalui susunan lapisan yang bekerja secara hierarkis. Keunggulan utama CNN terletak pada kemampuannya dalam menangkap informasi spasial dan pola lokal pada gambar, sehingga sangat efektif digunakan pada tugas klasifikasi citra, deteksi objek, segmentasi, serta berbagai aplikasi *computer vision* lainnya. Menurut *Krichen*, CNN banyak dimanfaatkan pada *image recognition, speech recognition, natural language processing*, dan berbagai tugas lain yang memerlukan ekstraksi fitur otomatis dari data [42].



Gambar 2. 2 *Architecture* CNN [42]

Secara umum, CNN terdiri atas beberapa lapisan utama yang mana pada penelitian ini menggunakan:

2.4.1 *Convolutional Layer*

Convolutional layer merupakan komponen inti pada CNN yang berfungsi mengekstraksi fitur dari citra masukan melalui penerapan filter atau *kernel* yang dapat dipelajari. Pada proses ini, *kernel* digeser pada *area* lokal citra dan dilakukan operasi perkalian elemen demi elemen yang diikuti penjumlahan untuk menghasilkan nilai pada *feature map* [43].

Untuk ukuran *output feature map*, rumus umumnya adalah:

$$O = \frac{W - F + 2P}{S} + 1 \quad (8)$$

Di mana:

O= ukuran *output*,

W= ukuran *input*,

F= ukuran *kernel*,

P= *padding*,

S= *stride*.

Stride mengatur jarak pergeseran *filter*, sedangkan *padding* mengontrol ukuran/resolusi *feature map* keluaran.

Operasi konvolusi dua dimensi dapat dinyatakan sebagai berikut [43]:

$$Y(i,j) = \sum_{m}^{f-1} \sum_{n}^{f-1} X(i+m, j+n) \cdot K(m,n) \quad (9)$$

Di mana:

Y(i,j) = nilai keluaran pada posisi (i,j) pada *feature map*

X(i+m,j+n) = nilai input pada posisi (i+m,j+n)

K(m,n) = nilai *kernel* pada posisi (m,n)

m,n = indeks *kernel*

2.4.2 *Activation Layer*

Activation layer merupakan lapisan pada *Convolutional Neural Network* (CNN) yang menerapkan fungsi aktivasi terhadap keluaran *convolution layer* untuk memberikan sifat *nonlinier* pada jaringan. Penambahan sifat *nonlinier* ini memungkinkan model mempelajari hubungan yang kompleks, meningkatkan kemampuan ekstraksi fitur, serta membantu proses pelatihan menjadi lebih efektif [44].

Pada penelitian ini, *activation layer* menggunakan fungsi aktivasi *ReLU* pada bagian pendeteksi wajah dan *PreLU* pada bagian pengenalan wajah. Pemilihan *PReLU* didasarkan pada kemampuannya dalam mempertahankan nilai *gradien* pada *area input* negatif melalui parameter α yang bersifat *learnable parameter* dan diperbarui selama proses *training* menggunakan *backpropagation*. Berbeda dengan *Leaky ReLU* yang menggunakan nilai kemiringan (*negative slope*) tetap, nilai α pada *PReLU* dioptimalkan bersama parameter jaringan lainnya sehingga fungsi aktivasi menjadi lebih adaptif terhadap karakteristik data. Menurut Jian Li dkk., penggunaan *PReLU* mampu meningkatkan kemampuan ekstraksi

fitur, mempercepat proses konvergensi, serta menghasilkan *error rate* yang lebih rendah pada model *deep learning* [45].

$$f(x) = \begin{cases} x, & x \geq 0 \\ \alpha x, & x < 0 \end{cases} \quad (10)$$

Di mana:

x= *input* ke *neuron*,

α= parameter yang dipelajari selama *training*,

jika αkecil dan tetap, maka *PReLU* akan mirip *Leaky ReLU,*

jika bagian negatif menjadi nol, maka ia mendekati *ReLU* [45]

Dan untuk *ReLU* mengubah nilai dibawah 0 menjadi 0 dan tidak mengubah nilai jika berada diatas 0 sehingga nilai nya berada diantara 0 hingga tak hingga dan tidak ada nilai negatif.

$$f(x)\ ReLu = max(0, x) \quad (11)$$

Di mana:

f(x) = keluaran fungsi aktivasi

x = nilai masukan dari hasil konvolusi [44]

2.4.3 *Batch Normalization layer*

Batch normalization merupakan teknik normalisasi pada jaringan saraf yang diterapkan pada aktivasi *intermediate* menggunakan statistik *mini-batch*. Proses ini dilakukan dengan menghitung rata-rata dan *varians* pada setiap *batch*, kemudian menormalkan *input* agar memiliki distribusi yang lebih stabil. Setelah itu, hasil normalisasi ditransformasikan kembali menggunakan parameter γ dan β yang dipelajari selama proses *training*. Kehadiran *batch normalization* membantu mempercepat konvergensi, menstabilkan proses pelatihan, dan meningkatkan kemampuan generalisasi model. Dalam perkembangan *deep learning* modern [46].

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \quad (12)$$

$$y_i = \gamma \hat{x}_i + \beta$$

Di mana:

x_i= input ke b*atch normalization*

μ_B= rata-rata *mini-batch*

σ_B^2= *varians mini-batch*

ε= konstanta kecil untuk mencegah pembagian dengan nol

$\hat{x}_i$= hasil normalisasi

γ= parameter skala yang dipelajari

β= parameter *shift* yang dipelajari

2.4.4 *Flatten Layer*

Flatten layer merupakan *layer* pada arsitektur *Convolutional Neural Network* (CNN) yang berfungsi untuk mengubah keluaran berbentuk *tensor* atau *feature map* multidimensi dari hasil proses konvolusi dan *pooling* menjadi vektor satu dimensi [47]. Proses ini diperlukan agar representasi fitur yang telah diekstraksi oleh *layer* sebelumnya dapat diteruskan ke *fully connected layer* atau *dense layer* untuk tahap klasifikasi [48].

Secara fungsional, *flatten layer* tidak mengekstraksi fitur baru seperti *layer* konvolusi, melainkan hanya mengubah bentuk representasi data. Dengan demikian, informasi penting yang telah diperoleh dari proses konvolusi dan *pooling* tetap dipertahankan, namun disusun ulang ke dalam format yang sesuai untuk proses klasifikasi. Sebagai contoh, keluaran *feature map* berukuran $7 \times 7 \times 512$ akan diubah menjadi vektor dengan panjang 25088. Contoh ini menunjukkan bahwa proses *flatten* dilakukan dengan mengalikan seluruh dimensi spasial dan kanal, sehingga seluruh elemen pada *tensor* tetap dipertahankan, hanya bentuknya saja yang berubah [47].

2.4.5 *Fully Connected Layer*

Fully connected layer adalah lapisan pada jaringan saraf di mana setiap *neuron* pada *layer* tersebut terhubung ke seluruh *neuron* pada *layer* sebelumnya. Pada tahap ini, *feature map* yang telah dihasilkan sebelumnya umumnya diubah terlebih dahulu menjadi vektor satu dimensi melalui proses *flattening*, lalu diproses oleh *fully connected layer* untuk melakukan kombinasi fitur secara global dan menghasilkan keputusan klasifikasi akhir [42].

Secara fungsi, *fully connected layer* bertindak sebagai *classifier* pada CNN. Jika *convolutional layer* berperan mengekstraksi fitur lokal seperti tepi, tekstur, dan pola spasial, maka *fully connected layer* menggabungkan seluruh fitur tersebut untuk menentukan kelas *output*. Karena semua *neuron* saling terhubung, layer ini memiliki kemampuan representasi yang kuat, tetapi juga cenderung memiliki jumlah parameter besar sehingga lebih rentan terhadap *overfitting* bila tidak dikendalikan dengan regularisasi [49].

Fully connected layer menggunakan rumus berikut:

$$Fci = \sum_{i=1}^{n} w_{ji}\, x_i + b_j \tag{13}$$

Di mana:

x= vektor *input* dari *layer* sebelumnya,

W= matriks bobot,

b= vektor *bias*,

2.5 Deteksi Wajah

Deteksi wajah merupakan tahap awal dalam sistem pengolahan citra dan visi komputer yang bertujuan untuk menemukan keberadaan serta posisi wajah manusia pada citra atau video. Hasil dari proses ini umumnya berupa koordinat atau *bounding boxes* yang menunjukkan lokasi wajah pada gambar. Dalam sistem pengenalan wajah, tahap deteksi wajah sangat penting karena menjadi dasar bagi proses berikutnya, seperti *cropping*, *alignment*, ekstraksi fitur, pengenalan identitas, dan pelacakan wajah [50].

2.6 SCRFD (*Sample and Computation Redistribution for Face Detection*)

Pada sistem absensi berbasis pengenalan wajah, tahap deteksi wajah perlu dilakukan terlebih dahulu agar sistem dapat menemukan *area* wajah secara akurat sebelum proses *cropping*, *alignment*, dan ekstraksi fitur dilakukan. Tahap ini penting agar citra yang diproses adalah citra yang memang memiliki wajah dan menjaga konsistensi masukan ke model pengenalan wajah, terutama pada kondisi pengambilan citra yang bervariasi [51]. Dalam penelitian ini, metode yang digunakan untuk memenuhi kebutuhan tersebut adalah SCRFD karena mampu mendeteksi wajah secara efisien dan, pada implementasi tertentu, menghasilkan *bounding boxes* serta titik *landmark* wajah yang mendukung proses penyelarasan wajah [52] [53].

SCRFD (*Sample and Computation Redistribution for Face Detection*) merupakan metode deteksi wajah berbasis jaringan saraf konvolusional satu tahap yang dirancang untuk memperoleh keseimbangan yang baik antara akurasi deteksi dan efisiensi komputasi [52]. Metode ini dikembangkan dengan dua gagasan utama, yaitu *Sample Redistribution* (SR) digunakan untuk menyesuaikan distribusi sampel pelatihan agar lebih sesuai dengan kebutuhan deteksi wajah pada berbagai skala, sedangkan *computation redistribution* (CR) digunakan untuk mengalokasikan komputasi secara lebih tepat pada

komponen *backbone, neck,* dan *head* [53]. Dengan pendekatan tersebut, SCRFD mampu menghasilkan deteksi wajah yang efisien sehingga sesuai digunakan pada sistem *real-time* dan perangkat dengan sumber daya komputasi terbatas [52].

Secara arsitektural, SCRFD tersusun atas *backbone, neck,* dan *detection head*. *Backbone* bertugas mengekstraksi fitur dasar dari citra masukan, *neck* bertugas menggabungkan *feature map* dari beberapa skala agar objek wajah dengan ukuran berbeda tetap dapat dideteksi, sedangkan *detection head* menghasilkan prediksi akhir deteksi wajah [52]. Dalam implementasi penelitian ini, alur kerja SCRFD dimulai dari citra masukan yang diproses pada *backbone* untuk memperoleh *feature map* pada beberapa *level*, kemudian *feature map* tersebut digabungkan pada *neck* menggunakan mekanisme *feature pyramid*, lalu diteruskan ke *detection head* untuk menghasilkan prediksi *bounding boxes* dan *landmark* wajah. Setelah itu, hasil prediksi didekodekan dan diseleksi sehingga diperoleh *bounding boxes* akhir beserta lima titik *landmark* wajah yang digunakan pada tahap *pre-processing* berikutnya [53].

