

**IMPLEMENTASI FINGERPRINT RECOGNITION PADA KEAMANAN FOLDER  
MENGUNAKAN ALGORITMA CONVOLUTIONAL  
NEURAL NETWORK**

**SKRIPSI**

**M. ADIB MANGARAJA**

**191402116**



**PROGRAM STUDI S1 TEKNOLOGI INFORMASI  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
2024**

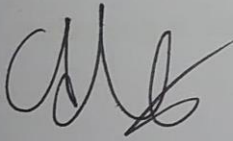
## PERNYATAAN

### IMPLEMENTASI FINGERPRINT RECOGNITION PADA KEAMANAN FOLDER MENGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK

## SKRIPSI

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 14 Juni 2024



M. Adib Mangaraja

191402116

## PERSETUJUAN

Judul : Implementasi Fingerprint Recognition Pada Keamanan Folder Menggunakan Algoritma Convolutional Neural Network

Kategori : Skripsi

Nama Mahasiswa : M. Adib Mangaraja

Nomor Induk Mahasiswa : 191402116

Program Studi : Sarjana (S1) Teknologi Informasi

Fakultas : Ilmu Komputer dan Teknologi Informasi  
Universitas Sumatera Utara

Medan, 14 Juni 2024

Komisi Pembimbing :

Pembimbing 2,



Dedy Arisandi, ST., M.Kom  
NIP. 197908312009121002

Pembimbing 1,



Dr. Erna Budhiarti Nababan, M.IT  
NIP. 196210262017042001

Diketahui/disetujui oleh  
Program Studi S1 Teknologi Informasi  
Ketua,



Dedy Arisandi, ST., M.Kom  
NIP. 197908312009121002

## ABSTRAK

Data, yang terdiri dari teks, gambar, suara, dan simbol, memenuhi dunia digital, terbagi menjadi data publik dan data pribadi. Data publik tidak berbahaya jika diketahui, sementara akses ilegal ke data pribadi dapat mengakibatkan kerugian besar. Metode keamanan data tradisional, seperti kata sandi, terbukti kurang efektif dengan kemampuan komputasi yang semakin maju. Penelitian ini mengusulkan untuk memanfaatkan biometrik sidik jari, khususnya menggunakan algoritma Convolutional Neural Network, untuk meningkatkan keamanan data dan folder. Menggunakan sebuah dataset yang terdiri dari 740 sidik jari, bersumber dari 74 partisipan berusia 12 hingga 71 tahun, diproses dan augmentasi, menghasilkan 79.920 gambar sidik jari. Dengan menggunakan model Siamese, penelitian ini mencapai akurasi 90% dalam membedakan gambar sidik jari yang beragam. Untuk meningkatkan integritas data, digunakan algoritma Advanced Encryption Standard untuk mengenkripsi dan mendekripsi data yang diamankan. Hasil akhir dari penelitian ini adalah sebuah aplikasi desktop berbasis Windows, yang memberikan solusi kuat dalam lanskap perlindungan data yang terus berkembang, yang dirangkum dalam penelitian ringkas ini.

**Kata kunci:** Keamanan data, Biometrik Sidik Jari, Pengenalan Biometrik, *Convolutional Neural Network* (CNN), *Advanced Encryption Standard* (AES)

## **IMPLEMENTATION OF FINGERPRINT RECOGNITION IN FOLDER SECURITY USING CONVOLUTIONAL NEURAL NETWORK ALGORITHM**

### **ABSTRACT**

*Data, encompassing text, images, sounds, and symbols, pervades the digital realm, split into public and private spheres. Public data poses no harm when known, while illegal access to private data can result in severe consequences. Traditional data security methods, like passwords, prove less effective with advancing computing capabilities. This study proposes leveraging fingerprint biometrics, specifically utilizing the Convolutional Neural Network algorithm, for enhanced data and folder security. Using a dataset of 740 fingerprints, sourced from 74 participants aged 12 to 71 years, undergoes processing and augmentation, yielding 79,920 fingerprint images. Employing the Siamese model, the research achieves 90% accuracy in distinguishing diverse fingerprint images. To bolster data integrity, the Advanced Encryption Standard algorithm encrypts and decrypts the secured data. The outcome is a Windows-based desktop application, providing a robust solution in the evolving landscape of data protection, encapsulated within this succinct research.*

**Keywords:** *Data security, Fingerprint Biometrics, Biometric Recognition, Convolutional Neural Network (CNN), Advanced Encryption Standard (AES)*

## DAFTAR ISI

PERNYATAAN	i
PERSETUJUAN	ii
ABSTRAK	iii
<i>ABSTRACT</i>	iv
DAFTAR ISI	v
DAFTAR TABEL	viii
DAFTAR GAMBAR	ix
BAB I PENDAHULUAN	2
1.1 Latar Belakang	2
1.2 Rumusan Masalah	4
1.3 Batasan Masalah	4
1.4 Tujuan Penelitian	4
1.5 Manfaat Penelitian	4
1.6 Metodologi Penelitian	5
1.7 Sistematika Penulisan	6
BAB II LANDASAN TEORI	7
2.1 Pengertian Folder dan File	7
2.2 Keamanan Folder	7
2.3 Sidik Jari	8
2.4 <i>Fingerprint Recognition</i>	9
2.5 Kriptografi	12
2.6 <i>Advanced Encryption Standard</i>	13
2.7 <i>Convolutional Neural Network</i>	14
2.8 Model	14
2.9 <i>TensorFlow</i>	14
2.10 <i>TensorFlow.js</i>	15
2.11 <i>Confusion Matrix</i>	15
2.11.1 Accuracy	16
2.11.2 Precision	17
2.11.3 Recall	17
2.11.4 Specificity	17
2.11.5 F1 Score	17
2.12 Penelitian Terdahulu	18

2.13 Perbedaan Penelitian	20
<b>BAB III ANALISIS DAN PERANCANGAN SISTEM</b>	<b>21</b>
3.1 Data yang akan digunakan	21
3.2 Analisis Sistem	21
3.2.1 Pemrosesan Awal	22
3.2.2 Augmentasi	25
3.2.3 Dataset Splitting dan Labeling	26
3.2.4 Arsitektur Model	29
3.2.5 Pelatihan Model	42
3.2.6 Uji Coba Model	42
3.2.7 Implementasi Sistem	43
3.3 Perancangan Antarmuka Sistem	45
3.3.1 Splash Screen	45
3.3.2 Tab Enkripsi	46
3.3.3 Dialog Registrasi Sidik Jari	47
3.3.4 Tab dekripsi	48
3.3.5 Dialog dekripsi	49
3.3.6 Tab Pengaturan	50
<b>BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM</b>	<b>52</b>
4.1 Implementasi Sistem	52
4.1.1 Perangkat Keras	52
4.1.2 Perangkat Lunak	52
4.1.3 Data Pendukung	53
4.2 Implementasi Rancangan Antarmuka	54
4.2.1 Splash Screen	54
4.2.2 Tab Enkripsi	55
4.2.3 Dialog Registrasi Sidik Jari	57
4.2.4 Tab Dekripsi	58
4.2.5 Dialog Dekripsi	59
4.2.6 Tab Pengaturan	60
4.3 Prosedur Operasional	61
4.4 Pelatihan Sistem	62
4.5 Pengujian Sistem	66
<b>BAB V KESIMPULAN DAN SARAN</b>	<b>70</b>

5.1 Kesimpulan	70
5.2 Saran	70
DAFTAR PUSTAKA	72



**DAFTAR TABEL**

<b>Tabel 2.1</b> Penelitian Terdahulu	18
<b>Tabel 4.1</b> Hasil pembagian data	63
<b>Tabel 4.2</b> Hasil uji coba pelatihan sistem	63
<b>Tabel 4.3</b> Hasil lengkap uji coba pelatihan ke-7	64
<b>Tabel 4.4</b> Beberapa contoh hasil pengujian sistem	67
<b>Tabel 4.5</b> <i>Confusion Matrix</i> dari hasil pengujian sistem	68

## DAFTAR GAMBAR

<b>Gambar 2.1</b> <i>Ridges</i> dan <i>Valleys</i> pada sidik jari	8
<b>Gambar 2.2</b> Arch, Loop, dan Whorl	9
<b>Gambar 2.3</b> Citra sidik jari dalam bentuk biner	11
<b>Gambar 2.4</b> Visualisasi model <i>Convolutional Neural Network</i>	14
<b>Gambar 3.1</b> Arsitektur Umum	21
<b>Gambar 3.2</b> Alur kerja pemrosesan awal citra	22
<b>Gambar 3.3</b> Citra mentah dari perangkat keras DigitalPersona U.are.U 4500	22
<b>Gambar 3.4</b> Citra setelah proses <i>cropping</i> dan <i>thresholding</i>	23
<b>Gambar 3.5</b> Contoh proses <i>resizing</i> citra	24
<b>Gambar 3.6</b> Alur kerja augmentasi citra	25
<b>Gambar 3.7</b> Augmentasi gambar berdasarkan 1 sidik jari	26
<b>Gambar 3.8</b> Alur kerja splitting dan labeling dataset	26
<b>Gambar 3.9</b> Contoh nama file dari subjek nomor 0 ibu jari kiri	27
<b>Gambar 3.10</b> Contoh nama file dari subjek 0 ibu jari kiri pengambilan pertama	28
<b>Gambar 3.11</b> Arsitektur model	29
<b>Gambar 3.12</b> Visualisasi Arsitektur Model Fitur	30
<b>Gambar 3.13</b> Visualisasi Arsitektur Model Klasifikasi	31
<b>Gambar 3.14</b> Visualisasi matriks input	32
<b>Gambar 3.15</b> Visualisasi pergerakan filter lapisan konvolusi	33
<b>Gambar 3.16</b> Visualisasi <i>max pooling</i> pada matriks fitur yang telah diberi <i>padding</i>	36
<b>Gambar 3.17</b> Visualisasi pengubahan matriks multi dimensi menjadi vektor satu dimensi	38
<b>Gambar 3.18</b> Alur kerja aplikasi	44
<b>Gambar 3.19</b> Rancangan tampilan <i>splash screen</i>	45
<b>Gambar 3.20</b> Rancangan tampilan tab enkripsi	47
<b>Gambar 3.21</b> Rancangan tampilan dialog registrasi sidik jari	48
<b>Gambar 3.22</b> Rancangan tampilan tab dekripsi	49
<b>Gambar 3.23</b> Rancangan tampilan dialog dekripsi	50
<b>Gambar 3.24</b> Rancangan tampilan tab pengaturan	51
<b>Gambar 3.25</b> Rancangan tampilan dialog informasi	51
<b>Gambar 4.1</b> Contoh dari kumpulan citra sidik jari yang telah dilakukan pemrosesan awal	53
<b>Gambar 4.2</b> Contoh dari kumpulan citra sidik teraugmentasi	54
<b>Gambar 4.3</b> Tampilan dari halaman <i>splash screen</i>	55
<b>Gambar 4.4</b> Tampilan dari halaman tab enkripsi	55
<b>Gambar 4.5</b> <i>Tooltip</i> ketika kursor diarahkan pada tombol non aktif “Pilih Folder”	56
<b>Gambar 4.6</b> <i>List</i> pilihan sidik jari ketika tidak ada sidik jari terdaftar didalam database	56
<b>Gambar 4.7</b> Konten dari tab enkripsi ketika enkripsi dimulai	57
<b>Gambar 4.8</b> Tampilan dialog registrasi sidik jari	57
<b>Gambar 4.9</b> Tampilan dari halaman tab dekripsi	58
<b>Gambar 4.10</b> <i>Tooltip</i> ketika kursor diarahkan pada tombol non aktif “Dekripsi”	58
<b>Gambar 4.11</b> Konten dari tab dekripsi ketika file <i>cipher</i> telah dipilih	59
<b>Gambar 4.12</b> Tampilan dialog dekripsi	60
<b>Gambar 4.13</b> Tampilan dari halaman tab pengaturan	60
<b>Gambar 4.14</b> Tampilan dialog informasi	61

<b>Gambar 4.15</b> Visualisasi akurasi dari hasil pelatihan sistem	65
<b>Gambar 4.16</b> Visualisasi loss dari hasil pelatihan sistem	65

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Pada zaman teknologi saat ini, informasi dan data dapat diartikan dengan harta dan memiliki nilai, tergantung pada apa yang dikandung di dalamnya. Makin penting informasi atau data tersebut, maka makin tinggi pula nilai yang dimilikinya. Studi kasus menunjukkan bahwa kebutuhan untuk mengamankan data makin meningkat. Begitu juga dengan metode untuk mengamankan data tersebut. Dengan dikemukakannya teknologi-teknologi baru seperti Cloud Storage dan Remote Access, mengamankan data menjadi hal yang menantang untuk dapat dilakukan dengan baik dan benar.

Keamanan folder merupakan teknik yang dilakukan untuk mengamankan data dan informasi yang terkandung di dalam sebuah folder dari berbagai ancaman seperti akses tanpa izin, modifikasi ilegal, ataupun penghapusan. Dalam era digital sekarang, di mana banyak informasi penting dan sensitif tersimpan di dalam perangkat elektronik, dan perangkat tersebut dapat terhubung ke internet, memastikan bahwa informasi tersebut tersimpan dan terjaga dengan nyaman adalah hal yang sangat penting. Seperti yang telah disebutkan, salah satu tujuan dalam mengamankan folder adalah mencegah pengguna tanpa izin dalam mendapatkan akses ke informasi dan data yang penting dan sensitif, sebagaimana hal tersebut dapat mengakibatkan hal-hal yang tidak diinginkan, seperti pencurian data, kehilangan reputasi, kerugian finansial, atau bahkan pencurian identitas. Untuk mencegah hal-hal tersebut, pengamanan folder pada umumnya dilakukan dengan berbagai macam teknik pengamanan seperti enkripsi, pengamanan kata sandi, kontrol akses, dan firewall. Pada umumnya, teknik-teknik yang telah dipaparkan efektif dalam melakukan perannya. Namun, teknik-teknik tersebut tidak selalu sempurna, dan hacker dan cybercriminal selalu menemukan cara baru untuk melangkahi pengamanan tersebut. Dengan demikian, dibutuhkan cara-cara baru untuk dapat mengamankan data dan informasi, di mana kedua hal tersebut dalam konteks digital selalu tersimpan di dalam folder.