Pada bagian *backbone*, citra masukan diproses melalui operasi konvolusi untuk mengekstraksi pola dasar seperti tepi, tekstur, dan struktur visual awal. Operasi ini menghasilkan *feature map* yang merepresentasikan informasi spasial dari wajah pada citra dengan menggunakan persamaan (9) [52].

Selain itu, ukuran keluaran dari proses konvolusi dipengaruhi oleh ukuran *input*, ukuran kernel, *padding,* dan *stride*, sehingga ukuran *output* dapat dihitung dengan persamaan (8) pada proses CNN [54]:

Setelah proses konvolusi, *feature map* diteruskan ke fungsi aktivasi *ReLU* untuk menambahkan sifat *nonlinier* pada jaringan. Fungsi ini mempertahankan nilai positif dan mengubah nilai negatif menjadi nol, sehingga dapat membantu model menonjolkan fitur yang relevan. Fungsi *ReLU* dinyatakan sebagai persamaan (11) [44]:

Melalui rangkaian konvolusi, fungsi aktivasi, dan *downsampling* bertahap, *backbone* menghasilkan *feature map* pada berbagai level resolusi yang selanjutnya digabungkan pada bagian *neck* untuk memungkinkan deteksi wajah dengan berbagai ukuran secara optimal [52].

Pada arsitektur SCRFD, bagian *neck* dibangun dengan prinsip penggabungan fitur multi-skala melalui integrasi antara *backbone, neck,* dan *head* dalam proses redistribusi komputasi [52]. Dalam penelitian ini, proses pada *neck* dijelaskan sebagai penggabungan *feature map* dari beberapa *level* yaitu dari stage 2 hingga stage 4 yang berukuran 80x80,

40x40 dan 20x20 menggunakan *convolution 1×1,* jalur *top-down,* jalur *bottom-up*, operasi *add*, serta *convolution* akhir *neck.* Pada jalur *top-down, feature map* dari *level* yang lebih tinggi di-*upsample* agar memiliki ukuran spasial yang sama dengan *feature map* dari level yang lebih rendah, kemudian keduanya digabungkan untuk memperkaya informasi semantik pada resolusi yang lebih tinggi [55]. Proses tersebut dapat dinyatakan sebagai berikut:

$$P_l = \text{Conv}_{1\times1}(f_l) + \text{Upsample}(P_{l+1}) \quad (14)$$

Di mana:

P_l = fitur hasil *fusi* pada *level* ke-l

F_l = fitur dari *backbone* pada *level* ke-l

P_{l+1} = fitur dari *level* yang lebih tinggi

Upsample = operasi pembesaran ukuran spasial [56]

Selain jalur *top-down*, penggabungan fitur juga dapat diperkuat melalui jalur *bottom-up* untuk menjaga detail spasial dari *level* bawah sekaligus mempertahankan informasi semantik dari *level* atas. Setelah *feature map* dari kedua jalur tersebut memiliki ukuran yang sesuai, dilakukan operasi *add* untuk menjumlahkan *feature map* secara elemen per elemen [55]. Operasi *add* dapat dituliskan sebagai:

$$K = F_{add}(G, H) \quad (15)$$

Di mana:

K adalah *feature map* baru hasil penggabungan

F_*add* adalah operasi *element-wise addition*

G adalah *feature map* pertama

H adalah *feature map* kedua [57]

Setelah proses penjumlahan tersebut, hasil *fusi* diproses kembali menggunakan *convolution* 3 × 3 pada bagian akhir *neck* untuk menyempurnakan representasi fitur sebelum diteruskan ke *detection head* [52] [54].

Pada bagian *detection head, feature map* keluaran *neck* diproses lebih lanjut untuk menghasilkan prediksi akhir deteksi wajah. Penggunaan *feature map* multi-skala pada tahap ini memungkinkan model mendeteksi wajah dalam berbagai ukuran [52]. Keluaran *detection head* selanjutnya diproses melalui tahap *decode* prediksi menggunakan rumus sebagai berikut [58]:

$$\begin{aligned} d\_l &= b\_l \times s \\ d\_t &= b\_t \times s \\ d\_r &= b\_r \times s \end{aligned} \quad (16)$$

$$d\_b = b\_b \times s$$

$$x1 = c\_x - d\_l$$

$$y1 = c\_y - d\_t$$

$$x2 = c\_x + d\_r$$

$$y2 = c\_y + d\_b$$

Di mana:

b_l = *output* regresi model untuk jarak ke sisi kiri *bounding boxes*.

b_t = *output* regresi model untuk jarak ke sisi atas *bounding boxes*.

b_r = *output* regresi model untuk jarak ke sisi kanan *bounding boxes*.

b_b = *output* regresi model untuk jarak ke sisi bawah *bounding boxes*.

s = *stride* pada *feature map*.

d_l = jarak sebenarnya dari titik pusat/*grid* ke sisi kiri *bounding boxes*.

d_t = jarak sebenarnya dari titik pusat/*grid* ke sisi atas *bounding boxes*.

d_r = jarak sebenarnya dari titik pusat/*grid* ke sisi kanan *bounding boxes*.

d_b = jarak sebenarnya dari titik pusat/*grid* ke sisi bawah *bounding boxes*.

c_x = koordinat horizontal titik pusat/*grid*.

c_y = koordinat vertikal titik pusat/*grid*.

x1 = koordinat horizontal sisi kiri *bounding boxes* hasil *decode*.

y1 = koordinat vertikal sisi atas *bounding boxes* hasil *decode*.

x2 = koordinat horizontal sisi kanan *bounding boxes* hasil *decode*.

y2 = koordinat vertikal sisi bawah *bounding boxes* hasil *decode*.

Kemudian seleksi kandidat untuk memperoleh *bounding boxes* akhir dan *landmark* wajah [53]. Berdasarkan *confidence score* dengan menggunakan rumus *sigmoid* berikut [59].

$$y = \frac{1}{1 + e^{-x}} \tag{17}$$

Di mana:

y = nilai keluaran dari fungsi *sigmoid*

x = nilai masukan ke fungsi *sigmoid*

e = konstanta *Euler*, dengan nilai mendekati 2,71828

Pada tahap *decode* prediksi, keluaran mentah dari *detection head* diubah menjadi koordinat *bounding boxes* dan *keypoint* wajah pada domain citra asli. Proses ini diperlukan

karena prediksi yang dihasilkan jaringan masih berada dalam bentuk representasi internal model, sehingga belum dapat langsung digunakan sebagai lokasi wajah akhir. Setelah proses *decode* selesai, dilakukan seleksi kandidat untuk menyaring hasil deteksi yang memiliki kualitas rendah atau saling bertumpang tindih. Kandidat dengan nilai keyakinan di bawah ambang batas akan dibuang, sedangkan kandidat yang memiliki *overlap* tinggi akan disaring menggunakan *Non-Maximum Suppression* (NMS) agar hanya kandidat terbaik yang dipertahankan [60]. Ukuran *overlap* antar *bounding boxes* umumnya dinyatakan menggunakan *Intersection over Union* (IoU), yang dirumuskan sebagai berikut:

$$\mathrm{IoU}(A,B) = \frac{|\,A \cap B\,|}{|\,A \cup B\,|} \tag{18}$$

Di mana:

A dan B = dua *bounding boxes*

$|A \cap B|$ = luas irisan kedua kotak

$|A \cup B|$ = luas gabungan kedua kotak

Dengan demikian, keluaran akhir SCRFD berupa *bounding boxes* wajah dan, pada varian tertentu, 5 titik *landmark* wajah yang dapat digunakan untuk proses analisis wajah lanjutan.

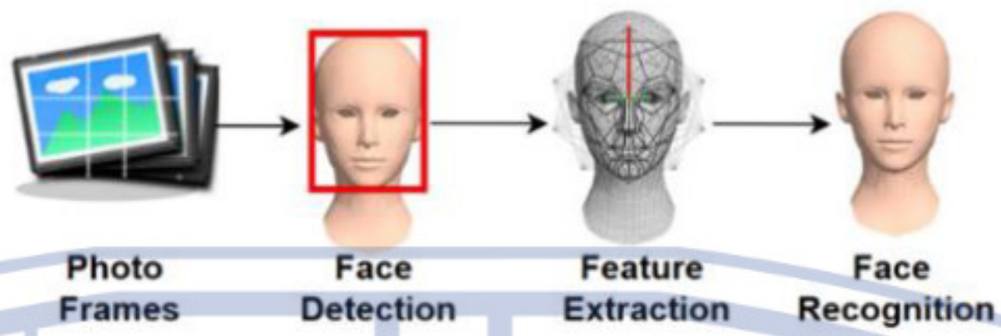
2.7 Pengenalan Wajah

Pengenalan wajah merupakan teknologi biometrik yang digunakan untuk mengidentifikasi, mengklasifikasikan, maupun memverifikasi identitas individu berdasarkan karakteristik wajah. Teknologi ini bekerja dengan memanfaatkan algoritma pembelajaran mesin dan pengolahan citra untuk menganalisis fitur unik wajah, seperti jarak antar mata, bentuk hidung, serta kontur wajah. Proses pengenalan wajah dilakukan dengan membandingkan pola atau fitur wajah individu dari citra digital maupun rekaman video untuk menentukan kecocokan identitas seseorang secara otomatis [37], [61].

Dalam penerapannya terdapat dua skenario, yaitu verifikasi dan identifikasi. Verifikasi (*one-to-one*) dilakukan dengan mencocokkan wajah dengan satu data referensi untuk memastikan identitas yang diklaim, sedangkan identifikasi (*one-to-many*) membandingkan wajah dengan seluruh data dalam basis data untuk mengetahui siapa individu tersebut [37].

Proses pengenalan wajah meliputi beberapa tahap, yaitu deteksi wajah (*face detection*), *pre-processing* dan penyelarasan citra, ekstraksi fitur, serta pencocokan identitas. *Face detection* hanya berfungsi menemukan lokasi wajah pada gambar, sedangkan *face recognition* bertujuan mengenali identitasnya melalui pembentukan vektor fitur (*embedding*)

dan perbandingan tingkat kemiripan. Pada pendekatan modern berbasis *deep learning, embedding* wajah dihasilkan oleh jaringan saraf dan dioptimalkan menggunakan metode seperti *ArcFace* agar wajah dari identitas yang sama memiliki jarak fitur yang berdekatan dalam ruang fitur [37] [62].

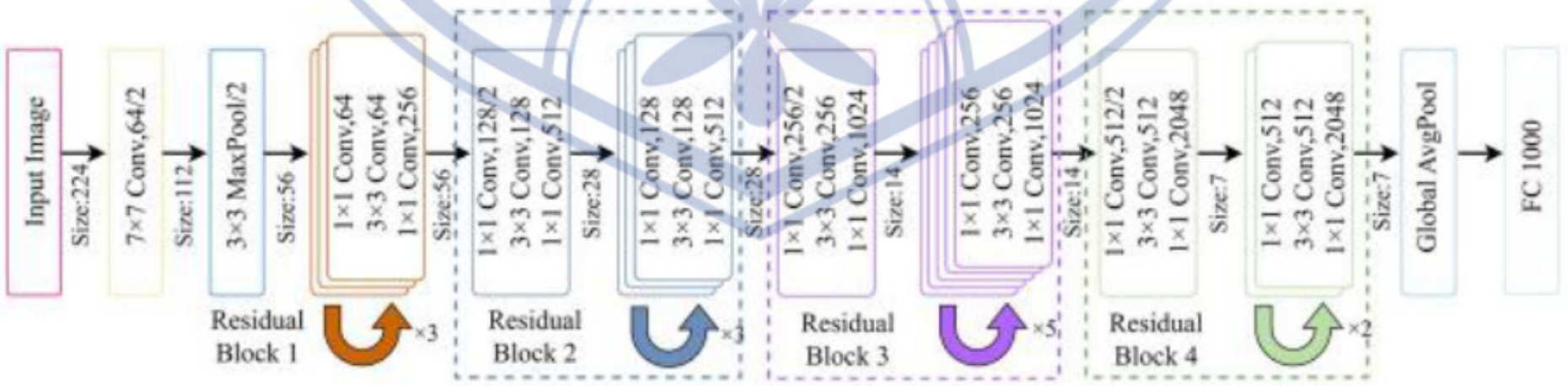


Gambar 2. 3 Proses Pengenalan Wajah [9]

Teknologi pengenalan wajah banyak dimanfaatkan pada sistem keamanan, pembuka kunci perangkat, dan sistem absensi otomatis. Sistem absensi berbasis wajah termasuk dalam kategori biometrik karena menggunakan karakteristik biologis unik individu sebagai media autentikasi kehadiran [37].

2.8 *ResNet-50*

ResNet-50 merupakan salah satu arsitektur *Convolutional Neural Network* yang menerapkan konsep *deep residual learning* melalui *skip connection*. Mekanisme tersebut memungkinkan jaringan yang dalam tetap dapat dilatih secara stabil, karena informasi dari lapisan sebelumnya masih dapat diteruskan ke lapisan berikutnya tanpa harus selalu melalui transformasi konvolusi penuh. Dengan pendekatan ini, *ResNet-50* menjadi lebih efektif dalam mengatasi penurunan performa akibat bertambahnya kedalaman jaringan dan tetap mampu membentuk representasi fitur yang kuat [63].

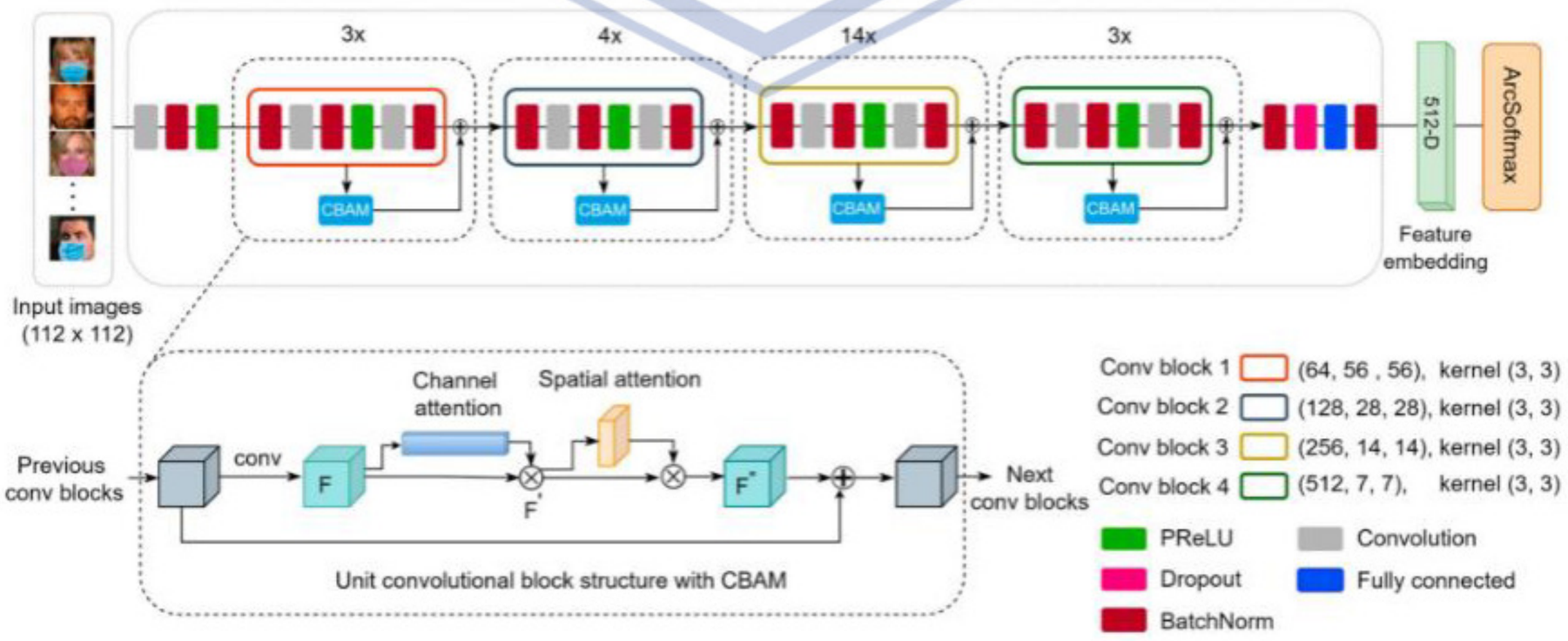


Gambar 2. 4 Arsitektur *ResNet-50* [64]

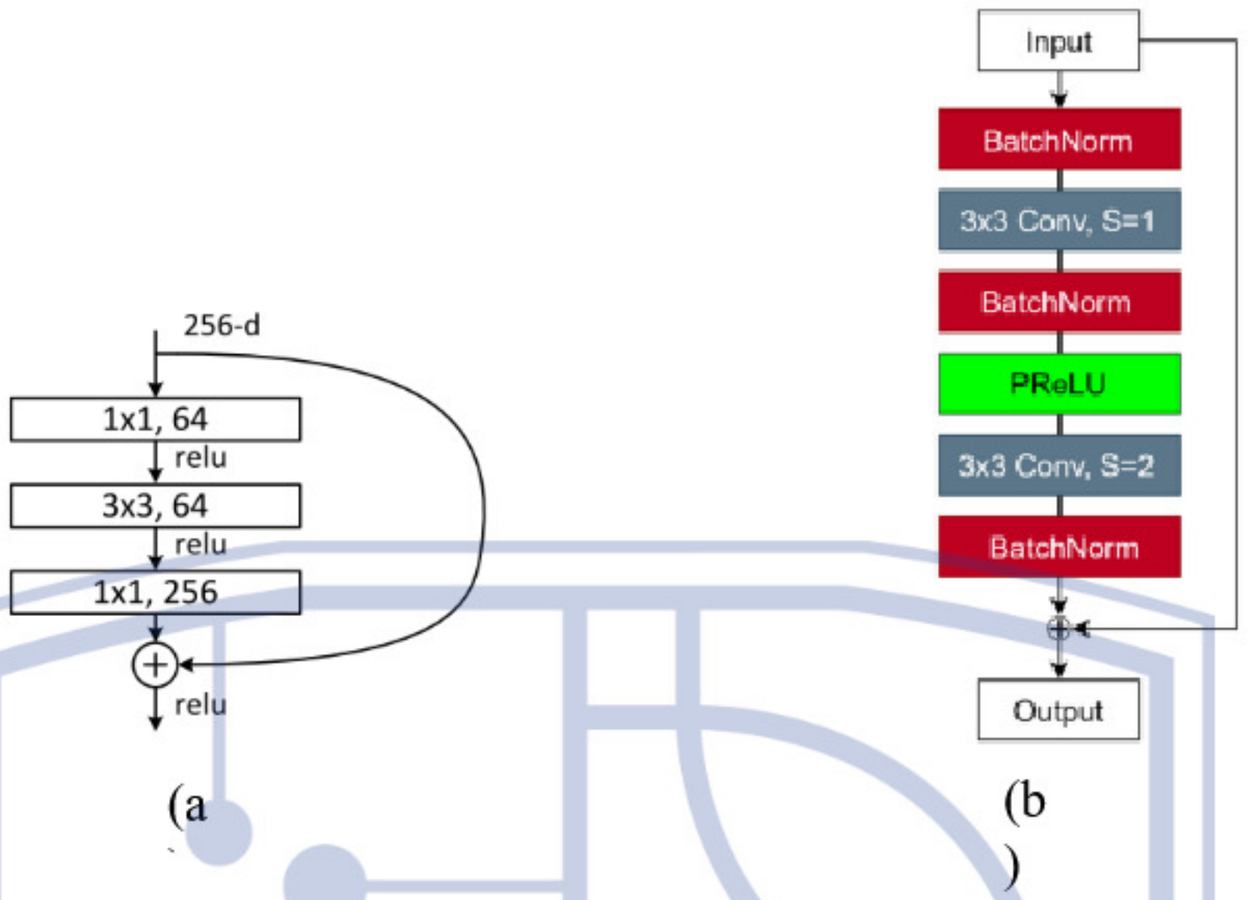
Pada bentuk standarnya, *ResNet-50* banyak diterapkan pada klasifikasi citra umum, di mana fitur hasil ekstraksi diarahkan ke lapisan klasifikasi akhir menggunakan *fully connected layer* dan *SoftMax* [63] [65]. Meskipun demikian, kebutuhan pada sistem

pengenalan wajah berbeda dengan klasifikasi citra umum. Pada klasifikasi umum, tujuan utama model adalah memetakan citra ke kelas-kelas tertentu yang telah ditentukan pada tahap pelatihan. Sebaliknya, pada *face recognition,* model perlu membentuk representasi fitur wajah yang tetap diskriminatif meskipun identitas yang diuji tidak selalu muncul pada data latih. Oleh karena itu, literatur Mutakhir menjelaskan bahwa sistem pengenalan wajah modern lebih menekankan pembelajaran *embedding* daripada sekadar keluaran klasifikasi, karena *embedding* tersebut nantinya digunakan kembali pada proses verifikasi satu-banding-satu maupun identifikasi satu-banding-banyak [66] [67]. Pendekatan ini juga diperkuat oleh penggunaan *loss function* berbasis margin sudut, seperti *ArcFace* dan metode sejenis, yang dirancang untuk memperkecil jarak intrakelas dan memperbesar jarak antarkelas pada ruang fitur, sehingga lebih sesuai untuk skenario *open-set face recognition* dibandingkan *softmax* klasifikasi biasa [66] [67] [21].