Untuk menjawab masalah dalam mengamankan folder, penulis menemukan beberapa penelitian terdahulu. Dalam beberapa penelitian tersebut, salah satu penelitian yang telah dilakukan adalah “Implementasi Fingerprint Recognition Pada Keamanan Folder” (Dermawan, 2016). Penelitian tersebut menerapkan pengamanan folder menggunakan fingerprint recognition dan kriptografi, dan menerapkan algoritma Euclidean distance untuk pencocokkan citra dari pengamanan fingerprint recognition. Dengan akurasi yang didapatkan menggunakan algoritma tersebut mencapai 80,9% , penelitian tersebut dapat dikatakan telah mencapai hasil yang diinginkan. Namun, dalam konteks keamanan, akurasi pencocokkan citra adalah hal yang sangat penting, sehingga dibutuhkan tingkat akurasi yang sangat tinggi untuk dapat mengamankan hal yang ingin diamankan. Dalam penelitian ini, penulis akan menggunakan langkah yang sama dengan penelitian terdahulu, dengan pengecualian, untuk meningkatkan akurasi dan memberikan hasil yang diinginkan, penulis kali ini akan menggunakan algoritma Convolutional Neural Network (CNN) untuk mencocokkan citra pada pengamanan fingerprint recognition.

Adapun beberapa penelitian terdahulu pada CNN dan fingerprint recognition seperti yang dilakukan oleh (Y. Amruth, et al 2020). Penelitian tersebut memfokuskan pada pencocokkan citra sidik jari dan tanda tangan menggunakan CNN dengan 3 jenis arsitektur, yaitu VGG-16, AlexNet, dan model yang telah dikustomisasi, dan menggunakan ANN sebagai komparasi terhadap CNN. Hasil yang didapatkan pada rekognisi sidik jari sangat tinggi menggunakan CNN dengan arsitektur VGG-16 dan AlexNet, yaitu 91.5% akurasi menggunakan arsitektur AlexNet, dan 96.5% akurasi menggunakan VGG-16. Penelitian terdahulu yang serupa menggunakan CNN untuk mencocokkan citra sidik jari yang dilakukan oleh (B. Behnam, V. Hadi 2019) di mana penelitian ini menggunakan CNN untuk melakukan pencocokkan citra sidik jari menggunakan sebuah arsitektur buatan sendiri yang diberi nama ConvNet. Pada penelitian tersebut, didapatkan hasil berupa 17.5% Equal Error Rate (EER), di mana nilai tersebut mengindikasikan nilai bersilang di mana sistem mengrekognisi sidik jari yang salah, dan tidak mengrekognisi sidik jari yang benar, maka nilai tersebut lebih rendah lebih baik.

## 1.2 Rumusan Masalah

Dalam zaman digital sekarang ini, sebuah folder dapat berisikan file-file, dimana file-file tersebut dapat berisikan data dan informasi yang bersifat pribadi. Bocornya data dan informasi tersebut dapat mengakibatkan berbagai macam hal yang tidak di inginkan, seperti pencurian data, kehilangan reputasi, kerugian finansial, atau bahkan pencurian identitas. Maka dari itu, dibutuhkan metode untuk mengamankan folder yang efektif dan efisien.

## 1.3 Batasan Masalah

Agar penelitian ini tetap di arah yang benar, maka diperlukan batasan masalah. Adapun batasan masalah dalam penelitian ini adalah sebagai berikut :

1. Data sidik jari yang akan digunakan diambil dari beberapa sampel acak
2. Data sidik jari diambil menggunakan perangkat *fingerprint scanner*
3. Sidik jari yang akan diuji akan disimpan dalam bentuk gambar
4. Aplikasi hanya akan digunakan untuk mengamankan folder, mencegah akses dari pihak yang tidak di inginkan

## 1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah untuk meningkatkan akurasi dan keamanan folder menggunakan metode pengenalan biometrik sidik jari, sehingga tidak dapat diakses oleh pihak yang tidak di inginkan.

## 1.5 Manfaat Penelitian

Adapun manfaat penelitian yang diperoleh dari penelitian ini adalah sebagai berikut :

1. Dihasilkan suatu aplikasi yang dapat menjaga keamanan *folder* dari akses dari pihak yang tidak di inginkan, menggunakan pengenalan biometrik sidik jari
2. Dapat dijadikan bahan referensi mengenai *Convolutional Neural Network* pada penelitian selanjutnya
3. Dapat dijadikan bahan referensi mengenai *Advanced Encryption Standard* (AES) pada penelitian selanjutnya

## 1.6 Metodologi Penelitian

Tahapan-tahapan yang akan dilakukan dalam penelitian ini adalah sebagai berikut :

1. Studi Literatur

Pada tahap ini dilakukan studi literatur dengan mengumpulkan bahan referensi dari jurnal, artikel, buku, *website*, dan sumber lainnya mengenai *fingerprint recognition*, *image processing*, *Advanced Encryption Standard*, dan *Convolutional Neural Network*.

2. Analisis Permasalahan

Pada tahap ini dilakukan analisis terhadap tahapan yang telah dilakukan sebelumnya untuk mendapatkan pemahaman mengenai algoritma *Convolutional Neural Network* pada pengenalan pola-pola sidik jari, dan *Advanced Encryption Standard* untuk digunakan pada keamanan folder yang akan di implementasikan dalam penelitian ini.

3. Perancangan Sistem

Pada tahap ini dilakukan proses perancangan sistem berdasarkan analisis permasalahan yang telah dilakukan sebelumnya, seperti perancangan logika dan grafis antarmuka.

4. Implementasi

Pada tahap ini dilakukan implementasi aplikasi atau perangkat lunak berdasarkan proses perancangan sistem yang telah dilakukan sebelumnya.

5. Pengujian Sistem

Pada tahap ini dilakukan pengujian terhadap sistem yang telah dibangun untuk mengetahui tingkat akurasi yang didapatkan dari implementasi *fingerprint recognition* menggunakan algoritma *Convolutional Neural Network*, dan enkripsi data menggunakan *Advanced Encryption Standard*.

6. Dokumentasi dan Penyusunan Laporan

Pada tahap ini dilakukan dokumentasi dan penyusunan laporan yang menjabarkan hasil dari penelitian yang telah dilakukan.

### **1.7 Sistematika Penulisan**

Sistem penulisan skripsi ini terdiri dari lima bagian, yaitu :

#### **BAB I : Pendahuluan**

Bab ini berisi tentang latar belakang, rumusan masalah, Batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

#### **BAB II : Landasan Teori**

Bab ini berisi tentang penjelasan dari teori-teori yang relevan dengan sistem dan permasalahan yang diungkap dalam penelitian ini.

#### **BAB III : Analisis dan Perancangan Sistem**

Bab ini berisi tentang arsitektur umum dan metode yang diterapkan dalam pembuatan sistem.

#### **BAB IV : Implementasi dan Pengujian**

Bab ini berisi tentang implementasi atau penerapan sistem berdasarkan analisis dan rancangan yang telah dibuat. Bab ini juga berisi tentang tampilan sistem yang telah dibangun dan pengujian untuk mengetahui akurasi sistem dalam menerjemahkan dan menormalisasi bahasa isyarat.

#### **BAB V : Kesimpulan dan Saran**

Bab ini berisi tentang kesimpulan dari hasil penelitian dan saran untuk penelitian-penelitian selanjutnya.



## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Pengertian Folder dan File**

Folder (atau *directory*) merupakan sebuah kontainer virtual di dalam sistem file komputer. Folder digunakan untuk menyusun dan menyimpan file dan folder lainnya. Umumnya, folder ditandai dengan sebuah ikon atau nama di dalam *Graphical User Interface* (GUI) (Edwards, 2021).

File adalah kumpulan informasi atau data yang disimpan di dalam komputer atau perangkat elektronik lainnya. File dapat berupa teks dokumen, gambar, musik, video, program aplikasi, atau jenis data lainnya. Setiap file memiliki nama yang unik dan disimpan di dalam lokasi tertentu di dalam perangkat penyimpanan data seperti *Hard Disk Drive* (HDD). File disusun dan disimpan di dalam folder (Edwards, 2021).

#### **2.2 Keamanan Folder**

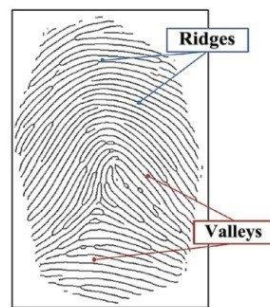
Keamanan folder merupakan teknik dan Tindakan yang dilakukan untuk mengatur siapa saja yang dapat mengakses, mengubah, atau menghapus konten di dalam sebuah folder. Dalam sistem operasi terkini, seperti Windows, macOS, dan Linux, keamanan folder dapat dikendalikan dengan memberi hak kepada pengguna secara individu atau grup. Hak-hak tersebut dapat berupa hak untuk melihat atau mengubah isi dari sebuah folder, dan juga hak untuk membuat atau menambahkan, menghapus, atau juga mengubah nama file di dalam sebuah folder (Brent, 2023).

Hak-hak tersebut dapat diberi kepada pengguna secara individu atau grup dengan tingkatan akses, umumnya dibagi menjadi tiga kategori: membaca (*Read*), menulis (*Write*), dan menjalankan (*Execute*). Hak membaca merupakan hak pengguna untuk membaca konten dari sebuah folder atau file. Hak menulis merupakan hak pengguna untuk membuat atau menambah, mengubah, atau menghapus file atau folder di dalam sebuah folder. Dan yang terakhir hak menjalankan merupakan hak pengguna untuk menjalankan aplikasi atau program di dalam folder (Brent, 2023).

### 2.3 Sidik Jari

Sidik jari (atau *Fingerprint*) adalah pola-pola guratan pada sidik jari yang bersifat unik kepada setiap individu. Pola-pola guratan tersebut terbentuk pada masa pertumbuhan janin, dan akan tetap memiliki bentuk dan pola yang sama selama seumur hidup. Sidik jari telah digunakan sebagai bentuk pengenalan identitas sejak dahulu, karena sifat uniknya yang membuatnya sulit untuk diubah atau diduplikat (Shawkat, Hazim, & Mossiah, 2018).

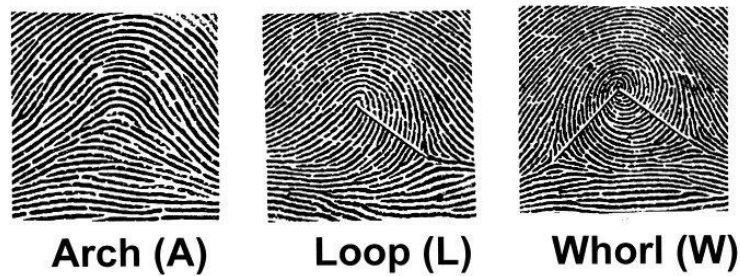
Sidik jari memiliki dua pola struktur, yaitu guratan (atau *ridges*) dan lembah (atau *valleys*), seperti pada Gambar 2.1.



**Gambar 2.1** *Ridges* dan *Valleys* pada sidik jari (Shawkat, Hazim, & Mossiah, 2018)

Sidik jari memiliki beberapa jenis klasifikasi, antara lain:

- 1) *Arch* – adalah jenis sidik jari sederhana, dengan guratan yang memasuki sidik jari dari satu sisi jari dan keluar di sisi yang lain. Sidik jari *Arch* tidak memiliki delta, di mana delta adalah guratan berpola segitiga yang dapat ditemukan di tengah pada jenis sidik jari lainnya, seperti pada Gambar 2.2.
- 2) *Loop* – adalah jenis sidik jari paling umum, dengan guratan yang memasuki sidik jari dari satu sisi jari dan melengkung berputar untuk keluar di sisi yang sama pada sisi yang dimasuki. Sidik jari *Loop* memiliki satu delta terletak di titik di mana guratan menyimpang dan menyatu lagi, seperti pada Gambar 2.2.
- 3) *Whorl* – adalah jenis sidik jari di mana guratannya membentuk pola lingkaran atau spiral, dan memiliki delta dua atau lebih. Sidik jari *Whorl* dapat dibagi menjadi empat sub-kategori, yaitu *plain whorls*, *central pocket whorls*, *double loop whorls*, dan *accidental whorls*. Sidik jari *Whorls* pada umumnya dapat dilihat pada Gambar 2.2 (Pal, Ray, & Bhattacharya, 2021).



**Gambar 2.3** Arch, Loop, dan Whorl (Pal, Ray, & Bhattacharya, 2021)

## **2.4 Fingerprint Recognition**

*Fingerprint recognition* (rekognisi sidik jari) telah digunakan secara umum sebagai tanda pengenal unik selama puluhan tahun, dikarenakan setiap orang memiliki sidik jari yang berbeda-beda, dan tidak pernah sama. Keunikan yang dimiliki sidik jari membuatnya ideal untuk dijadikan sebagai penanda biometrik untuk hal keamanan, sebagaimana mereka dapat digunakan untuk menandai dan mengenal individu dengan akurasi yang tinggi.

Rekognisi sidik jari bekerja dengan membuat representasi digital dari gambar pola-pola guratan pada ujung jari. Representasi digital tersebut kemudian dibandingkan dengan sidik jari yang terdaftar di dalam sebuah database untuk mengidentifikasi identitas dari pemilik sidik jari tersebut. Setiap orang memiliki sidik jari yang berbeda, membuatnya menjadi biometrik yang ideal untuk digunakan sebagai alat pengidentifikasi.

Adapun tahap-tahap yang dibutuhkan untuk melakukan rekognisi sidik jari adalah sebagai berikut:

- 1) Pengambilan sidik jari: Tahap pertama adalah untuk mendapatkan sidik jari. Pada umumnya, untuk mendapatkan sidik jari secara digital, sidik jari tersebut ditempelkan ke perangkat untuk memindai sidik jari, yang biasanya dikenal dengan nama *fingerprint scanner*. Sidik jari yang telah dipindai akan disimpan kedalam bentuk gambar.
- 2) Pemrosesan awal: Pada tahap ini, gambar dari sidik jari yang telah dipindai harus diproses terlebih dahulu. Hal ini dibutuhkan untuk meningkatkan kualitas dan memperjelas ciri khas yang dimiliki sidik jari tersebut. Pemrosesan dapat berupa penghapusan *noise*, memperjelas kontras, dan normalisasi ukuran dan orientasi gambar.