Berdasarkan karakteristik tersebut, *backbone ResNet-50* yang digunakan pada sistem pengenalan wajah umumnya tidak dipakai dalam bentuk standar, tetapi dimodifikasi agar lebih sesuai untuk ekstraksi ciri wajah. Dalam konteks ini, modifikasi arsitektur dilakukan bukan hanya pada bagian akhir model, tetapi juga pada bagian awal dan susunan blok residualnya. Pada *backbone ResNet-50* modifikasi yang digunakan dalam penelitian ini, lapisan konvolusi awal tidak lagi memakai *kernel* 7x7 seperti pada *ResNet-50* standar, melainkan diganti menjadi konvolusi 3x3. Perubahan ini membuat ekstraksi fitur awal menjadi lebih sesuai untuk citra wajah yang berukuran lebih kecil dan lebih terfokus pada detail lokal wajah. Selain itu, setelah lapisan awal tersebut, arsitektur ini juga tidak lagi menggunakan *max pooling*, sehingga informasi spasial pada tahap awal dapat dipertahankan lebih lama sebelum mengalami reduksi ukuran fitur. Pola seperti ini sejalan dengan kecenderungan *backbone face recognition* modern yang mempertahankan detail wajah lebih baik dibandingkan arsitektur klasifikasi umum [66] [21] [68].



Gambar 2. 5 Arsitektur *ResNet-50 Modified* [10]



Gambar 2. 6 (a) *Block Residual Bottleneck ResNet-50 with relu* [64], (b) *Block Residual Bottleneck ResNet-50 with prelu* [10]

Perbedaan lainnya terlihat pada susunan blok *residual*. Jika *ResNet-50* standar secara umum menggunakan komposisi blok 3, 4, 6, 3, maka *backbone ResNet-50* modifikasi pada penelitian ini menggunakan komposisi 3, 4, 14, 3. Penambahan jumlah blok pada tahap menengah menunjukkan bahwa proses ekstraksi fitur diperkuat pada bagian representasi yang lebih dalam, sehingga model memiliki kapasitas yang lebih besar untuk mempelajari karakteristik wajah yang halus dan diskriminatif. Pada bagian aktivasi, fungsi *ReLU* juga tidak lagi digunakan, tetapi diganti dengan *PReLU*. Penggunaan *PReLU* memberi fleksibilitas lebih besar pada aktivasi negatif dibandingkan *ReLU*, sehingga model dapat mempertahankan lebih banyak informasi fitur dan menyesuaikan *respons* aktivasi secara adaptif selama pelatihan. [69] [68] [70].

Pada bagian akhir arsitektur juga terdapat perbedaan yang cukup penting. *ResNet-50* standar umumnya memanfaatkan *average pooling* atau *global average pooling* sebelum fitur dipetakan ke lapisan klasifikasi akhir [63] [65]. Namun, pada *backbone ResNet-50* modifikasi yang digunakan dalam penelitian ini, *average pooling* maupun *global average pooling* tidak lagi digunakan. Sebagai gantinya, fitur hasil ekstraksi dari lapisan konvolusi akhir langsung diubah ke bentuk satu dimensi menggunakan *flatten*. Pendekatan ini kemudian diikuti oleh pembentukan vektor fitur tetap yang digunakan sebagai representasi wajah. Dengan demikian, keluaran akhir *backbone* ini bukan diarahkan sebagai probabilitas kelas, melainkan sebagai *embedding* wajah berdimensi tetap yang dipakai untuk proses pencocokan identitas. Karakteristik tersebut lebih selaras dengan tujuan *face recognition*,

yaitu menghasilkan representasi wajah yang konsisten dan mudah dibandingkan menggunakan ukuran kemiripan seperti *cosine similarity* [66] [67] [68].

2.9 *Backpropagation*

Backpropagation merupakan proses pelatihan pada jaringan saraf tiruan yang digunakan untuk menghitung pengaruh kesalahan atau *loss* terhadap parameter model, seperti bobot dan bias. Proses ini dilakukan dari lapisan keluaran menuju lapisan sebelumnya dengan menerapkan prinsip turunan berantai, sehingga sistem dapat mengetahui seberapa besar perubahan setiap bobot memengaruhi nilai kesalahan model. Proses *backpropagation* berkaitan dengan perhitungan *gradien loss* terhadap bobot model. *Gradien* menunjukkan arah dan besar perubahan *loss* apabila suatu bobot diubah. Semakin besar nilai *gradien*, semakin besar pula pengaruh bobot tersebut terhadap kesalahan model [71].

Rumus *gradien* terhadap bobot sebagai berikut:

$$gradien = \frac{\partial E}{\partial W_l} \tag{19}$$

Di mana:

l = nilai *loss*

W = bobot model

∂E / ∂W = perubahan nilai *loss* terhadap bobot

Setelah nilai *gradien* diperoleh, bobot model diperbarui menggunakan aturan pembaruan berbasis *gradient descent*. Rumus pembaruan bobot sebagai berikut [71]:

$$W\_new = W\_old - \eta \frac{\partial E}{\partial W_l} \tag{20}$$

Di mana:

W_*new* = bobot setelah diperbarui

W_*old* = bobot sebelum diperbarui

η = *learning rate*

∂E/∂W = *gradien loss* terhadap bobot

Learning rate merupakan nilai kecil yang ditentukan untuk mengatur besar langkah pembaruan bobot pada setiap iterasi pelatihan [72]. Jika *learning rate* terlalu besar, proses pembaruan bobot dapat menjadi tidak stabil karena perubahan bobot terlalu jauh. Sebaliknya, jika *learning rate* terlalu kecil, proses pelatihan dapat berjalan lambat karena perubahan bobot terlalu sedikit pada setiap iterasi. Oleh karena itu, *learning rate* berfungsi

sebagai pengendali besar langkah optimasi agar proses pelatihan model dapat bergerak menuju nilai *loss* yang lebih kecil secara lebih terarah [73].

2.10 *ArcFace*

ArcFace merupakan salah satu algoritma pada sistem pengenalan wajah yang dirancang untuk meningkatkan kualitas hasil identifikasi dengan menerapkan konsep margin sudut pada representasi fitur wajah. Penerapan margin tersebut bertujuan agar perbedaan karakteristik antar identitas wajah menjadi lebih jelas sehingga kemampuan model dalam melakukan klasifikasi dapat meningkat [37].

Penggunaan *ArcFace* banyak diterapkan dalam berbagai sistem biometrik modern karena kemampuannya dalam menghasilkan *embedding* wajah yang lebih stabil dan *robust* terhadap variasi ekspresi, pencahayaan, maupun sudut pengambilan citra. Penelitian menunjukkan bahwa *ArcFace* mampu menghasilkan *performa* pengenalan wajah yang lebih baik dibandingkan beberapa metode lain karena mampu memaksimalkan pemisahan antar identitas secara lebih optimal. Selain itu, algoritma ini sering digunakan dalam sistem keamanan, verifikasi identitas, serta aplikasi yang membutuhkan tingkat ketelitian tinggi [74].

Dalam penelitian komparatif yang membandingkan beberapa algoritma pengenalan wajah seperti CNN, *FaceNet*, dan *ArcFace*, diketahui bahwa *ArcFace* memiliki keunggulan dalam pengolahan *dataset* berukuran besar. *ArcFace* mampu mencapai tingkat akurasi hingga sekitar 98% pada kondisi tertentu, khususnya ketika model dilatih menggunakan *dataset* yang memiliki variasi wajah yang tinggi. Namun, performa algoritma ini dapat mengalami penurunan pada kondisi lingkungan yang tidak terkontrol atau ketika jumlah *dataset* pelatihan terbatas [37].

Selain digunakan pada penelitian berbasis analisis algoritma, *ArcFace* juga telah diterapkan dalam berbagai sistem nyata seperti sistem absensi berbasis pengenalan wajah. Pada implementasi sistem absensi berbasis visi komputer, *ArcFace* digunakan bersama model *deep learning* lainnya seperti *ResNet* untuk melakukan ekstraksi fitur wajah dan proses pencocokan identitas. Hasil penelitian menunjukkan bahwa penggunaan *ArcFace* mampu meningkatkan akurasi sistem absensi hingga lebih dari 95% dengan waktu pemrosesan yang relatif cepat, sehingga cocok digunakan pada aplikasi *real-time* berbasis perangkat *mobile* maupun sistem berbasis *cloud* [9].

Pengembangan terbaru terhadap *ArcFace* juga menunjukkan bahwa fungsi *loss* ini tidak hanya digunakan pada pengenalan wajah, tetapi dapat diterapkan pada berbagai tugas

klasifikasi citra lainnya. Penelitian menunjukkan bahwa *ArcFace* bekerja dengan menambahkan margin sudut pada nilai kesamaan kosinus antar *embedding* fitur sehingga memperbesar jarak sudut antar kelas dalam ruang fitur. Pendekatan ini meningkatkan kemampuan model dalam menghasilkan representasi fitur yang lebih diskriminatif serta meningkatkan performa klasifikasi tanpa harus menambah kompleksitas arsitektur jaringan saraf [62].

Meskipun memiliki tingkat akurasi yang tinggi, penggunaan *ArcFace* juga memiliki beberapa tantangan. Berdasarkan studi literatur pada penelitian pengenalan wajah dalam kondisi lingkungan yang kompleks, *ArcFace* cenderung memberikan performa optimal pada kondisi citra wajah yang relatif terkontrol, seperti pencahayaan yang baik dan posisi wajah yang jelas. Namun, *performa* model dapat menurun ketika dihadapkan pada kondisi dunia nyata seperti *occlusion*, variasi *pose* ekstrem, atau kualitas citra yang rendah. Oleh karena itu, pengembangan lanjutan diperlukan untuk meningkatkan ketahanan model terhadap kondisi lingkungan yang dinamis [18].

Secara keseluruhan, algoritma *ArcFace* merupakan salah satu metode pengenalan wajah yang efektif dan banyak digunakan dalam berbagai aplikasi biometrik modern. Keunggulan utama algoritma ini terletak pada kemampuannya menghasilkan fitur wajah yang lebih diskriminatif melalui pendekatan margin sudut, sehingga meningkatkan akurasi sistem pengenalan wajah. Namun, efektivitas *ArcFace* tetap dipengaruhi oleh kualitas *dataset* dan kondisi lingkungan saat proses pengenalan wajah dilakukan. *ArcFace* menggunakan rumus berikut [10]:

$$L_{ArcFace} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j=1,\, j\neq y_i}^{n} e^{s\cos\theta_j}} \quad (21)$$

Di mana:

$L_{ArcFace}$ = Nilai *loss* dari metode *ArcFace*

N = Ukuran *batch* atau jumlah sampel data dalam satu proses pelatihan

n = Jumlah kelas atau jumlah identitas wajah

θj = Sudut antara bobot kelas dan fitur wajah hasil ekstraksi *deep learning*

yi = Label kelas yang benar dari sampel ke-*i*

s = *Feature scale*, yaitu faktor pengali untuk memperbesar nilai fitur

m = *Angular margin penalty*, yaitu margin sudut tambahan agar jarak antar kelas lebih jelas

e = Fungsi eksponensial yang digunakan dalam perhitungan *softmax*.

2.11 *Cosine Similarity*

Cosine Similarity adalah metode pengukuran kemiripan dua data yang direpresentasikan dalam bentuk vektor fitur dengan menghitung nilai *cosinus* sudut antara kedua vektor. Metode ini tidak mengukur jarak langsung antar data, melainkan membandingkan orientasi vektor, sehingga semakin kecil sudut antar vektor maka semakin besar nilai kemiripan yang dihasilkan dan menunjukkan kedua data memiliki identitas yang sama [75], [76].

$$Simi\ (\mathrm{V,T}) = cos(\theta) = \frac{V.T}{||V||||T||} \quad (22)$$

Di mana:

V= vektor fitur wajah *input* (*embedding citra*)

T= vektor nilai bobot kelas

$||\ V\ ||$= jumlah vektor V

$||\ T\ ||$= jumlah vektor T

θ= sudut antara dua vektor

2.12 *Liveness Detection*

Liveness Detection atau *Face Liveness Detection* merupakan teknik pada sistem biometrik yang digunakan untuk memastikan bahwa wajah yang terdeteksi berasal dari individu nyata (hidup) dan bukan dari media tiruan seperti foto, video, maupun topeng. Metode ini dikembangkan untuk melindungi sistem pengenalan wajah dari serangan *spoofing* atau *presentation attack* yang bertujuan menipu sistem autentikasi dengan menggunakan representasi wajah palsu [77].

Liveness Detection menjadi komponen penting dalam sistem pengenalan wajah karena proses verifikasi identitas tidak hanya perlu mengenali kesamaan wajah, tetapi juga memastikan bahwa wajah yang diproses berasal dari pengguna yang hadir secara langsung. Tanpa mekanisme *liveness detection,* sistem *face recognition* berpotensi menerima serangan presentasi seperti foto cetak, tampilan wajah pada layar perangkat, rekaman video, atau media tiruan lainnya yang menyerupai wajah pengguna sah [77] [78]. Oleh karena itu, *liveness detection* digunakan sebagai lapisan keamanan tambahan untuk membedakan wajah asli dan wajah palsu sebelum sistem melanjutkan proses verifikasi identitas. Pembahasan lebih lanjut mengenai mekanisme pencegahan *spoofing* dan metode verifikasi gerakan

kepala akan dijelaskan pada bagian *Anti-Spoofing*, *Head Pose Estimation* dan *MediaPipe Face Mesh.*

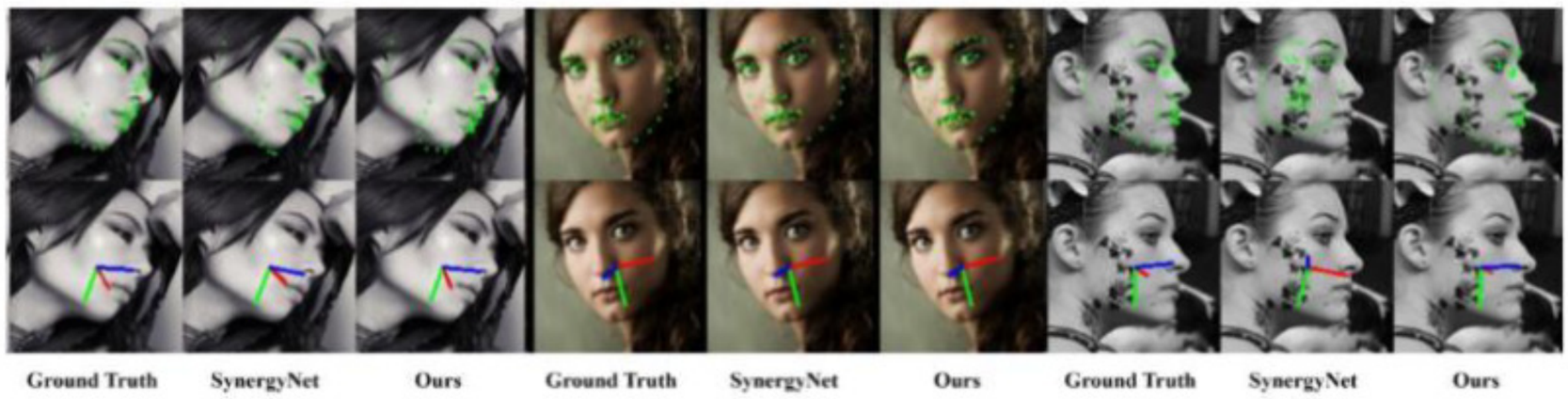
2.12.1 *Anti-Spoofing*

Anti-spoofing pada sistem pengenalan wajah merupakan mekanisme pengamanan yang digunakan untuk mencegah serangan *spoofing* atau *presentation attack*, yaitu upaya menipu sistem biometrik dengan menyajikan artefak wajah palsu kepada kamera atau sensor, seperti foto, video *replay*, layar perangkat, maupun media tiruan lainnya [77] [78]. *Anti-spoofing* bertujuan sebagai pendekatan perlindungan untuk membedakan wajah asli dan wajah palsu sebelum sistem melakukan verifikasi identitas pengguna [78].

Dalam konteks sistem absensi berbasis website, *anti-spoofing* diperlukan karena pengenalan wajah belum cukup untuk memastikan bahwa pengguna benar-benar hadir secara langsung di depan kamera. Sistem *face recognition* dapat mengenali kemiripan wajah, tetapi masih berpotensi tertipu oleh foto atau video karena *presentation attack* dapat dibuat dari foto atau video pengguna yang tersedia secara publik, misalnya dari media sosial. Oleh karena itu, *liveness detection* dapat diposisikan sebagai salah satu teknik utama dalam *anti-spoofing* karena digunakan untuk melindungi sistem autentikasi biometrik dari serangan *spoofing* seperti *printed photo* dan video *replay*. *Liveness detection* membedakan wajah hidup dan wajah palsu dengan memanfaatkan indikator kehidupan atau respons dinamis, seperti kedipan mata, gerakan bibir, perubahan ekspresi wajah, denyut nadi, maupun gerakan kepala. Pada penelitian ini, penggunaan *head pose estimation* dapat dikaitkan dengan pendekatan *motion-based liveness detection*, karena gerakan kepala termasuk salah satu indikator dinamis yang dapat digunakan untuk membedakan wajah hidup dari media statis seperti foto [77].

2.12.2 *Head Pose Estimation*

Head pose estimation adalah proses memperkirakan orientasi kepala manusia terhadap kamera atau sistem koordinat citra. Secara umum, orientasi kepala direpresentasikan menggunakan tiga sudut *Euler,* yaitu *yaw, pitch,* dan *roll*. *Yaw* menunjukkan gerakan kepala ke kiri atau ke kanan, *pitch* menunjukkan gerakan kepala menunduk atau mendongak, sedangkan *roll* menunjukkan kemiringan kepala ke samping. Informasi *pose* kepala penting dalam analisis wajah karena perubahan sudut kepala dapat memengaruhi tampilan wajah, posisi *landmark*, serta akurasi pengenalan wajah [21].



Gambar 2. 7 *Head Pose Estimation* [21]

Dalam gambar 2.7 mendeteksi *landmark* wajah serta memperkirakan orientasi kepala. Baris atas menunjukkan *landmark* wajah, sedangkan baris bawah menunjukkan sumbu *pose* kepala.

2.12.3 *MediaPipe Face Mesh*

MediaPipe Face Mesh merupakan metode pendeteksian *landmark* wajah yang digunakan untuk memperoleh titik-titik penting pada wajah secara *real-time* melalui citra kamera. *MediaPipe Face Mesh* mampu menghasilkan 468 titik *landmark* wajah 3D yang merepresentasikan bagian-bagian wajah seperti mata, hidung, mulut, rahang, dan kontur wajah [79], [80].

MediaPipe Face Mesh digunakan untuk mendukung proses *liveness detection* berbasis gerakan kepala pada sistem absensi wajah. Prosesnya dimulai dari pengambilan citra melalui kamera, deteksi *area* wajah, estimasi *landmark* wajah, lalu *landmark* tersebut digunakan sebagai acuan untuk memperkirakan *pose* kepala, seperti menoleh ke kiri, menoleh ke kanan, menunduk, atau kembali menghadap kamera [79]. Estimasi *pose* kepala dilakukan dengan mendeteksi *landmark* wajah 2D, mencocokkannya dengan titik pada model wajah atau kepala 3D, kemudian menghitung *pose* menggunakan metode *Perspective-n-Point* (PnP) untuk memperoleh sudut orientasi kepala, yaitu *yaw, pitch,* dan *roll* [81], [82].

2.13 Teknologi Berbasis Lokasi

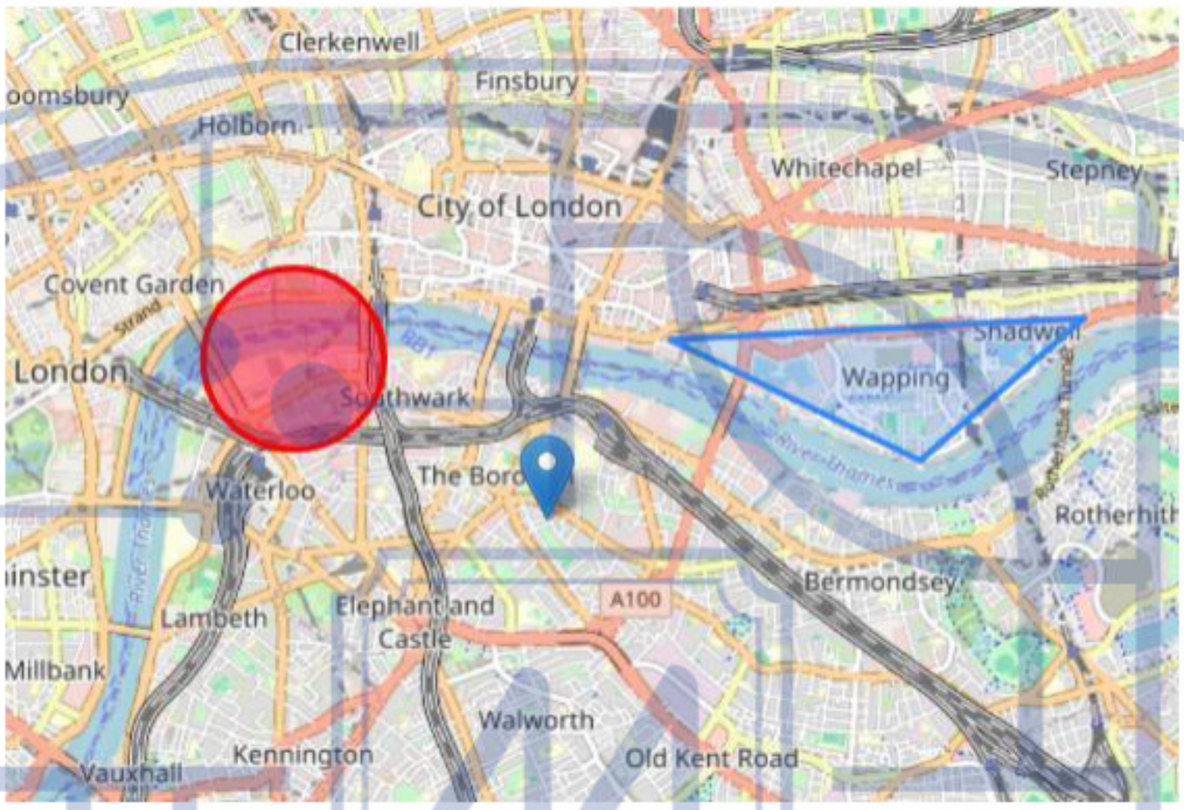
Teknologi berbasis lokasi memanfaatkan informasi posisi geografis pengguna atau perangkat untuk menyediakan layanan yang bergantung pada lokasi. Pada aplikasi berbasis web maupun *mobile*, teknologi *geolocation* dapat digunakan bersama layanan peta digital untuk mengetahui lokasi pengguna secara *real-time* atau mendekati *real-time* [83].