- 3) Pengenalan ciri khas: Pada tahap ini, ciri khas pada sidik jari yang telah diproses kemudian dikenali dan disimpan. Ciri khas yang dikenali berupa ujung, percabangan, dan penyimpangan guratan yang ada pada sidik jari. Detail-detail tersebut kemudian diubah kedalam bentuk kode *binary* untuk mewakili sidik jari secara keseluruhan.
- 4) Pendaftaran: Kode *binary* yang telah didapatkan dari sidik jari tersebut kemudian disimpan kedalam *database* beserta dengan informasi dari pemilik sidik jari tersebut. Dengan tahap ini, individu yang memiliki sidik jari ini akan dapat dikenali berdasarkan sidik jari yang telah terdaftar dan disimpan kedalam *database*.
- 5) *Matching*: Ketika seseorang ingin mengidentifikasikan dirinya, orang tersebut meletakkan jarinya pada *fingerprint scanner* untuk dipindai sidik jarinya. Setelah sidik jari tersebut didapatkan oleh sistem, di proses, dikenali ciri khasnya, dan didapatkan perwakilan kode *binary* nya, sistem kemudian melakukan perbandingan antara sidik jari yang diberikan dengan kumpulan dari sidik jari yang telah tersimpan di dalam *database*. Pengenalan tersebut pada umumnya menggunakan teknik penilaian berdasarkan kemiripan. Jika nilai kecocokkan tersebut berada di atas dari batas yang telah ditentukan, sidik jari tersebut dapat dinyatakan cocok dengan sidik jari yang telah disimpan di dalam *database*.
- 6) Keputusan: Setelah sidik jari dapat dikenali dengan sidik jari yang telah disimpan di dalam *database*, didapatkan informasi dari pemilik sidik jari tersebut. Berdasarkan informasi tersebut, dilakukan pengambilan keputusan untuk memberikan jawaban apakah jari yang diberikan memiliki hak terhadap apa yang di inginkan atau tidak.

Adapun tahap-tahap yang dilakukan pada pemrosesan awal adalah sebagai berikut:

- 1) *Resizing*

Pada tahap ini, citra sidik jari yang telah didapatkan dari *fingerprint scanner* atau sejenisnya, diubah ukurannya menjadi ukuran yang telah ditetapkan untuk seluruh citra yang telah dan akan didapatkan. Hal ini dilakukan untuk menyeragamkan ukuran dan kejelasan yang ada pada citra sidik jari.

### 2) *Cropping*

Pada tahap ini, citra sidik jari dipotong untuk menghilangkan fitur-fitur yang tidak diinginkan. Tahap ini dilakukan agar citra sidik jari hanya mengandung fitur-fitur yang perlu diserap, sehingga berkurangnya *noise* yang diserap oleh model pembelajaran mesin.

### 3) *Thresholding*

Pada tahap ini, citra sidik jari yang telah dinormalisasi dan diubah kedalam format warna *shades of gray* kemudian diubah menjadi citra hitam-putih atau biner, di mana hanya terdapat 2 warna di dalam citra tersebut, yaitu warna hitam dan warna putih. Hal ini dilakukan untuk mempermudah proses pengenalan pada citra sidik jari, karena dalam bentuk biner, citra sidik jari yang akan dikenal telah didapat ciri khas dan fitur istimewanya, yaitu pola bentuk guratan *Arch*, *Loop*, ataupun *Whorl* yang ada pada citra sidik jari tersebut. Pola bentuk guratan tersebut akan direpresentasikan pada citra sidik jari dengan satu warna yang sama, sedangkan hal selain pola bentuk guratan akan direpresentasikan pada citra sidik jari dengan warna yang berbeda dari warna pola bentuk guratan.



**Gambar 2.3** Citra sidik jari dalam bentuk biner (Bajahzar & Guedri, 2019)

Untuk mengubah citra sidik jari dari format *shades of gray* kedalam bentuk biner, dapat dilakukan dengan mengkalkulasikan apakah pixel tersebut diubah kedalam warna hitam atau putih pada setiap pixel yang ada pada citra sidik jari menggunakan rumus sebagai berikut:

$$B = (G \geq T) ? 1 : 0$$

Di mana:

- B = nilai binari, di mana nilai tersebut hanya boleh berupa angka 1 atau 0
- G = nilai warna abu-abu pada pixel
- T = nilai ambang batas

Pada umumnya, nilai warna pada pixel direpresentasikan dengan angka dari 0 sampai 255, di mana 255 adalah nilai tertinggi dengan artian warna tersebut paling dominan pada pixel tersebut. Untuk mengkalkulasikan nilai ambang batas yang akan digunakan dalam rumus tersebut, diambil nilai tengah dari 0 sampai 255, yaitu 127,5. Nilai tersebut pada umumnya dibulatkan menjadi 128, sehingga jika nilai G adalah lebih besar atau sama dengan 128, maka nilai B adalah 1. Sebaliknya, jika nilai G adalah lebih kecil dari 128, maka nilai B adalah 0. Kemudian, pixel tersebut diubah menjadi warna hitam atau warna putih berdasarkan nilai B, di mana pada umumnya jika nilai B adalah 1, maka pixel tersebut akan menjadi warna putih. Sebaliknya, jika nilai B adalah 0, maka pixel tersebut akan menjadi warna hitam (Lee, Yap, Hum, Goi, & Tee, 2018).

## 2.5 Kriptografi

Kriptografi adalah praktik dan ilmu dari teknik untuk mengamankan komunikasi dan data dari pihak yang tidak berwenang. Ilmu ini melibatkan penggunaan algoritma matematika dan prinsip untuk mengubah data asli (*plaintext*) menjadi data yang ter-enkripsi (*ciphertext*). Data yang telah di enkripsi tidak dapat dimengerti secara harfiah oleh manusia, sehingga melindungi data yang terkandung di dalamnya (de Alencar, 2022).

Untuk dapat mengubah *ciphertext* yang tidak dapat dimengerti secara harfiah tersebut, *ciphertext* harus di dekripsi terlebih dahulu menggunakan kunci yang sama untuk meng enkripsi data tersebut.

Secara matematis, proses umum enkripsi dapat dijelaskan menggunakan rumus sebagai berikut:

$$E(K, P) = C$$

Di mana:

- E = proses enkripsi
- K = kunci untuk proses enkripsi / dekripsi
- P = Data asli (*plaintext*)
- C = Data ter-enkripsi (*ciphertext*)

Dalam enkripsi, data asli dipermutasikan menggunakan sebuah kunci untuk menghasilkan data yang telah di enkripsi. Data yang telah di enkripsi mengandung data yang asli yang telah dipermutasi sehingga tidak dapat dibaca atau dimengerti secara harfiah oleh manusia. Secara matematis, proses umum dekripsi dapat dijelaskan menggunakan rumus sebagai berikut:

$$D(K, C) = P$$

Di mana:

- D = proses dekripsi
- K = kunci untuk proses enkripsi / dekripsi
- C = Data ter-enkripsi (*ciphertext*)
- P = Data asli (*plaintext*)

Untuk dapat melakukan dekripsi, dibutuhkan kunci yang sama dari proses enkripsi yang menghasilkan *ciphertext* yang akan di dekripsi. Setelah *ciphertext* di dekripsi, didapatkan *plaintext* yang dapat dibaca dan dimengerti secara harfiah oleh manusia.

## **2.6 Advanced Encryption Standard**

*Advanced Encryption Standard* (AES) adalah algoritma enkripsi berbasis kunci simetris yang dikembangkan oleh *The National Institute of Standards and Technology* (NIST) sebagai penerus algoritma terdahulu yaitu *Data Encryption Standard* (DES) dikarenakan semakin mudahnya algoritma tersebut dibuka paksa dengan metode *brutal force*. Berbeda dengan DES yang hanya menyediakan kunci 64 bit, terdapat 3 macam kunci AES berdasarkan panjang bit nya, yaitu 128 bit, 192 bit, dan 256 bit.

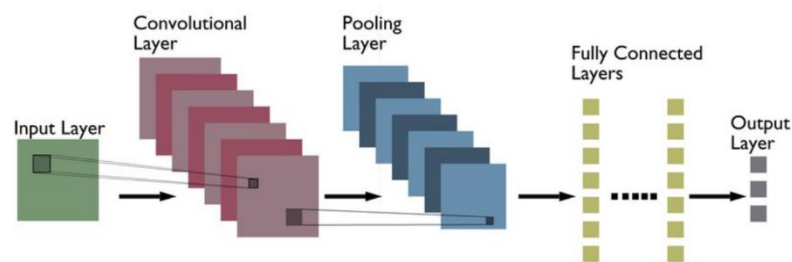
Masing-masing panjang kunci memiliki jumlah tahap proses yang berbeda juga, mulai dari 10 tahap untuk 128 bit, 12 tahap untuk 192 bit, dan 14 tahap untuk 256 bit. Semakin panjang kunci yang digunakan, semakin sulit enkripsi tersebut untuk dibobol, namun semakin besar pula jumlah daya proses yang diperlukan, sehingga pada umumnya, kunci 128 bit sudah lebih dari cukup untuk digunakan untuk keperluan mengamankan secara standar (Smid, 2021).

## 2.7 Convolutional Neural Network

*Convolutional Neural Network* (CNN) adalah salah satu model turunan dari arsitektur *Neural Network* (NN), di mana perbedaan utama antara CNN dengan NN adalah spesialisasi yang dirancang untuk CNN. CNN dirancang khusus untuk meniru saraf otak manusia, khususnya pada bagian depan otak manusia (*frontal lobe*), di mana pemrosesan stimulasi visual dilakukan pada bagian otak tersebut.

CNN terdiri dari beberapa lapisan, termasuk lapisan konvolusi, lapisan pooling, dan lapisan terhubung penuh (fully connected). Lapisan konvolusi menggunakan filter untuk mengekstrak fitur-fitur penting dari gambar. Filter ini melakukan operasi konvolusi, yaitu mengaplikasikan filter dengan bagian-bagian gambar untuk menghasilkan peta fitur.

Setelah lapisan konvolusi, lapisan pooling digunakan untuk mengurangi dimensi spasial dari peta fitur dengan mempertahankan informasi penting. Hal ini membantu dalam mengurangi jumlah parameter yang dibutuhkan dan mempercepat proses pelatihan (Gu, et al., 2018).



**Gambar 2.4** Visualisasi model *Convolutional Neural Network* (Mattera, Nele, & Paoella, 2023)

## 2.8 Model

Dalam konteks ilmu pembelajaran mesin, model merupakan sebuah representasi dari rancangan struktur komputasi yang digunakan untuk menangkap dan mempelajari pola, hubungan, atau kecenderungan dalam data.

Model adalah inti dari pembelajaran mesin, dimana model digunakan dalam *supervised learning* untuk dapat memberikan prediksi atau membuat keputusan berdasarkan data yang diberikan (What is an AI model? | IBM, t.thn.).

## 2.9 TensorFlow

*TensorFlow* adalah *framework machine learning* bersifat *open-source* yang dikembangkan oleh tim Google Brain. *TensorFlow* dirancang untuk memenuhi kebutuhan pengembangan, pelatihan, dan penggunaan model pembelajaran mesin. *Framework* ini dirancang menggunakan bahasa pemrograman Python, sehingga *framework* ini dapat memperdayakan *library* dan *framework* dalam Python yang terkenal untuk pembelajaran mesin dan ilmu matematika, seperti *NumPy* dan *Pandas* (Banoula, 2023).



### **2.10 TensorFlow.js**

Berbeda dengan *TensorFlow* pada bahasa pemrograman Python, *TensorFlow.js* adalah *TensorFlow* yang dapat digunakan dengan bahasa pemrograman JavaScript. (Chidi, 2020) *Framework* ini digunakan jika pengguna ingin mengembangkan atau menggunakan model didalam sistem lingkungan web atau Node.js. Dengan memanfaatkan sistem lingkungan Node.js, *TensorFlow.js* dapat digunakan di berbagai macam platform, seperti Windows, Mac OS X, Linux, ataupun mobile seperti Android dan iOS.

### **2.11 Confusion Matrix**

*Confusion Matrix* adalah metode umum dalam pembelajaran mesin dan statistika yang digunakan untuk mengevaluasi kemampuan klasifikasi dari model. *Confusion Matrix* dapat memperlihatkan kemampuan prediksi dan hasil nyata dari model secara detail dan jelas. Tujuan dari penggunaan metode ini adalah untuk mencari tahu akurasi model dalam melakukan prediksi.

*Confusion Matrix* terdiri dari 4 nilai, yaitu:

1) *True Positives* (TP)

TP adalah nilai dimana model memprediksi nilai positif secara akurat. Dalam kata lain, model memprediksi akan kehadiran sebuah kondisi, atau kondisi tersebut diprediksi akan atau telah terjadi, sehingga model memberikan nilai positif secara akurat.

2) *True Negatives* (TN)

TN adalah nilai dimana model memprediksi nilai negatif secara akurat. Kebalikan dari TP, nilai ini didapat dari model memprediksikan akan ketidak adaannya sebuah kondisi, sehingga model memberikan nilai negatif secara akurat.

3) *False Positives* (FP)

FP, atau *Type I Errors*, adalah nilai dimana model memprediksi nilai positif secara tidak akurat. Didalam FP, model memprediksikan akan kehadiran sebuah kondisi, dimana sebenarnya kondisi tersebut tidaklah ada, sehingga model memberikan nilai positif, dimana seharusnya model memberikan nilai negatif.

4) *False Negatives* (FN)

FN, atau juga disebut *Type II Errors*, adalah kebalikan dari FP. Model memprediksikan ketidak adaannya sebuah kondisi, namun sebenarnya kondisi tersebut ada, akan, atau telah terjadi, sehingga model memberikan nilai negatif, dimana seharusnya model memberikan nilai positif.

Berdasarkan 4 nilai tersebut, terdapat 5 jenis perhitungan untuk mengukur performa model, antara lain sebagai berikut:

### 2.11.1 Accuracy

Nilai *accuracy* adalah nilai dari semua prediksi akurat atau benar yang dilakukan oleh model. Untuk mendapatkan nilai ini, dapat digunakan 4 nilai sebelumnya dalam persamaan seperti berikut:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

### 2.11.2 Precision

Nilai *precision* adalah nilai dimana model memberikan prediksi positif secara akurat. Untuk mendapatkan nilai ini, dapat digunakan persamaan sebagai berikut:

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

### 2.11.3 Recall

Nilai *recall* adalah nilai dimana model memberikan nilai positif dibandingkan semua nilai positif yang sesungguhnya. Untuk mendapatkan nilai ini, dapat digunakan persamaan sebagai berikut:

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

### 2.11.4 Specificity

Nilai *Specificity* adalah nilai dimana model memberikan nilai negatif secara akurat. Untuk mendapatkan nilai ini, dapat digunakan persamaan sebagai berikut:

$$Specificity = \frac{TN}{TN + FP}$$

### 2.11.5 F1 Score

*F1 Score* adalah nilai akurasi dari model dari perspektif yang berbeda dari nilai *accuracy*. *F1 Score* dapat memberikan informasi dari akurasi model berdasarkan nilai *precision* dan nilai *recall*. *F1 Score* dapat memberikan nilai akurasi yang lebih baik jika dataset yang digunakan tidak seimbang. Sebagai contoh, sebuah dataset dapat dikatakan tidak seimbang jika jumlah data dari satu jenis kelas lebih banyak atau lebih sedikit dari jenis kelas yang lainnya. Untuk mendapatkan nilai, dapat digunakan persamaan sebagai berikut:

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

(Narkhede, 2018)

## 2.12 Penelitian Terdahulu

Adapun beberapa penelitian terdahulu pada CNN dan fingerprint recognition seperti “*Fingerprint and Signature Authentication System using CNN*” oleh Y. Amruth, et al (2020). Penelitian tersebut memfokuskan pada pencocokkan citra sidik jari dan tanda tangan menggunakan CNN dengan 3 jenis arsitektur, yaitu VGG-16, AlexNet, dan model yang telah dikustomisasi, dan menggunakan ANN sebagai komparasi terhadap CNN. Hasil yang didapatkan pada rekognisi sidik jari sangat tinggi menggunakan CNN dengan arsitektur VGG-16 dan AlexNet, yaitu 91.5% akurasi menggunakan arsitektur AlexNet, dan 96.5% akurasi menggunakan VGG-16.

Penelitian terdahulu yang serupa menggunakan CNN untuk mencocokkan citra sidik jari adalah “*End to End Fingerprint Verification Based on Convolutional Neural Network*” oleh B. Behnam, V. Hadi (2019) di mana penelitian ini menggunakan CNN untuk melakukan pencocokkan citra sidik jari menggunakan sebuah arsitektur buatan sendiri yang diberi nama ConvNet. Pada penelitian tersebut, didapatkan hasil berupa 17.5% Equal Error Rate (EER), di mana nilai tersebut mengindikasikan nilai bersilang di mana sistem mengrekognisi sidik jari yang salah, dan tidak mengrekognisi sidik jari yang benar, maka nilai tersebut lebih rendah lebih baik.

**Tabel 2.1** Penelitian Terdahulu

No.	Penulis	Judul	Tahun	Hasil
1.	Dermawan, D.R.	Implementasi Fingerprint Recognition Pada Keamanan Folder	2016	Folder berhasil dikunci, tingkat akurasi mencapai 80.9% menggunakan Euclidean Distance.
2.	Shehu, Y. Isah, et al	Detailed Identification of Fingerprints using Convolutional Neural Networks	2018	Penelitian berhasil mencapai tingkat akurasi sebesar 93.5% pada pengklasifikasian tangan.

**Tabel 2.1** Penelitian Terdahulu (Lanjutan)

No.	Penulis	Judul	Tahun	Hasil
3.	Bakhshi, B., et al	End to End Fingerprint Verification Based on Convolutional Neural Network	2019	Penelitian berhasil tingkat ERR 17.5%, dimana nilai tersebut menandakan tingkat kesalahan dalam pengklasifikasian, lebih rendah lebih baik.
4.	Amruth, Y., et al	Fingerprint and Signature Authentication System using CNN	2020	Penelitian mencapai tingkat akurasi setinggi 96.5% menggunakan CNN VGG-16, dan 91.5% menggunakan CNN AlexNet pada klasifikasi sidik jari.
5.	Khan, A.I.	LAFIN: A Convolutional Neural Network-based Technique for Singular Point Extraction and Classification of Latent Fingerprints	2018	Penelitian menggunakan citra sidik jari dengan kualitas buruk, mencapai akurasi maksimum 88.1%.
6.	Pandya, B., et al	Fingerprint Classification using a Deep Convolutional Neural Network	2018	Penelitian mencapai akurasi setinggi 98.21% menggunakan dataset yang sama dengan penelitian terdahulu yang menggunakan algoritma k-NN dengan akurasi 77%

**Tabel 2.1** Penelitian Terdahulu (Lanjutan)

7.	T.K, A.K, et al	Convolutional Networks for Fingerprint Liveness Detection System	Neural 2019	Penelitian berpusat mengklasifikasi sidik jari asli dan palsu dan mencapai akurasi maksimum 95%.
----	-----------------	--	----------------	--

### 2.13 Perbedaan Penelitian

Adapun perbedaan penelitian ini dari penelitian terdahulu seperti penelitian yang dilakukan oleh (Dermawan, 2016) adalah penelitian sebelumnya menggunakan algoritma Euclidean Distance untuk melakukan pencocokkan dalam rekognisi sidik jari, sedangkan penelitian ini menggunakan algoritma CNN. Penelitian tersebut juga menggunakan algoritma DES untuk mengamankan folder, sedangkan penelitian ini sudah menggunakan algoritma AES dengan panjang kunci 256-bit.

Penelitian yang dilakukan oleh (Amruth, Gopinatha, Gowtham, Kiran, & Harshalatha, 2020) menggunakan CNN VGG-16 dan CNN AlexNet, yang artinya penelitian tersebut menggunakan model *pre-trained*, sedangkan penelitian ini menggunakan model yang dilatih dari 0. Penelitian tersebut juga mencakup rekognisi tanda tangan, sedangkan penelitian ini hanya mencakup rekognisi sidik jari saja.

Penelitian terdahulu yang dilakukan oleh (Shehu, Garcia, & Palade, 2018) menggunakan dataset publik yang bernama Sokoto Coventry Fingerprint Dataset (SOCOFing) dengan jumlah partisipan sebesar 600 dari Afrika, sedangkan penelitian ini mengumpulkan dan membuat dataset dari 74 partisipan dari Indonesia.

## BAB III

### ANALISIS DAN PERANCANGAN SISTEM

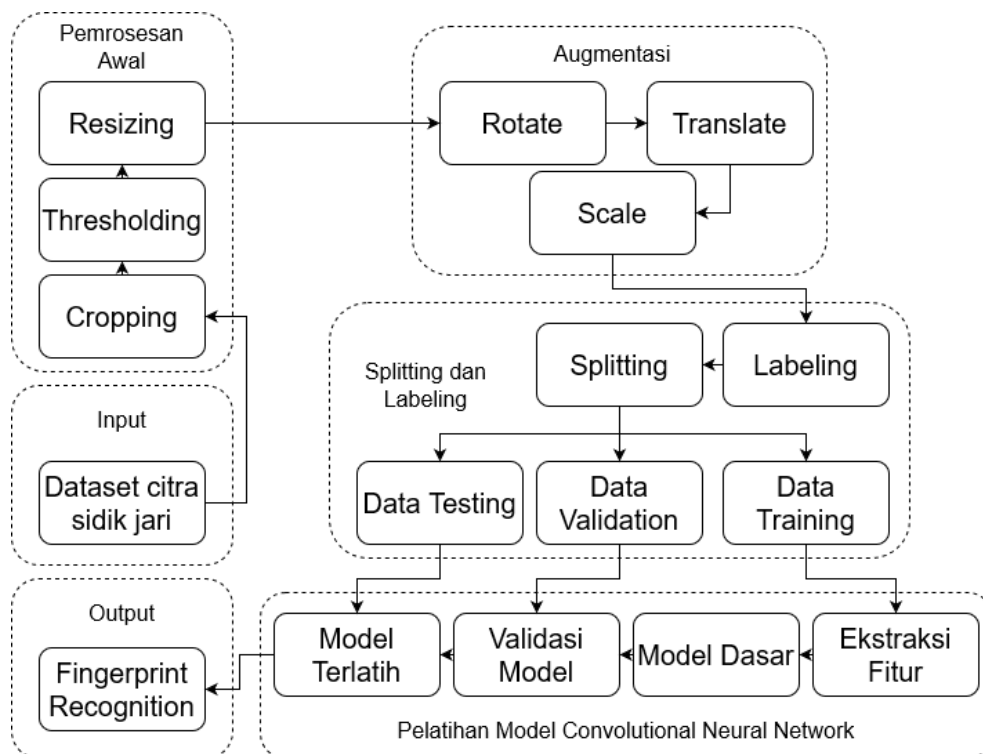
#### 3.1 Data yang akan digunakan

Data yang akan digunakan diambil dari 74 responden, laki-laki dan perempuan, dengan jarak usia dari 12 tahun sampai 71 tahun. Setiap responden diambil kesepuluh jari tangan, dan setiap jari diambil citranya tiga kali. Pengambilan citra menggunakan perangkat keras DigitalPersona U.are.U 4500, dengan resolusi 512ppi dan dimensi citra yang didapat berukuran lebar 357px dan tinggi 392px.

#### 3.2 Analisis Sistem

Penelitian ini akan menggunakan algoritma CNN didalam model yang akan dilatih untuk dapat mengenal atau membedakan sidik jari. Terdapat 6 tahap yang akan dilakukan dalam penelitian ini, yaitu pemrosesan awal data, augmentasi data, *splitting* dan *labeling* data, pelatihan model, uji coba model, dan implementasi sistem. Subbab ini akan membahas ke 6 tahap yang telah disebutkan secara teliti.

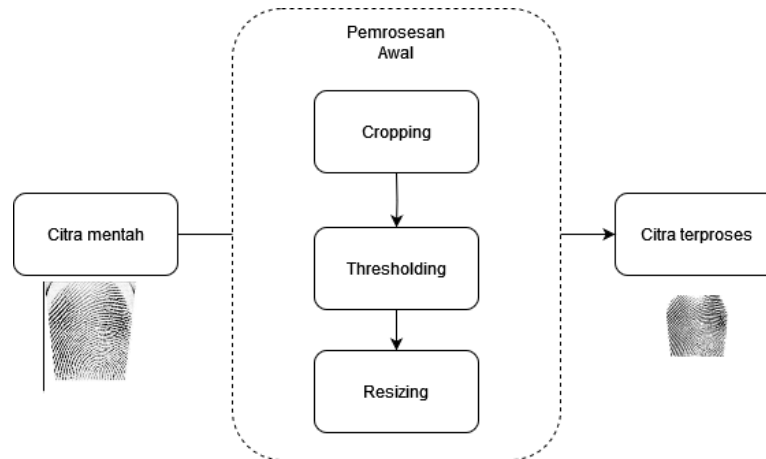
Adapun arsitektur umum dari penelitian ini dapat dilihat pada Gambar 3.1.



**Gambar 3.1** Arsitektur Umum

### 3.2.1 Pemrosesan Awal

Citra yang didapat dari perangkat keras DigitalPersona U.are.U 4500 sudah dalam bentuk format warna abu-abu (*Grayscale*), sehingga tidak diperlukan tahap pemrosesan awal *Grayscale* dengan menggunakan perangkat ini. Adapun alur kerja pemrosesan awal dalam penelitian ini dapat dilihat pada Gambar 3.2.



**Gambar 3.2** Alur kerja pemrosesan awal citra

Dari 74 responden, didapatkan 740 sidik jari unik, dimana setiap 740 sidik jari tersebut diambil 3 kali, sehingga didapatkan 2220 citra. Data tersebut kemudian dilakukan pemrosesan awal untuk membersihkan citra agar dapat dengan mudah dipelajari oleh model. Dalam pemrosesan awal, dilakukan *cropping*, *resizing*, dan *thresholding*.

Pada citra asli yang didapat langsung dari perangkat keras, terdapat bingkai lonjong parsial pada posisi bagian atas kiri dan kanan yang dapat dilihat seperti pada Gambar 3.3.



**Gambar 3.3** Citra mentah dari perangkat keras DigitalPersona U.are.U 4500



Untuk model dapat mempelajari fitur-fitur penting dari citra sidik jari, maka bingkai tersebut harus dihapus. Untuk menghapus bingkai tersebut dari citra sidik jari, maka dilakukan *cropping* dengan bentuk oval. Setelah dilakukan pengukuran dari titik tengah citra, didapatkan ukuran oval yang akan di *crop* untuk menghapus bingkai dari citra. Dari titik tengah citra, lebar ukuran oval adalah 162px, dan tingginya adalah 250px. Setelah dilakukan *cropping*, didapatkan citra gambar tanpa bingkai. Citra tersebut kemudian dilakukan *thresholding* untuk menghilangkan *noise* pada citra. Hasil citra yang telah dilakukan *cropping* dan *thresholding* dapat dilihat pada Gambar 3.4.