Dengan menggunakan Formula *Haversine*, sistem dapat menghitung jarak antara koordinat pengguna dan titik pusat presensi, sehingga presensi hanya dinyatakan berhasil apabila pengguna berada dalam radius yang telah ditentukan [84]. Teknologi berbasis lokasi

yang digunakan dalam penelitian ini meliputi *Map Pinning, Geolocation,* HTML5 *Geolocation* API, *Haversine* dan *Geofencing.*

2.13.1 *Map Pinning*

Map Pinning merupakan proses penempatan *marker* pada peta digital untuk menunjukkan lokasi tertentu. *Marker* tersebut berfungsi sebagai penanda visual lokasi tetap yang menjadi acuan pembatasan lokasi presensi berdasarkan koordinat geografis [85].



Gambar 2. 8 *Pin Mapping* [86]

Gambar 2.8 tersebut merupakan contoh penerapan *map pinning* pada peta interaktif menggunakan *Leaflet*. Pin berwarna biru digunakan sebagai penanda titik lokasi tertentu berdasarkan koordinat geografis. Selain itu, terdapat lingkaran merah yang menunjukkan *area* atau radius tertentu, serta poligon biru yang menggambarkan batas suatu wilayah [86].

2.13.2 *Geolocation*

Geolocation merupakan teknologi yang digunakan untuk menentukan atau mengidentifikasi lokasi geografis suatu perangkat, pengguna, atau sumber data berdasarkan informasi tertentu seperti alamat IP, GPS, jaringan *Wi-Fi*, maupun sinyal seluler. *Geolocation* berfungsi sebagai penghubung antara dunia digital dan dunia fisik dengan menyediakan informasi lokasi yang dapat dimanfaatkan dalam berbagai layanan berbasis lokasi, seperti navigasi, keamanan sistem, serta rekomendasi layanan berdasarkan lokasi pengguna [87].

Dalam penerapannya, data lokasi umumnya direpresentasikan dalam bentuk koordinat lintang dan bujur, sehingga dapat diproses oleh sistem untuk mengetahui posisi pengguna terhadap suatu titik acuan tertentu. Pada aplikasi berbasis web, *geolocation* dapat

digunakan untuk memperoleh posisi pengguna secara *real-time* atau mendekati *real-time*, kemudian menampilkan atau memanfaatkan posisi tersebut dalam layanan peta digital [83].

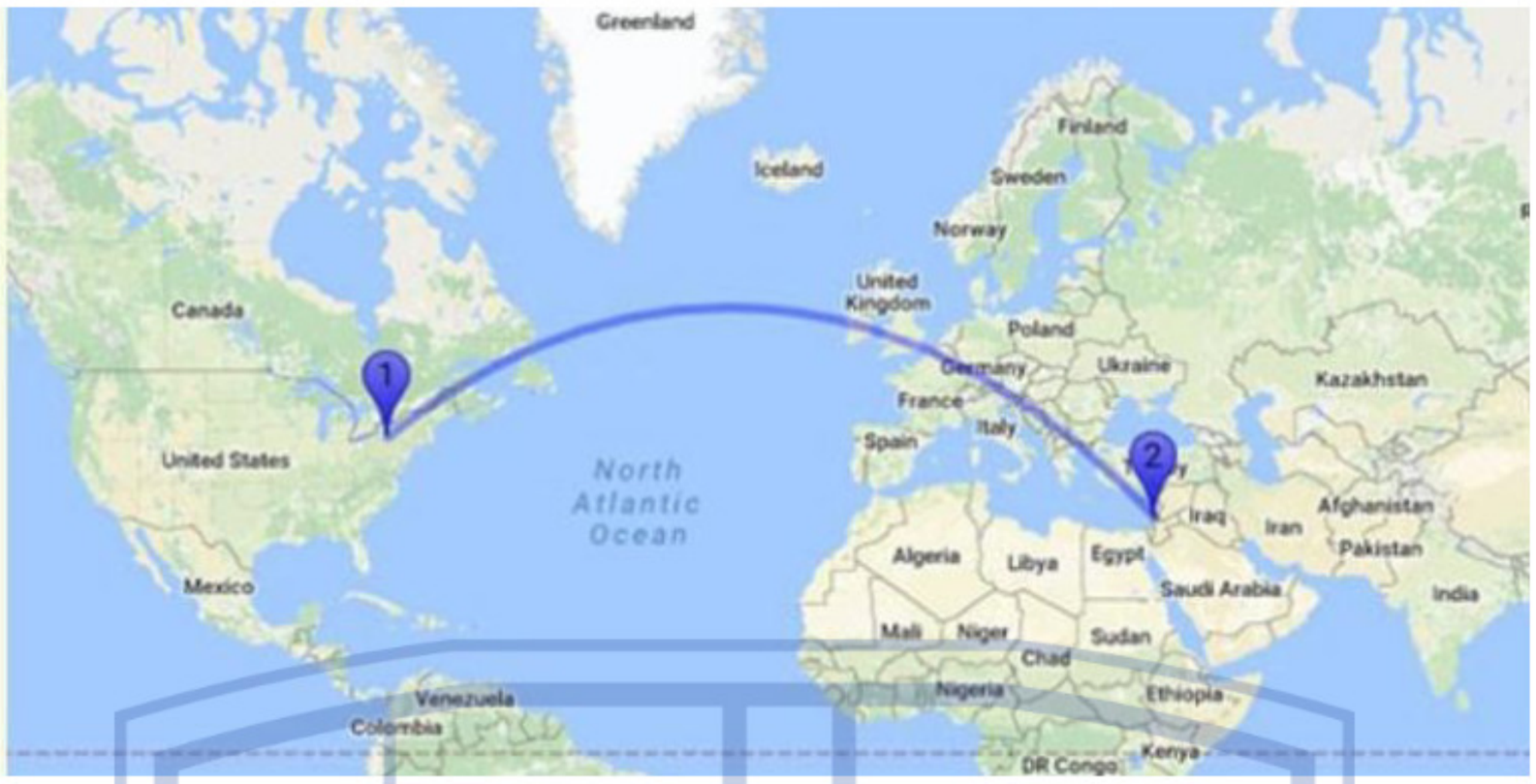
2.13.3 HTML5 *Geolocation* API

HTML5 *Geolocation* API diakses melalui objek *navigator.geolocation* pada *browser*. Melalui API ini, aplikasi web dapat meminta koordinat lokasi perangkat pengguna dalam bentuk nilai lintang dan bujur, serta informasi tambahan seperti tingkat akurasi lokasi. Proses pengambilan lokasi dilakukan setelah pengguna memberikan izin akses lokasi pada *browser*, sehingga fitur ini tetap memperhatikan aspek privasi pengguna. Pada sistem berbasis web, mekanisme ini penting karena aplikasi tidak perlu memasang perangkat tambahan untuk memperoleh lokasi pengguna, melainkan cukup memanfaatkan kemampuan lokasi yang tersedia pada perangkat seperti GPS, *Wi-Fi*, atau jaringan seluler [87] [88].

Dalam penerapan sistem absensi, HTML5 *Geolocation* API berfungsi sebagai komponen awal untuk memperoleh posisi guru saat melakukan presensi. Koordinat yang diperoleh dari *browser* kemudian dapat dibandingkan dengan koordinat sekolah yang telah ditentukan sebagai titik acuan. Apabila lokasi pengguna berada dalam jarak atau radius yang diperbolehkan, maka proses absensi dapat dilanjutkan ke tahap berikutnya. Sebaliknya, apabila lokasi berada di luar *area* yang ditentukan atau pengguna menolak izin akses lokasi, maka sistem dapat menolak proses absensi atau menampilkan peringatan. Dengan demikian, HTML5 *Geolocation* API mendukung validasi lokasi pada sistem absensi berbasis website agar kehadiran tidak hanya diverifikasi berdasarkan identitas wajah, tetapi juga berdasarkan keberadaan fisik pengguna di *area* sekolah [83] [88].

2.13.4 *Haversine*

Metode *Haversine* merupakan metode yang membutuhkan dua pasang koordinat, yaitu koordinat titik asal dan koordinat titik tujuan. Koordinat tersebut umumnya terdiri dari nilai lintang dan bujur yang terlebih dahulu dikonversi ke satuan radian sebelum dimasukkan ke dalam fungsi trigonometri. Perhitungan ini menghasilkan jarak antara dua titik berdasarkan kelengkungan permukaan bumi, sehingga lebih sesuai digunakan untuk menghitung jarak geografis dibandingkan perhitungan jarak biasa pada bidang datar [89] [90].



Gambar 2. 9 *Haversine Formula* [91]

$$d = 2rargsin\left(\sqrt{sin^2\left(\frac{lat2-lat1}{2}\right) + cos(lat2)\ cos(lat1)\ sin^2\left(\frac{lon2-lon1}{2}\right)}\right) \quad (23)$$

Di mana:

d= jarak (antara *origin* dan *destination*),

r= jari-jari Bumi,

$lat_1, lon_1 dan\ \ lat_2, lon_2$= koordinat (harus dalam radian saat dipakai pada fungsi trigonometri) [90]

2.13.5 *Geofencing*

Geofencing adalah metode pembentukan batas wilayah *virtual* pada suatu lokasi geografis yang digunakan untuk menentukan apakah pengguna berada di dalam atau di luar *area* tertentu. Batas wilayah tersebut dapat berbentuk radius melingkar dari satu titik pusat atau berbentuk poligon sesuai batas *area* yang ditentukan pada peta digital. Pada sistem berbasis lokasi, posisi pengguna yang diperoleh melalui *geolocation* dibandingkan dengan batas virtual tersebut, sehingga sistem dapat memberikan respons tertentu ketika pengguna berada di dalam, masuk, atau keluar dari *area geofence*, seperti mengizinkan akses, menolak akses, mengirim notifikasi, atau mencatat aktivitas pengguna [92].

Gambar 2. 10 *Geofencing Technology* [93]

2.14 Evaluasi Sistem

Evaluasi sistem dilakukan untuk menilai sejauh mana sistem pengenalan wajah mampu mengenali wajah pengguna secara tepat dalam berbagai kondisi pengujian. Pada sistem pengenalan wajah, evaluasi dilakukan untuk mengukur kemampuan sistem dalam mengenali wajah pengguna secara benar pada berbagai kondisi pengujian [94].