**Gambar 3.4** Citra setelah proses *cropping* dan *thresholding*

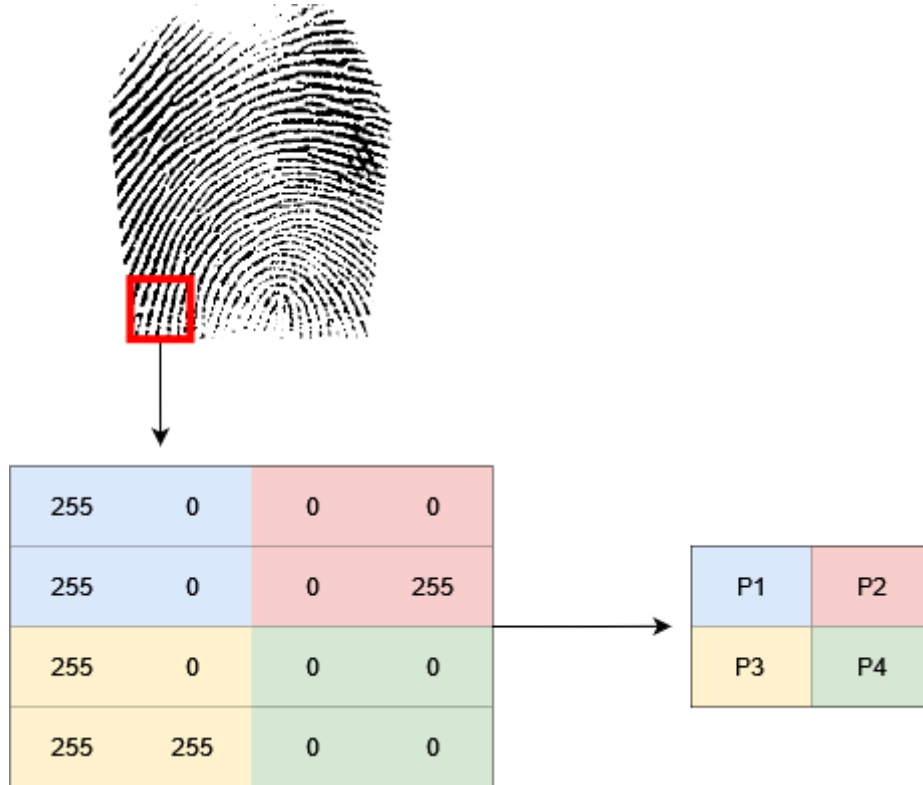
Setelah proses *thresholding*, didapatkan citra sidik jari dalam bentuk biner, yaitu hitam dan putih. Namun, pada citra yang telah diproses, terdapat bingkai putih yang mengelilingi raut sidik jari yang telah menjadi warna hitam. Untuk model dapat mempusatkan pembelajaran fitur pada sidik jari, maka bingkai putih tersebut harus dihapus. Untuk menghapus bingkai putih ini, harus dilakukan *cropping*.

Namun, berbeda dengan *cropping* sebelumnya, untuk menghapus bingkai putih pada citra sidik jari yang berbeda-beda ukuran sidik jarinya, perlu dilakukan *cropping* yang tepat sesuai dengan jarak dari ujung citra sampai tepat ke bagian sidik jari. Setelah bingkai putih dihapus dari citra, didapatkan citra-citra sidik jari yang tepat ukuran.

Karena sidik jari memiliki ukuran yang berbeda-beda, setelah proses *cropping* terakhir, setiap citra memiliki resolusi yang berbeda-beda. Untuk menyeragamkan citra sidik jari agar dapat diserap oleh model secara efisien, dilakukan proses normalisasi atau *resizing*. Semakin kecil resolusi citra maka semakin efisien model menyerap fitur-fiturnya.

Namun, jika resolusi citra terlalu kecil, maka model akan gagal mempelajari fitur-fitur penting yang ada dalam citra tersebut. Setelah proses *trial and error*, didapatkan resolusi yang tepat, yaitu dengan lebar 90px dan tinggi 90px.

Proses *resizing* dapat di ilustrasikan dengan contoh gambar pada Gambar 3.5.



**Gambar 3.5** Contoh proses *resizing* citra

Pada Gambar 3.5, dapat dilihat contoh dari pixel 4x4 yang akan di *resize* menjadi 2x2, yang berarti pada contoh ini citra akan di *resize* menjadi setengah dari ukuran aslinya. Proses *resize* dilakukan dengan mencari *mode* dari ke 4 bagian matriks dengan ukuran 2x2.

Pada umumnya, proses *resize* dilakukan dengan mencari *mean* dari bagian matriks, namun dikarenakan citra yang akan di *resize* dalam penelitian ini dalam bentuk binari, maka nilai yang dicari adalah *mode* dari bagian matriks.

Dikarenakan jumlah dari nilai didalam matriks adalah genap, maka dibutuhkan operator untuk memecahkan kasus dimana kedua nilai (0 dan 255) berjumlah sama. Didalam penelitian ini, operator yang digunakan adalah lebih besar atau sama dengan ( $\geq$ ), dimana rumus yang akan digunakan dapat dilihat sebagai berikut:

$$P = f_{255} \geq \frac{1}{2} m ? 255 : 0$$

Di mana:

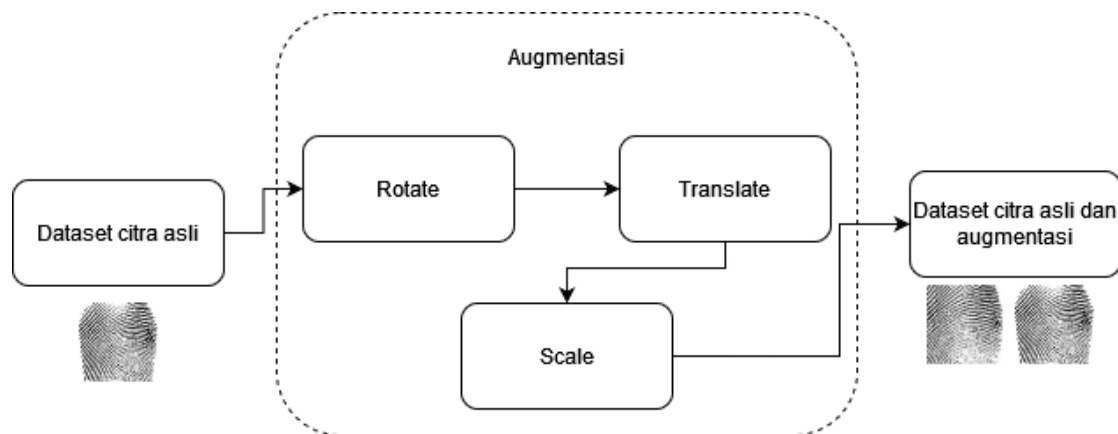
- $P$  = Nilai dari pixel
- $f_{255}$  = Frekuensi adanya nilai 255 (warna putih)
- $m$  = Populasi matriks

Dengan demikian, didapatkan nilai untuk hasil *resize* pixel 4x4 menjadi 2x2 sebagai berikut:

- $P1 = \{255, 255, 0, 0\} = 255$
- $P2 = \{0, 0, 0, 255\} = 0$
- $P3 = \{255, 0, 255, 255\} = 255$
- $P4 = \{0, 0, 0, 0\} = 0$

### 3.2.2 Augmentasi

Untuk dapat mengadaptasikan model terhadap situasi nyata, citra yang telah di proses di augmentasikan lebih lanjut. Augmentasi ini menggunakan 3 teknik alterasi gambar, yaitu *Scale*, *Rotate*, dan *Translate*. Masing-masing gambar yang tersedia didalam dataset di augmentasikan menggunakan 3 teknik tersebut berdasarkan 7 jenis kombinasi. Alur kerja augmentasi citra dapat dilihat pada Gambar 3.6.



**Gambar 3.6** Alur kerja augmentasi citra

Adapun 7 jenis kombinasi tersebut yaitu:

- 1) *Rotate*
- 2) *Translate*

- 3) *Scale*
- 4) *Rotate, Translate*
- 5) *Rotate, Scale*
- 6) *Scale, Translate*
- 7) *Rotate, Translate, Scale*

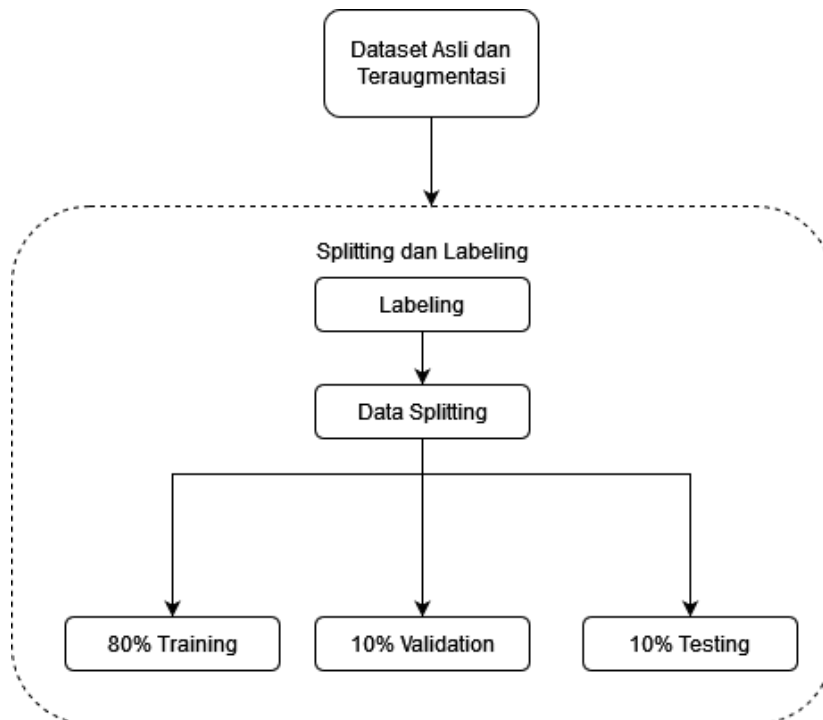
Masing-masing jenis kombinasi menghasilkan 5 gambar dengan spesifikasi teknik yang berbeda-beda. Setelah di augmentasi, dataset 740 sidik jari yang awalnya memiliki 2220 citra, sekarang memiliki 35 gambar per citra, menghasilkan 77700 citra ter-augmentasi. Dijumlahkan dengan citra asli, total keseluruhan dataset memiliki 79920 citra sidik jari. Contoh augmentasi gambar dapat dilihat pada Gambar 3.7.



**Gambar 3.7** Augmentasi gambar berdasarkan 1 sidik jari

### 3.2.3 Dataset Splitting dan Labeling

Dataset yang telah di augmentasi kemudian dibelah menjadi 3 bagian berdasarkan persentase: 80% untuk training, 10% untuk validasi, dan 10% untuk testing. Sebelum dibelah, dataset dilabel berdasarkan nomor subjek dan bagian jari. Metadata per citra terdapat pada nama dari file citra tersebut. Alur splitting dan labeling dataset dapat dilihat pada Gambar 3.8.



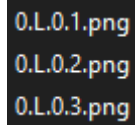
**Gambar 3.8** Alur kerja splitting dan labeling dataset

Adapun format nama dari file citra dari dataset adalah sebagai berikut:

- 1) Nomor subjek: Nomor dari subjek. Dapat diartikan sebagai nomor *id*.
- 2) Delimiter: Pembelah komponen dalam format. Delimiter digunakan untuk membelah komponen-komponen yang ada. Delimiter adalah *dot* ( “ . “ ).
- 3) Bagian tangan: Bagian tangan dari jari yang diambil sidik jarinya. Bagian tangan dapat berupa kiri (L) atau kanan (R).
- 4) Indeks jari: Nomor indeks dari jari. Indeks jari menggunakan angka, mulai dari 0 sampai 4, dengan 0 sebagai ibu jari, dan 4 sebagai jari kelingking.
- 5) Indeks pengambilan: Nomor indeks pengambilan. Dikarenakan setiap jari diambil 3 kali, maka indeks ini diperlukan untuk membedakan citra sidik jari yang diambil. Indeks pengambilan dimulai dari angka 1, sampai angka 3.

Berdasarkan format tersebut, susunan metadata yang terdapat didalam nama file citra adalah sebagai berikut:

**“{nomor subjek}.{bagian tangan}.{indeks jari}.{indeks pengambilan}.png”**



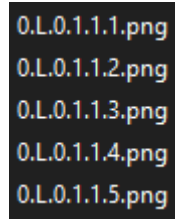
**Gambar 3.9** Contoh nama file dari subjek nomor 0 ibu jari kiri

Citra teraugmentasi memiliki format nama file yang berbeda dengan citra asli. Dikarenakan augmentasi diulangi sebanyak 35 kali per citra, metadata didalam nama file memiliki 2 komponen tambahan yang berada di akhir, diantaranya sebagai berikut:

- 1) Nomor jenis kombinasi augmentasi: Nomor dari jenis kombinasi augmentasi. Seperti yang telah disebutkan sebelumnya, terdapat 7 jenis kombinasi augmentasi. Nomor ini dimulai dari angka 1, sampai angka 7
- 2) Indeks iterasi augmentasi: Nomor dari pengulangan jenis augmentasi. Seperti yang telah disebutkan sebelumnya, setiap jenis kombinasi augmentasi dilakukan sebanyak 5 kali per citra. Nomor ini dimulai dari angka 1, sampai angka 5.

Berdasarkan format tersebut, susunan metadata yang terdapat didalam nama file citra yang teraugmentasi adalah sebagai berikut:

**“{nomor subjek}.{bagian tangan}.{indeks jari}.{indeks pengambilan}.{nomor jenis kombinasi}.{indeks iterasi augmentasi}.png”**



0.L.0.1.1.1.png  
0.L.0.1.1.2.png  
0.L.0.1.1.3.png  
0.L.0.1.1.4.png  
0.L.0.1.1.5.png

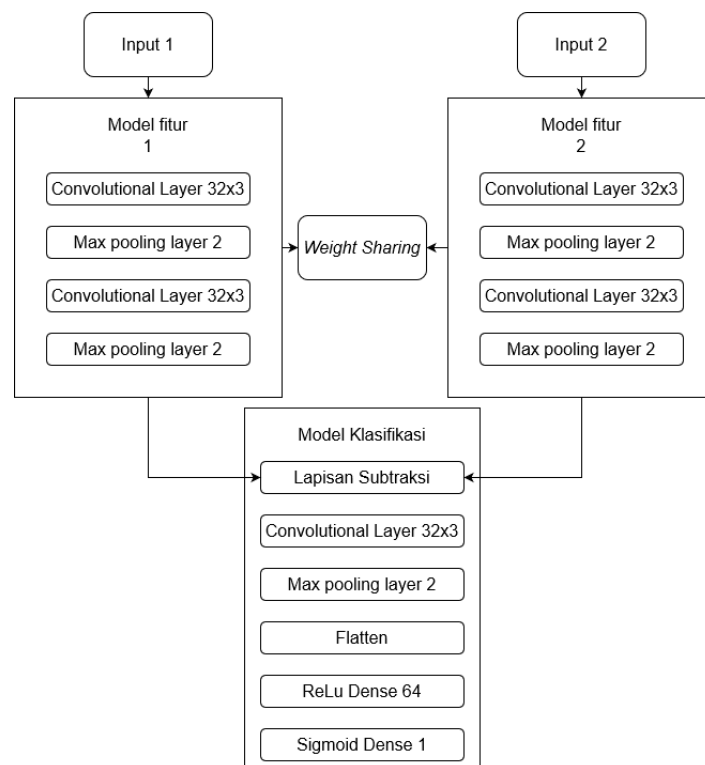
**Gambar 3.10** Contoh nama file dari subjek 0 ibu jari kiri pengambilan pertama

Berdasarkan metadata yang terdapat didalam nama file, citra yang ada didalam dataset dapat diberi label dan dikelompokkan dalam satu kelas berdasarkan metadata tersebut. Didalam setiap kelas, citra asli dipisahkan dengan citra yang teraugmentasi. Untuk menyederhanakan format label, beberapa komponen dihapus. Yang kemudian didapatkan format baru yang digunakan sebagai nama kelas sebagai berikut:

**“{nomor subjek}.{bagian tangan}.{indeks jari}”**

### 3.2.4 Arsitektur Model

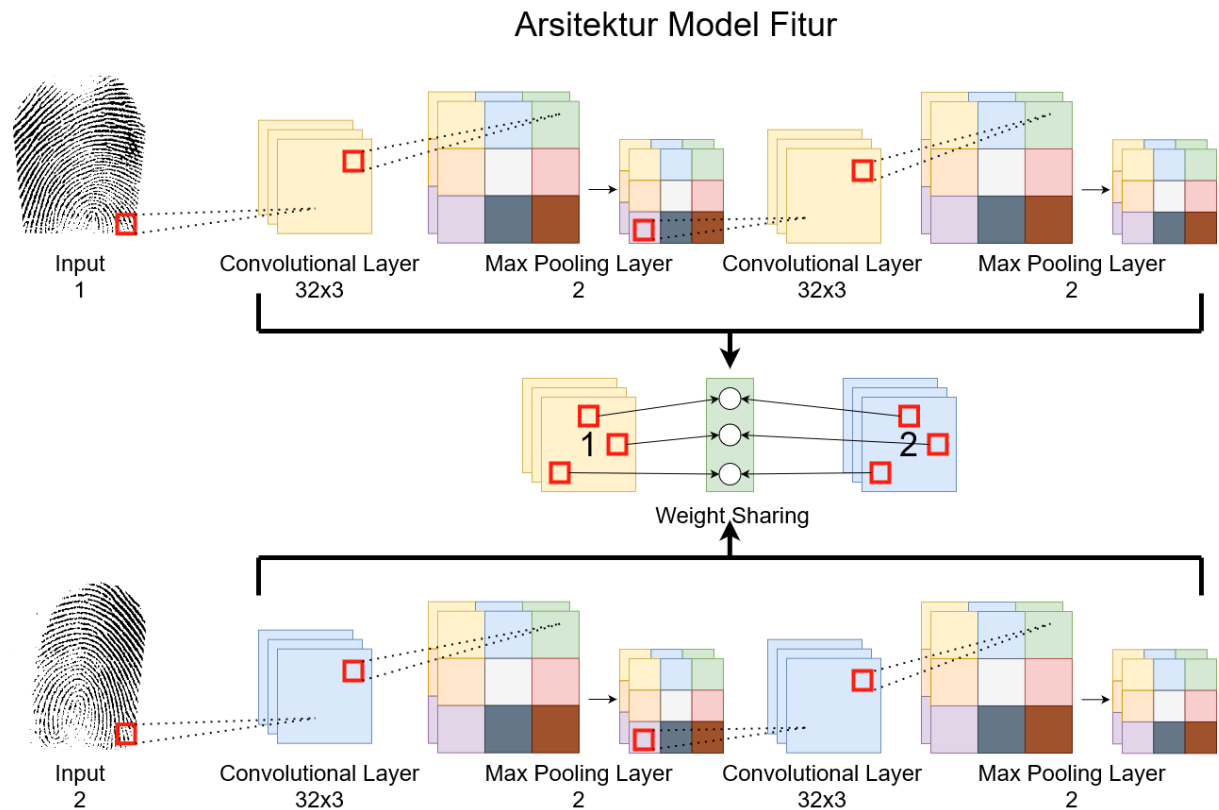
Dalam penelitian ini, jenis arsitektur model yang akan digunakan adalah model siamese, atau disebut juga model kembar. Dalam model kembar, digunakan 2 model serupa, dimana kedua model tersebut mengambil 2 input yang berbeda, namun berbagi berat yang sama. Peran dari kedua model tersebut adalah untuk ekstraksi fitur yang ada didalam kedua input, yang kemudian fitur dari kedua model tersebut di berikan kepada model ketiga. Karenanya, kedua model tersebut juga dapat diberi nama sebagai model fitur. Peran dari model ketiga adalah untuk menerima input dari kedua model dan memberikan hasil akhir.



**Gambar 3.11** Arsitektur model

Didalam model fitur, terdapat 4 lapisan, yaitu 2 lapisan *Convolutional*, dan 2 lapisan *max pooling*. Lapisan *Convolutional* masing-masing memiliki 32 filter, sedangkan lapisan *max pooling* masing-masing memiliki ukuran *pool* sebesar 2x2. Pada lapisan *Convolutional*, digunakan ukuran kernel sebesar 3x3, dimana nilai ini diambil berdasarkan perbandingan 1 : 10 dari ukuran citra yang akan diberikan sebagai input kepada model.

Kedua fitur model berbagi satu berat, atau juga disebut *weight sharing*. *Weight sharing* umum ditemukan didalam model kembar, dimana *weight sharing* dapat membantu model dalam mempelajari persamaan dan pertidak samaan yang dimiliki oleh kedua input, sehingga dapat membantu model dalam mempelajari cara membedakan dan menyamakan fitur yang ada pada citra.



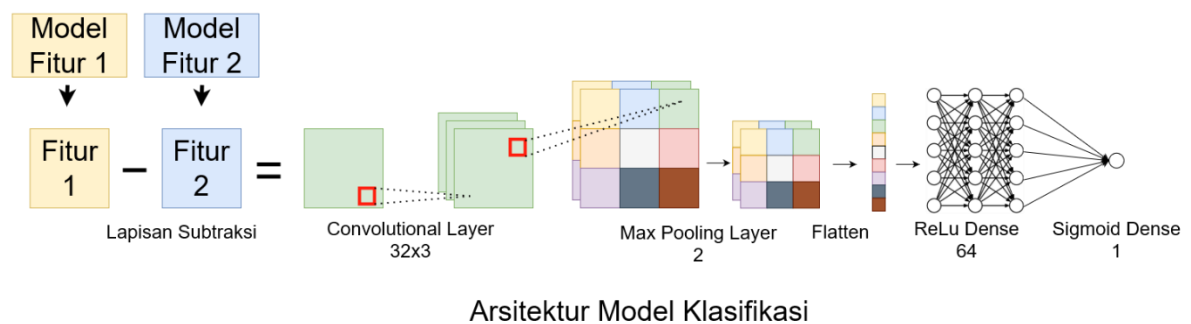
**Gambar 3.12** Visualisasi Arsitektur Model Fitur

Setelah kedua model fitur selesai memproses citra, fitur dari kedua model tersebut kemudian diteruskan kepada model ketiga. Didalam model ketiga, terdapat 6 lapisan, yaitu 1 lapisan subtraksi, 1 lapisan *Convolutional*, 1 lapisan *max pooling*, 1 lapisan *Flatten*, 1 lapisan *dense* dengan aktivasi *Rectified Linear Unit* (ReLU), dan 1 lapisan *dense Sigmoid* sebagai lapisan output. Hasil dari kedua model pertama diteruskan kepada lapisan subtraksi didalam model ketiga, dimana didalam lapisan ini, kedua hasil yang didapatkan dari kedua model fitur dikurangi. Hasil pengurangan ini adalah sebuah fitur yang baru, dimana fitur ini adalah perbedaan absolut atau kuadrat dari kedua fitur yang didapat dari kedua model kembar.



Fitur tersebut kemudian diteruskan ke lapisan *Convolutional* dengan parameter yang sama seperti dalam model fitur, yaitu dengan 32 filter dan ukuran kernel sebesar 3x3. Kemudian, fitur diteruskan ke lapisan *max pooling* untuk dilakukan *downsampling* terhadap fitur yang didapat dari lapisan *Convolutional*. Setelah itu, fitur kemudian diberikan ke lapisan *Flatten*, dimana fitur matriks multi dimensi diubah menjadi fitur vektor satu dimensi.

Setelah menjadi vektor satu dimensi, fitur kemudian diteruskan ke lapisan *dense ReLU*, dimana lapisan ini memiliki 64 unit. Jumlah unit ini didapatkan dari proses *trial and error*. Lapisan *dense ReLU* kemudian meneruskan neuron yang telah terhubung dari ke 64 unit didalamnya ke lapisan selanjutnya, yaitu lapisan *dense sigmoid*. Sebagai model yang akan memberikan prediksi kecocokkan atau tidak dari 2 gambar, lapisan terakhir ini hanya memiliki 1 unit untuk dapat memberikan 1 nilai saja. Dengan aktivasi *sigmoid*, nilai tersebut dapat dipastikan berupa nilai float dari 0.0 hingga 1.0. Nilai ini adalah nilai akhir, dimana nilai ini adalah tingkat persentase dari persamaan kedua citra yang diberi ke lapisan input dari model.

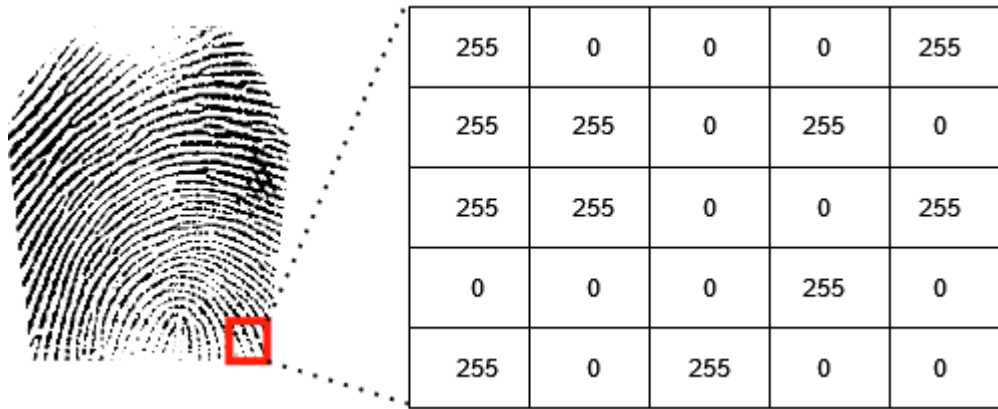


**Gambar 3.13** Visualisasi Arsitektur Model Klasifikasi

Adapun beberapa penjelasan tentang lapisan-lapisan yang digunakan didalam model ini adalah sebagai berikut:

#### 1) Lapisan Konvolusi (*Convolutional Layer*)

Lapisan Konvolusi berperan sebagai lapisan pengekstrak fitur-fitur yang ada didalam input yang berupa matriks. Lapisan Konvolusi menggunakan filter yang dikalkulasikan dengan input, dan menggerakkan filter tersebut (*convolve*) ke area matriks selanjutnya. Sebagai contoh, sebuah matriks input dapat dilihat pada Gambar 3.14.



**Gambar 3.14** Visualisasi matriks input

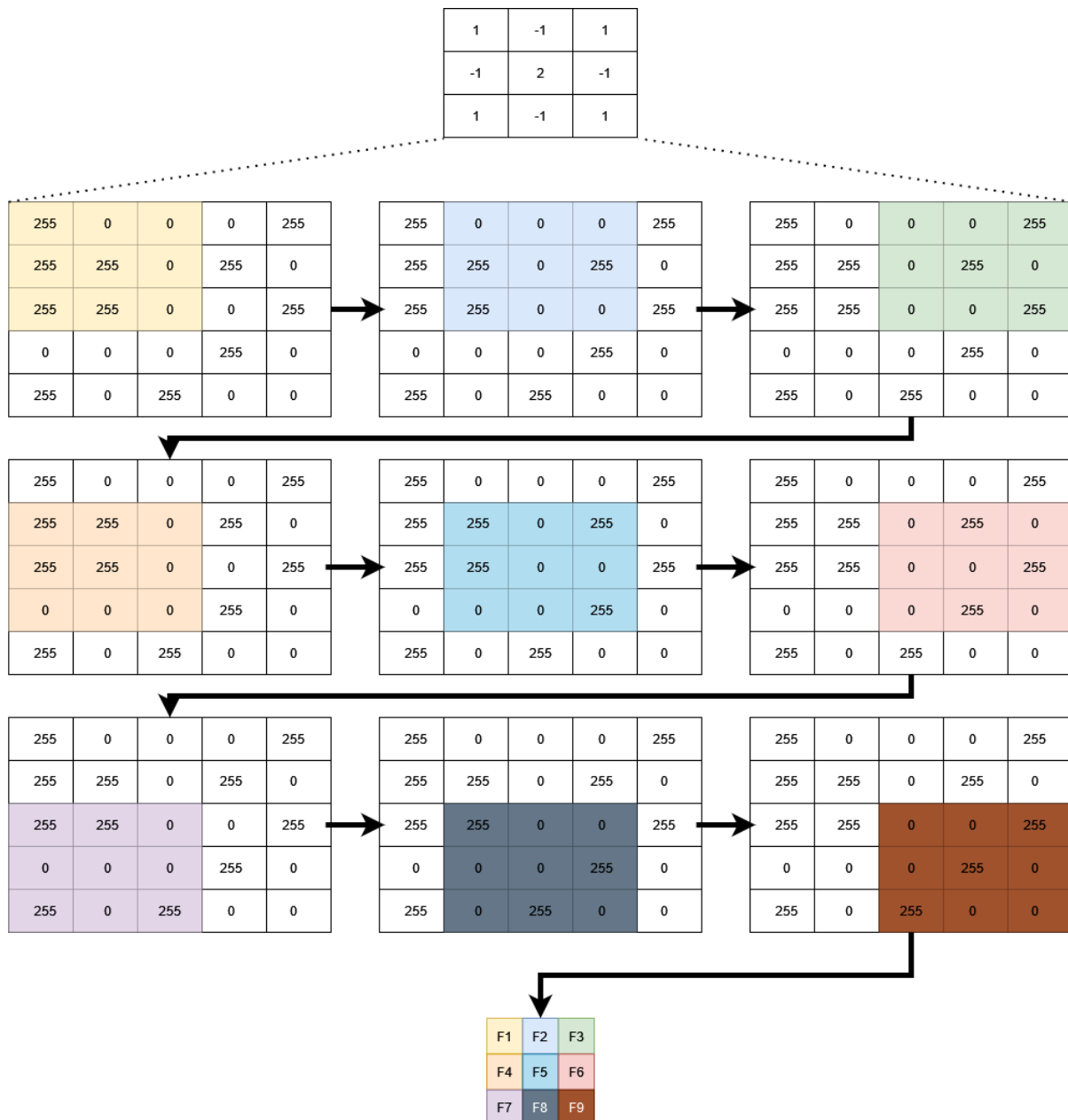
Pada Gambar 3.14, dapat dilihat sebuah matriks 5x5 dari potongan citra sidik jari. Matriks ini adalah nilai dari warna pada pixel. Nilai tersebut adalah 0 (hitam) dan 255 (putih) dikarenakan citra yang diberikan adalah citra binari atau hitam putih.