Evaluasi sistem diperlukan untuk menilai keandalan sistem absensi berbasis website dalam menerima pengguna yang sah dan menolak pengguna yang tidak sesuai. Evaluasi ini penting karena kesalahan dalam sistem biometrik dapat memengaruhi validitas data absensi, misalnya ketika wajah guru yang benar ditolak oleh sistem atau ketika wajah yang tidak sesuai justru diterima [94]. Evaluasi sistem pada penelitian ini dilakukan menggunakan *Confusion Matrix*. *Confusion matrix* merupakan tabel perbandingan antara hasil prediksi sistem dan kondisi sebenarnya.

Komponen-komponen *confusion matrix* terdiri dari:

a. *True Positive* (TP): data positif yang dikenali dengan benar
b. *True Negative* (TN): data negatif yang dikenali dengan benar
c. *False Positive* (FP): data negatif tetapi dikenali sebagai positif
d. *False Negative* (FN): data positif tetapi dikenali sebagai negative

Keempat parameter tersebut digunakan untuk membandingkan hasil klasifikasi model terhadap kondisi sebenarnya [95]

1. *Accuracy* adalah proporsi prediksi yang benar dari seluruh sampel.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (24)$$

2. *Precision* merupakan seluruh prediksi positif, berapa proporsi yang benar-benar positif (mengukur *false positives*).

$$Precision = \frac{TP}{TP+FP} \tag{25}$$

3. *Recall* merupakan seluruh kasus positif yang sebenarnya, berapa proporsi yang berhasil dideteksi (mengukur *false negatives*).

$$Recall = \frac{TP}{TP+FN} \tag{26}$$

4. *F1-Score* merupakan rata-rata harmonis dari *precision* dan *recall*; berguna ketika ingin menyeimbangkan *precision* & *recall* (terutama pada data tidak seimbang).

$$\mathrm{F1} = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{27}$$

2.15 *User Acceptance Testing* (UAT)

User Acceptance Testing (UAT) merupakan tingkat penerimaan pengguna terhadap suatu sistem berdasarkan pengalaman mereka setelah menggunakan sistem tersebut. Penerimaan pengguna menjadi salah satu aspek penting dalam pengembangan sistem karena keberhasilan sistem tidak hanya ditentukan oleh kemampuan teknis, tetapi juga oleh kemampuannya dalam memenuhi kebutuhan pengguna. Reimer dkk. menjelaskan bahwa pendekatan *user-centered design* melibatkan pengguna secara langsung untuk mengidentifikasi kebutuhan, mengevaluasi rancangan, serta meningkatkan pengalaman dan kepuasan pengguna terhadap sistem [96].

Evaluasi penerimaan pengguna dapat dilakukan melalui survei menggunakan kuesioner dengan skala *Likert* untuk mengukur tingkat persetujuan pengguna terhadap berbagai aspek sistem. Selain itu, Haindl dkk. menjelaskan bahwa setiap *stakeholder* memiliki peran dan aktivitas yang berbeda dalam menggunakan sistem sehingga karakteristik kualitas yang dinilai juga dapat berbeda sesuai dengan *role* dan *use case* masing-masing [97]. Dengan perhitungan sebagai berikut [98].

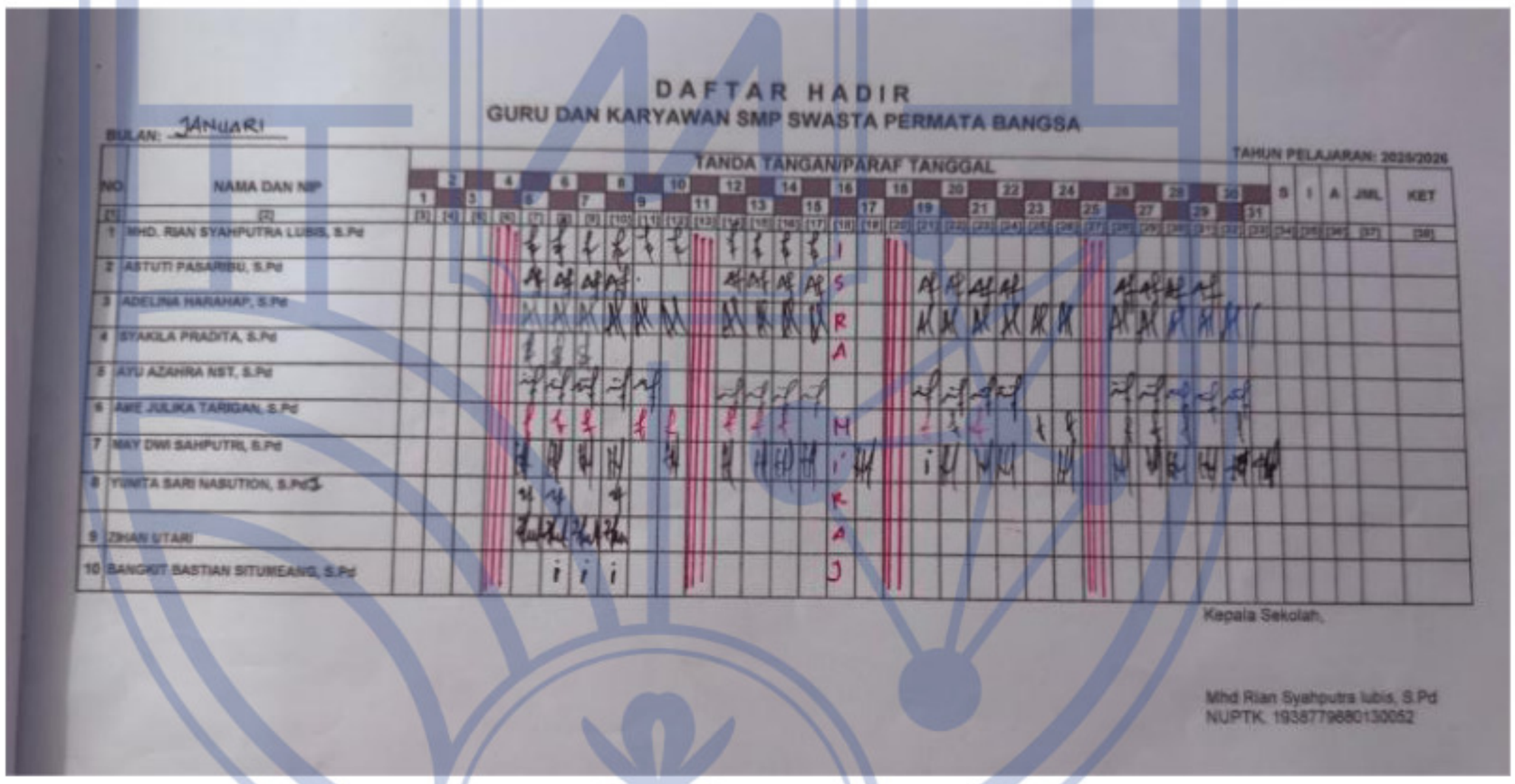
$$\begin{aligned}
&\text{Skor Ideal} = \text{Jumlah Responden} \times \text{Bobot Tertinggi} \times \text{Jumlah Pertanyaan} \\
&\text{Total Skor} = (\Sigma\ \text{STS} \times 1) + (\Sigma\ \text{TS} \times 2) + (\Sigma\ \text{N} \times 3) + (\Sigma\ \text{S} \times 4) + (\Sigma\ \text{SS} \times 5) \\
&\text{Persentase} = (\text{Total Skor} / \text{Skor Ideal}) \times 100\% \\
&\text{Mean} = \text{Total Skor} / \text{Total Item Jawaban}
\end{aligned} \tag{28}$$

2.16 Sistem Absensi

Sistem absensi adalah suatu sistem informasi berbasis teknologi yang dirancang untuk mencatat, menyimpan, dan mengelola data kehadiran tenaga pendidik secara otomatis dan terstruktur, dengan tujuan meningkatkan akurasi pencatatan, mengurangi kesalahan metode manual, serta mendukung *monitoring* dan evaluasi kehadiran secara efisien melalui pemanfaatan teknologi digital dan jaringan, sehingga mampu menunjang efektivitas pengelolaan administrasi dan kinerja di lingkungan pendidikan [99], [100].

Jenis-jenis Sistem Absensi sebagai Berikut:

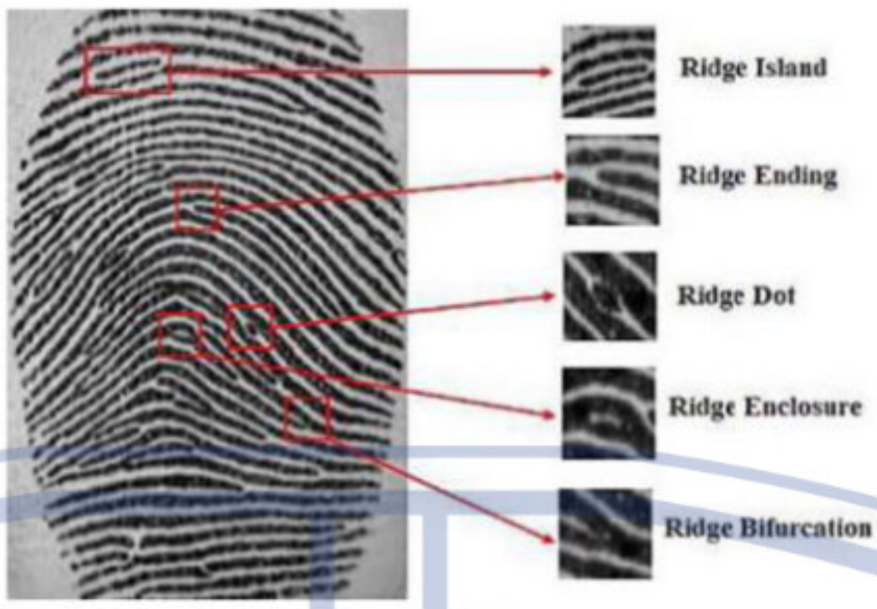
1. Sistem Absensi Tradisional (Manual): Sistem absensi manual yang dilakukan dengan memanggil nama siswa satu per satu memerlukan waktu yang cukup lama dalam pelaksanaannya. Selain itu, metode ini memiliki kelemahan berupa tingginya potensi terjadinya kesalahan pencatatan yang disebabkan oleh faktor manusia, sehingga data kehadiran yang dihasilkan sering kali tidak akurat [101].