Nilai dari filter 3x3 yang akan digunakan pada matriks ini adalah sebagai berikut:

$$\begin{bmatrix} 1 & -1 & 1 \\ -1 & 2 & -1 \\ 1 & -1 & 1 \end{bmatrix}$$

Nilai filter ini adalah nilai semu. Pada operasi lapisan konvolusi yang sebenarnya, nilai filter didapat dari apa yang telah dipelajari oleh jaringan neural dari operasi filter sebelumnya, sehingga nilai filter tidak bersifat tetap.

Ukuran filter yang digunakan adalah 3x3, namun pergerakan filter tidak didapat dari perbandingan ukuran filter dengan ukuran input. Pergerakan filter ditetapkan dari parameter *stride*. Pada contoh ini, nilai yang digunakan untuk parameter *stride* adalah 1, sehingga filter akan bergerak setiap 1 kolom dari kiri ke kanan, dan dari atas ke bawah. Visualisasi pergerakan filter dapat dilihat pada Gambar 3.15.



**Gambar 3.15** Visualisasi pergerakan filter lapisan konvolusi

Pada contoh ini, pergerakan filter akan menghasilkan matriks fitur 3x3, dimana nilai dari matriks tersebut didapat dari hasil pengkalian filter pada area matriks input.

Adapun operasi filter secara lengkap pada contoh kali ini adalah sebagai berikut:

$$F1 = (255 \times 1) + (0 \times -1) + (0 \times 1) + (255 \times -1) + (255 \times 2) + (0 \times -1) + (255 \times 1) + (255 \times -1) + (0 \times 1) = 510$$

$$F2 = (0 \times 1) + (0 \times -1) + (0 \times 1) + (255 \times -1) + (0 \times 2) + (255 \times -1) + (255 \times 1) + (0 \times -1) + (0 \times 1) = -255$$

$$F3 = (0 \times 1) + (0 \times -1) + (255 \times 1) + (0 \times -1) + (255 \times 2) + (0 \times -1) + (0 \times 1) + (0 \times -1) + (255 \times 1) = 1020$$

$$F4 = (255 \times 1) + (255 \times -1) + (0 \times 1) + (255 \times -1) + (255 \times 2) + (0 \times -1) + (0 \times 1) + (0 \times -1) + (0 \times 1) = 765$$

$$F5 = (255 \times 1) + (0 \times -1) + (255 \times 1) + (255 \times -1) + (0 \times 2) + (0 \times -1) + (0 \times 1) + (0 \times -1) + (255 \times 1) = 510$$

$$F6 = (0 \times 1) + (255 \times -1) + (0 \times 1) + (0 \times -1) + (0 \times 2) + (255 \times -1) + (0 \times 1) + (255 \times -1) + (0 \times 1) = -765$$

$$F7 = (255 \times 1) + (255 \times -1) + (0 \times 1) + (0 \times -1) + (0 \times 2) + (0 \times -1) + (255 \times 1) + (0 \times -1) + (255 \times 1) = 510$$

$$F8 = (255 \times 1) + (0 \times -1) + (0 \times 1) + (0 \times -1) + (0 \times 2) + (255 \times -1) + (0 \times 1) + (255 \times -1) + (0 \times 1) = -255$$

$$F9 = (0 \times 1) + (0 \times -1) + (255 \times 1) + (0 \times -1) + (255 \times 2) + (0 \times -1) + (255 \times 1) + (0 \times -1) + (0 \times 1) = 1020$$

Sehingga didapatkan matriks fitur dari hasil operasi filter sebagai berikut:

$$\begin{bmatrix} 510 & -255 & 1020 \\ 765 & 510 & -765 \\ 510 & -255 & 1020 \end{bmatrix}$$

## 2) Lapisan *Max Pooling*

Lapisan *Max Pooling* berperan dalam mengurangi *noise* didalam fitur matriks dengan mengambil nilai terbesar didalam area matriks, sehingga model dapat mempelajari fitur yang paling penting dengan menghapus fitur-fitur yang tidak menonjol pada area tersebut.

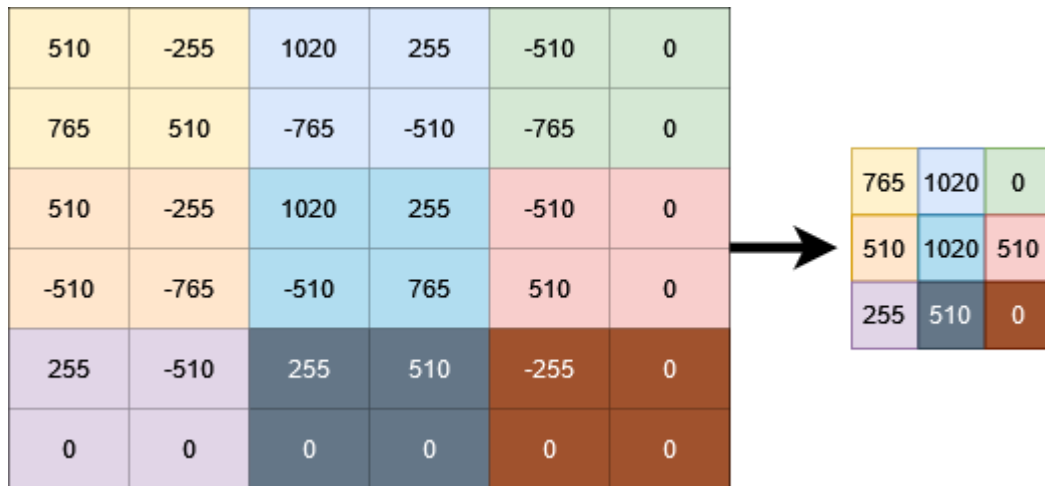
Ukuran dari area *max pooling* ditetapkan dari parameter *pool size*, dimana pada umumnya nilai ini adalah 2 atau 3. Sebagai contoh, apabila nilai *pool size* adalah 2, maka nilai yang akan diambil untuk dimasukkan ke matriks fitur yang baru adalah nilai tertinggi yang ada pada area 2x2. Apabila ukuran dari fitur matriks yang diberi tidak dapat dibagi dengan nilai *pool size*, maka matriks tersebut akan diberi *padding* berupa angka 0 pada bagian bawah dan kanan matriks, sehingga *max pooling* dapat dilakukan pada fitur matriks. Sebagai contoh, terdapat sebuah fitur matriks dengan ukuran 5x5 sebagai berikut:

$$\begin{bmatrix} 510 & -255 & 1020 & 255 & -510 \\ 765 & 510 & -765 & -510 & -765 \\ 510 & -255 & 1020 & 255 & -510 \\ -510 & -765 & -510 & 765 & 510 \\ 255 & -510 & 255 & 510 & -255 \end{bmatrix}$$

Matriks fitur tersebut kemudian diberikan ke lapisan *max pooling* dengan *pool size* senilai 2. Dikarenakan 5 tidak dapat dibagi habis dengan 2, maka matriks harus diberi *padding*, sehingga ukurannya dapat dibagi habis dengan 2. Nilai terdekat dari 5 yang dapat dibagi habis dengan 2 adalah 6, maka matriks akan diberi *padding* dengan ukuran 1x1, sehingga menghasilkan fitur matriks 6x6 sebagai berikut:

$$\begin{bmatrix} 510 & -255 & 1020 & 255 & -510 & 0 \\ 765 & 510 & -765 & -510 & -765 & 0 \\ 510 & -255 & 1020 & 255 & -510 & 0 \\ -510 & -765 & -510 & 765 & 510 & 0 \\ 255 & -510 & 255 & 510 & -255 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fitur matriks ini akan menghasilkan fitur matriks baru dengan ukuran 3x3, dimana ukuran ini didapat dari hasil pembagian ukuran fitur matriks yang dapat dibagi dengan nilai *pool size*, dimana pada contoh ini adalah 6 dibagi dengan 2. Visualisasi *max pooling* dapat dilihat pada Gambar 3.16.



**Gambar 3.16** Visualisasi *max pooling* pada matriks fitur yang telah diberi *padding*

### 3) Lapisan Subtraksi

Lapisan Subtraksi berperan dalam mendapatkan delta perbedaan dari 2 matriks fitur yang didapat dari kedua model fitur. Operasi yang dilakukan pada lapisan subtraksi adalah hanya mengurangi nilai dari kedua matriks dengan urutan yang tetap pada setiap iterasi, seperti mengurangi model 1 dengan model 2, atau model 2 dengan model 1, dan mengurangi fitur matriks selanjutnya dengan urutan yang sama dengan fitur matriks sebelumnya.

Sebagai contoh, terdapat 2 fitur matriks yang didapatkan dari kedua model fitur dengan ukuran 3x3 sebagai berikut:

$$M1 = \begin{bmatrix} 765 & 1020 & 0 \\ 510 & 1020 & 510 \\ 255 & 510 & 0 \end{bmatrix}$$

$$M2 = \begin{bmatrix} 210 & 510 & 1020 \\ 210 & 210 & 765 \\ 510 & 0 & 1020 \end{bmatrix}$$

Kedua fitur matriks tersebut kemudian diberikan ke lapisan subtraksi dengan urutan fitur 2 dikurangi fitur 1, sehingga menghasilkan sebuah delta matriks dari kedua matriks fitur.

Adapun persamaan delta matriks adalah sebagai berikut:

$$M\Delta = M2 - M1$$

$$M\Delta = \begin{bmatrix} 765 & 1020 & 0 \\ 510 & 1020 & 510 \\ 255 & 510 & 0 \end{bmatrix} - \begin{bmatrix} 210 & 510 & 1020 \\ 210 & 210 & 765 \\ 510 & 0 & 1020 \end{bmatrix}$$

$$M\Delta = \begin{bmatrix} 765 - 210 & 1020 - 510 & 0 - 1020 \\ 510 - 210 & 1020 - 210 & 510 - 765 \\ 255 - 510 & 510 - 0 & 0 - 1020 \end{bmatrix}$$

$$M\Delta = \begin{bmatrix} 555 & 510 & -1020 \\ 300 & 810 & -255 \\ -255 & 510 & -1020 \end{bmatrix}$$

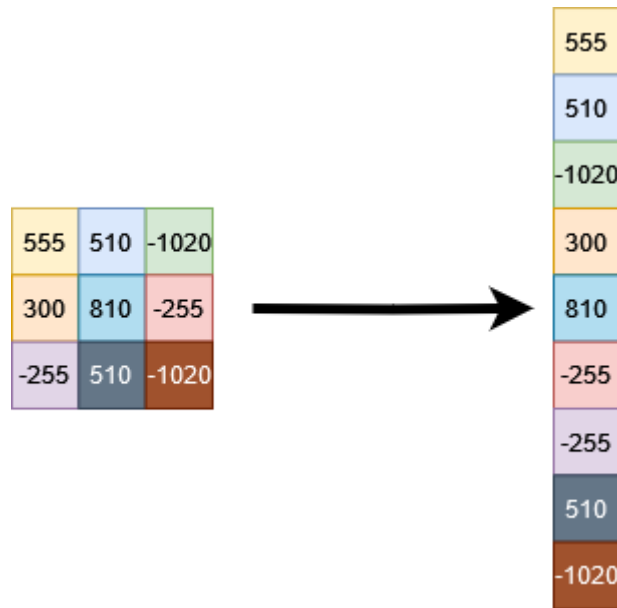
#### 4) Lapisan *Flatten*

Lapisan *Flatten* berperan dalam mengubah matriks multi dimensi menjadi vektor satu dimensi. Pada umumnya, lapisan ini digunakan sebelum fitur diberikan kepada lapisan *dense*. Tidak ada operasi matematika yang dilakukan didalam lapisan *flatten* dikarenakan tugas dari lapisan *flatten* hanyalah mengubah bentuk dari input multi dimensi yang diberikan menjadi satu dimensi.

Sebagai contoh, terdapat sebuah matriks sebagai berikut:

$$\begin{bmatrix} 555 & 510 & -1020 \\ 300 & 810 & -255 \\ -255 & 510 & -1020 \end{bmatrix}$$

Matriks tersebut memiliki ukuran 3x3. Namun, lapisan *dense* hanya dapat menerima input satu dimensi. Tugas dari lapisan *flatten* adalah untuk mempersiapkan matriks yang akan diberikan kepada lapisan *dense* dengan mengubah matriks multi dimensi tersebut menjadi vektor satu dimensi. Adapun visualisasi perubahan tersebut dapat dilihat pada Gambar 3.17.