Gambar 2. 11 Absensi Manual [102]

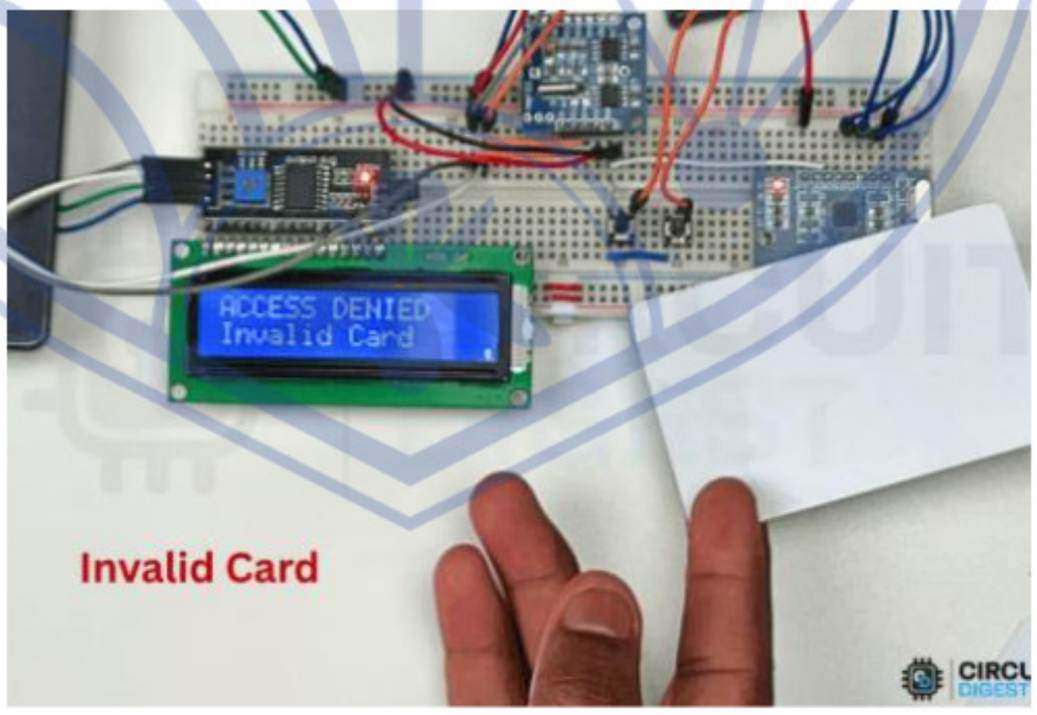
2. Sistem absensi berbasis biometrik sidik jari (*fingerprint*): memanfaatkan keunikan pola sidik jari setiap individu sebagai identitas autentik untuk proses verifikasi kehadiran. Sistem bekerja dengan menangkap citra pola *ridge* dan *valley* menggunakan sensor pemindai, kemudian mencocokkannya dengan *template* yang tersimpan pada basis data untuk menentukan validitas pengguna, sehingga mampu meminimalkan kecurangan seperti absensi titipan. Namun, penerapan metode ini memerlukan perangkat keras khusus berupa sensor sidik jari dan modul pemrosesan data yang berdampak pada biaya pengadaan serta pemeliharaan sistem. Selain itu, proses absensi yang dilakukan secara bergantian pada satu perangkat dapat

menimbulkan antrean pengguna dan mengurangi *fleksibilitas* penggunaan, terutama pada lingkungan dengan jumlah pengguna yang besar.



Gambar 2. 12 Pengenalan pola detail sidik jadi [103]

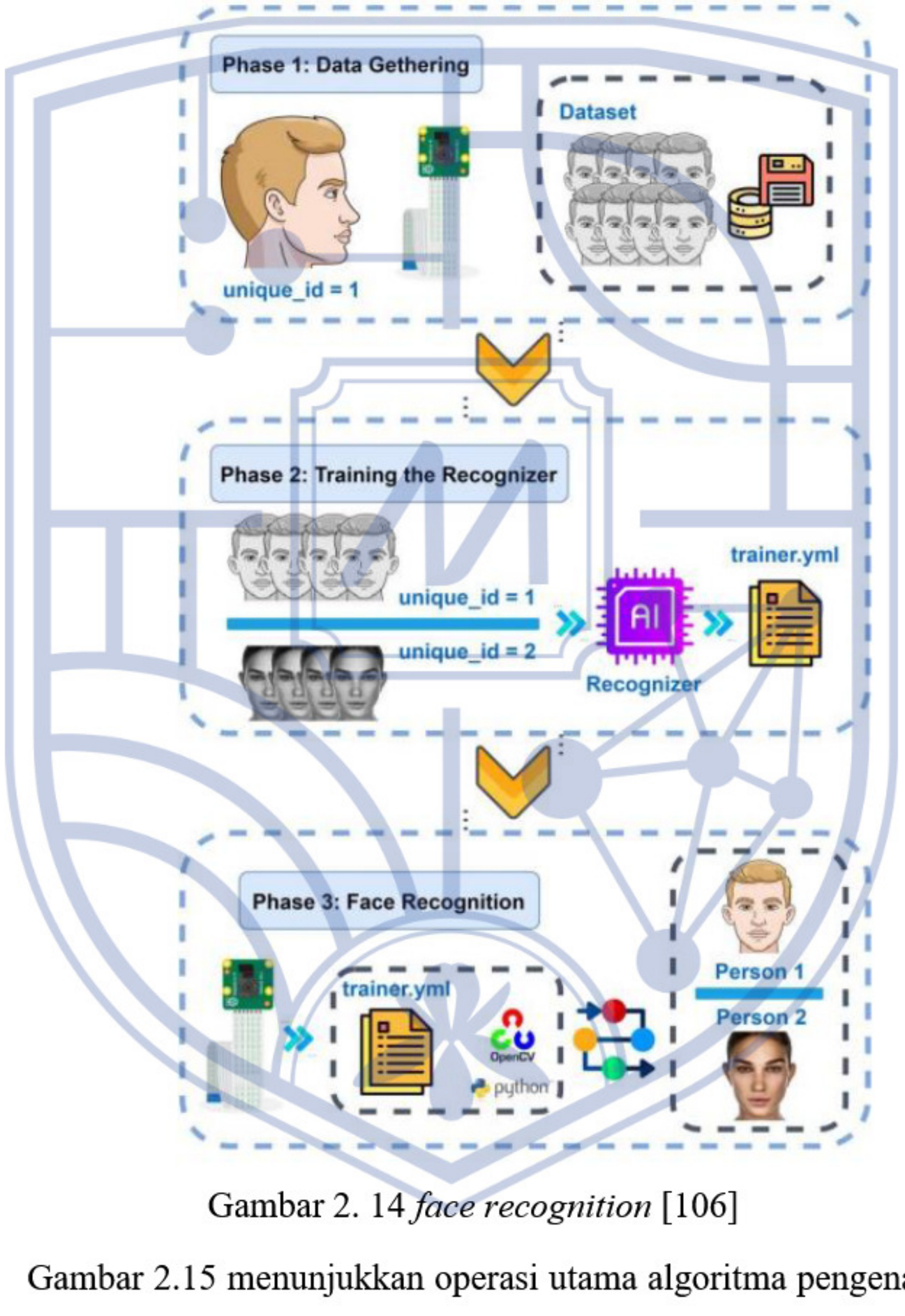
3. Sistem absensi berbasis kartu (*smart card*/RFID) merupakan bagian dari presensi elektronik yang memanfaatkan identifikasi otomatis untuk mencatat kehadiran secara cepat dan terstruktur. Pengguna cukup menempelkan kartu pada perangkat pembaca sehingga data kehadiran dapat tercatat secara otomatis dan langsung tersimpan dalam sistem informasi, yang membantu meningkatkan efisiensi serta akurasi pencatatan dibanding metode manual, terutama pada institusi dengan jumlah anggota yang besar. Implementasi sistem ini juga memungkinkan integrasi data secara digital, namun penerapannya memerlukan kesiapan infrastruktur, integrasi perangkat, serta pengelolaan keamanan data dan kesiapan pengguna agar sistem dapat berjalan optimal [5].



Gambar 2. 13 Card RFID [104]

4. Sistem manajemen absensi berbasis pengenalan wajah (*face recognition*) merupakan metode pencatatan kehadiran otomatis yang memanfaatkan analisis fitur wajah individu sebagai identitas pengguna. Pada sistem ini, kamera menangkap citra wajah kemudian dilakukan proses deteksi dan *alignment* wajah, dilanjutkan dengan

ekstraksi ciri menggunakan model *deep learning* untuk membentuk representasi wajah (*embedding*). Hasil fitur tersebut dibandingkan dengan data yang tersimpan pada basis data untuk menentukan kecocokan identitas, dan apabila sesuai maka kehadiran pengguna akan dicatat secara otomatis ke dalam sistem. Dengan mekanisme tersebut, proses identifikasi tidak lagi memerlukan interaksi fisik pengguna seperti penandatanganan atau pemindaian perangkat, sehingga pencatatan kehadiran menjadi lebih cepat dan akurat [105]



Gambar 2. 14 *face recognition* [106]

Gambar 2.15 menunjukkan operasi utama algoritma pengenalan wajah pada sistem absensi otomatis. Proses tersebut terdiri atas tiga tahapan utama, yaitu pengumpulan data, pelatihan model pengenal, dan pengenalan wajah. Pada tahap pengumpulan data, citra wajah pengguna direkam menggunakan kamera dan setiap data diberi identitas unik untuk membentuk *dataset* wajah. Selanjutnya, pada tahap pelatihan, *dataset* yang telah diberi label diproses, sehingga dihasilkan model pelatihan. Pada tahap pengenalan wajah, kamera menangkap citra wajah secara *real-*

time, kemudian sistem mendeteksi wajah, mengekstraksi ciri, dan membandingkannya dengan model yang telah dilatih sebelumnya. Jika hasil pencocokan sesuai, maka identitas pengguna dapat dikenali dan kehadiran dicatat secara otomatis [106].

2.17 Profil SMP Swasta Permata Bangsa

SMP Swasta Permata Bangsa merupakan sekolah menengah pertama swasta yang berlokasi di Jl. Beringin Gg. Belibis No. 4, Tembung, Kecamatan Percut Sei Tuan, Kabupaten Deli Serdang, Provinsi Sumatera Utara, dan berada di bawah naungan Yayasan Pendidikan Nusantara Angkasa Maritim. Sekolah ini telah terakreditasi B dan menerapkan Kurikulum Merdeka sesuai dengan kebijakan Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi. Berdasarkan data profil sekolah, SMP Swasta Permata Bangsa memiliki sekitar 57 peserta didik yang terdiri dari 31 siswa laki-laki dan 26 siswa perempuan, dengan 9 orang guru yang membimbing proses pembelajaran. Kegiatan pembelajaran diselenggarakan pada tiga tingkat kelas, yaitu kelas VII, VIII, dan IX, dengan jumlah peserta didik yang relatif tidak terlalu besar di setiap kelas sehingga memungkinkan proses pembelajaran berlangsung lebih kondusif serta interaksi antara guru dan peserta didik dapat berjalan secara lebih efektif [107].

Gambar 2. 15 SMPS Permata Bangsa [108]

Gambar 2. 16 Kondisi ruangan kelas [108]

Berdasarkan dokumentasi sekolah mitra, Gambar 2.15 dan Gambar 2.16 memperlihatkan tampak depan SMP Swasta Permata Bangsa sebagai identitas fisik lembaga, dan menunjukkan suasana kegiatan pembelajaran di dalam kelas yang mencerminkan lingkungan belajar yang aktif dan kondusif.