**Gambar 3.17** Visualisasi pengubahan matriks multi dimensi menjadi vektor satu dimensi

#### 5) Lapisan *Dense*

Lapisan *Dense* adalah inti dari setiap jaringan neural (*neural network*). Peran dari lapisan *dense* adalah untuk mempelajari pola yang ada pada data, atau pada fitur. Lapisan *dense* terdiri dari beberapa neuron, dimana setiap neuron terhubung dengan setiap nilai yang didapat dari input lapisan sebelumnya. Pada umumnya, rumus yang digunakan dalam lapisan *dense* untuk setiap neuron adalah sebagai berikut:

$$d = A\left(\sum_{n=1}^{N-1} (i_n \times w_n) + b\right)$$

Di mana:

- $d$  = Hasil akhir dari satu lapisan *dense*
- $A$  = Fungsi aktivasi
- $i$  = Input dari lapisan sebelumnya
- $w$  = Berat dari lapisan *dense*
- $b$  = Bias dari neuron yang lain dalam satu lapisan *dense* yang sama



Setiap lapisan *dense* memiliki fungsi aktivasi, dimana setiap fungsi aktivasi memiliki rumus yang berbeda, dan memiliki tujuan yang berbeda-beda. Didalam penelitian ini, terdapat 2 lapisan *dense* dengan aktivasi yang berbeda, yaitu aktivasi *Rectified Linear Unit* (ReLU) dan aktivasi Sigmoid. Adapun penjelasan dari kedua aktivasi tersebut adalah sebagai berikut:

a) *Rectified Linear Unit*

Aktivasi ReLU bertujuan untuk memberikan nonlinier kepada jaringan, membuat jaringan tersebut dapat mempelajari pola kompleks yang ada pada data. Secara sederhana, fungsi dari aktivasi ReLU mengganti nilai negatif menjadi 0, dan menyimpan nilai positif. Rumus aktivasi ReLU adalah sebagai berikut:

$$ReLU(i) = \max(0, i)$$

Di mana:

- ReLU = Fungsi aktivasi ReLU
- i = Input dari lapisan sebelumnya
- max = Fungsi maximum matematika

Sebagai contoh, terdapat sebuah vektor satu dimensi sebagai berikut:

$$v = \begin{bmatrix} 555 \\ 510 \\ -1020 \\ 300 \\ 810 \\ -255 \\ -255 \\ 510 \\ -1020 \end{bmatrix}$$

Vektor satu dimensi tersebut kemudian diberikan kepada lapisan *dense* dengan aktivasi ReLU. Lapisan *dense* ReLU tersebut memiliki bias sebesar 0.1 dan berat sebagai berikut:

$$w = \begin{bmatrix} 0.5 \\ -0.2 \\ 0.8 \\ 0.2 \\ 0.2 \\ -0.1 \\ -0.5 \\ 0.3 \\ 0.1 \end{bmatrix}$$

Sebelum aktivasi ReLU, lapisan *dense* akan mengaplikasikan berat ke vektor satu dimensi tersebut. Adapun operasi pengaplikasian berat ke vektor satu dimensi adalah sebagai berikut:

$$i = (v \times w) + b$$

$$i = \left( \begin{bmatrix} 555 \\ 510 \\ -1020 \\ 300 \\ 810 \\ -255 \\ -255 \\ 510 \\ -1020 \end{bmatrix} \times \begin{bmatrix} 0.5 \\ -0.2 \\ 0.8 \\ 0.2 \\ 0.2 \\ -0.1 \\ -0.5 \\ 0.3 \\ 0.1 \end{bmatrix} \right) + 0.1$$

$$i = (555 \times 0.5 + 510 \times -0.2 - 1020 \times 0.8 + 300 \times 0.2 + 810 \times 0.2 - 255 \times -0.1 - 255 \times -0.5 + 510 \times 0.3 - 1020 \times 0.1) + 0.1$$

$$i = -265.5 + 0.1$$

$$i = -265.4$$

Setelah operasi pengaplikasian berat, didapatkan 1 nilai  $i$  yang berupa -265.4. Kemudian, nilai tersebut diberikan ke aktivasi ReLU. Adapun operasi aktivasi ReLU adalah sebagai berikut:

$$ReLU = \max(0, i)$$

$$ReLU = \max(0, -265.4)$$

$$ReLU = 0$$

Dikarenakan -265.4 lebih rendah dari pada 0, maka hasil akhir yang akan diberikan lapisan ini adalah 0.

#### b) Sigmoid

Aktivasi Sigmoid bertujuan untuk mengubah nilai hasil dari lapisan sebelumnya, umumnya lapisan *dense*, menjadi nilai antara 0 dan 1. Umumnya, lapisan ini digunakan terakhir dalam model jaringan neural dengan tujuan untuk memberikan tingkat perbandingan dalam tugas klasifikasi.

Adapun rumus dari aktivasi Sigmoid adalah sebagai berikut:

$$Sigmoid(i) = \frac{1}{1 + e^{-\sum_{n=1}^{N-1}(i_n \times w_n)}}$$

Di mana:

- Sigmoid = Fungsi aktivasi Sigmoid
- $i$  = Input dari lapisan sebelumnya
- $e$  = Nilai konstan matematika Euler dengan nilai perkiraan 2.718
- $w$  = Berat dari lapisan *dense*

Sebagai contoh, terdapat nilai yang diberikan dari lapisan *dense* ReLU sebagai berikut:

$$ReLU = 0$$

Nilai tersebut kemudian diberikan kepada lapisan *dense* dengan aktivasi Sigmoid. Lapisan *dense* Sigmoid tersebut memiliki bias sebesar 0.4 dan berat 0.7.

Sebelum aktivasi Sigmoid, lapisan *dense* akan mengaplikasikan berat ke nilai tersebut. Adapun operasi pengaplikasian berat ke nilai yang didapat dari lapisan *dense* ReLU adalah sebagai berikut:

$$i = (ReLU \times w) + b$$

$$i = (0 \times 0.7) + 0.4$$

$$i = 0 + 0.4$$

$$i = 0.4$$

Setelah operasi pengaplikasian berat, didapatkan nilai  $i$  yang berupa 0.4. Kemudian, nilai tersebut diberikan ke aktivasi Sigmoid. Adapun operasi aktivasi Sigmoid adalah sebagai berikut:

$$Sigmoid = \frac{1}{1 + e^{-\sum_{n=1}^{N-1}(i_n \times w_n)}}$$

$$Sigmoid = \frac{1}{1 + e^{-0.4}}$$

$$Sigmoid = \frac{1}{1 + 0.67032}$$

$$Sigmoid = 0.599$$

Dengan demikian, didapatkan hasil akhir yang diberikan oleh lapisan *dense* Sigmoid, yaitu 0.599.

### 3.2.5 Pelatihan Model

Pelatihan model dilakukan dengan menggunakan *framework TensorFlow* menggunakan bahasa pemrograman Python. Berbeda dengan model pada umumnya, model kembar membutuhkan 2 input data yang berbeda, dimana input tersebut adalah 2 citra dari sidik jari yang sama, atau berbeda. Jika kedua citra yang diberi adalah citra dari satu sidik jari yang sama, maka input tersebut diberi label 1.0, dan jika kedua citra yang diberi adalah citra dari 2 sidik jari yang berbeda maka input tersebut diberi label 0.0. Label 1.0 menandakan bahwa kedua input memiliki kecocokkan 100%, sedangkan label 0.0 menandakan bahwa kedua input memiliki kecocokkan 0%.

Untuk dapat memastikan bahwa model terlatih dengan benar, maka dibutuhkan validasi model. Validasi model dilakukan pada setiap akhir *epoch* pelatihan. Data validasi yang di bagi pada subbab 3.2.3 adalah kumpulan dari kelas sidik jari yang tidak pernah dilihat oleh model ketika pelatihan dilakukan. Validasi dilakukan dengan proses yang sama dengan pelatihan model, dengan perbedaan tujuan dari validasi adalah untuk meng evaluasi performa dari model ketika diberikan data yang tidak pernah dilihat pada saat pelatihan.

### 3.2.6 Uji Coba Model

Uji coba model dilakukan untuk mengases performa yang dimiliki oleh model pada saat diberikan data yang tidak pernah dilihat sebelumnya. Berbeda dengan validasi, uji coba model tidak mempengaruhi parameter, berat, ataupun bias yang telah dimiliki oleh model ketika pelatihan model dilakukan. Oleh karena itu, data yang digunakan untuk uji coba model harus berbeda dengan data yang digunakan ketika pelatihan dan validasi model dilakukan.

Uji coba model dilakukan dengan metode *Confusion Matrix*. Dengan melihat nilai-nilai yang didapat dari 4 komponen *Confusion Matrix*, kita dapat melihat performa model dalam melakukan prediksi dengan data yang tidak pernah dilihat sebelumnya. Proses uji coba memiliki prinsip yang sama dengan proses pelatihan dan validasi, dengan perbedaan setiap prediksi akan menghasilkan nilai prediksi dan nilai yang sebenarnya.

### 3.2.7 Implementasi Sistem

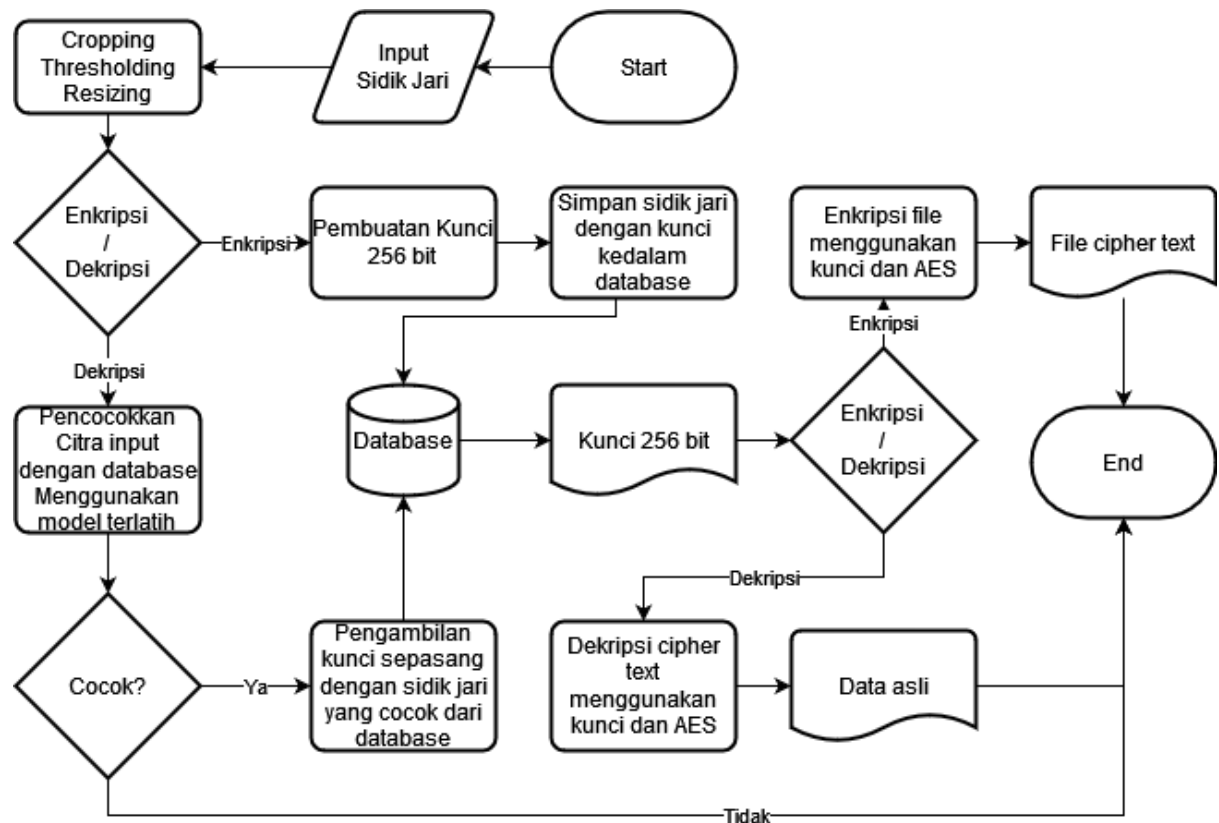
Pada tahap ini, model yang sudah dilatih dan di uji coba kemudian di implementasikan kedalam sistem. Didalam sistem, terdapat 2 jenis input yang akan dibutuhkan, yaitu citra sidik jari yang didapat oleh perangkat keras DigitalPersona U.are.U 4500, dan data asli, atau folder yang akan di enkripsi. Proses dekripsi juga membutuhkan citra sidik jari yang didapat oleh perangkat keras yang sama dengan proses enkripsi, namun proses dekripsi akan membutuhkan *cipher text* yang didapat dari hasil proses enkripsi.

Untuk dapat melakukan enkripsi, pengguna harus mendaftarkan sidik jarinya terlebih dahulu kedalam database lokal sistem. Sidik jari yang didapat dari perangkat keras kemudian di lakukan pemrosesan awal untuk dapat menyamakan bentuk dengan citra yang digunakan untuk pelatihan model. Setelah pemrosesan awal, citra yang telah di proses kemudian di simpan kedalam database lokal, bersama dengan kunci yang dibuat untuk sidik jari tersebut.

Setelah sidik jari terdaftar didalam database, sistem kemudian memiliki kunci yang dapat digunakan untuk proses enkripsi. Kunci ini sifatnya unik, dan hanya akan dapat digunakan lagi jika sidik jari yang tersimpan bersama dengannya cocok dengan sidik jari yang akan digunakan dalam proses dekripsi nantinya.

Dengan melakukan persamaan yang ada pada subbab 2.5, proses enkripsi akan menghasilkan *cipher text*, dimana *cipher text* ini adalah data yang tidak dapat dipahami secara harfiah oleh manusia, sehingga melindungi konten yang dikandung didalamnya. Secara prinsip, konten yang ada didalam *cipher text* adalah data yang digunakan didalam proses enkripsi. Untuk mengubah *cipher text* kembali ke data tersebut, maka pengguna harus melakukan dekripsi.

Untuk dapat melakukan dekripsi, pengguna akan membutuhkan sidik jari yang sama dengan sidik jari yang digunakan dalam proses enkripsi untuk dapat menemukan kunci yang sama dengan kunci yang digunakan dalam proses enkripsi dari database. Sama dengan proses enkripsi pada awalnya, sidik jari yang didapat dari perangkat keras kemudian di lakukan pemrosesan awal. Selanjutnya, sidik jari yang telah di proses tersebut kemudian dilakukan pencocokkan dengan sidik jari yang ada di dalam database sistem.



**Gambar 3.18** Alur kerja aplikasi

Seperti yang dapat dilihat pada Gambar 3.18, didalam sistem yang telah dirancang, model yang telah dikembangkan sebelumnya akan digunakan dalam proses pencocokkan. Didalam proses pencocokkan, model melakukan prediksi kecocokkan antara citra sidik jari yang diberikan oleh pengguna, dan citra sidik jari yang telah tersimpan didalam database.

Jika salah satu dari citra sidik jari yang ada didalam database memiliki tingkat kecocokkan sama atau diatas ambang batas toleransi, maka kunci yang disimpan bersama dengan citra yang tersimpan didalam database akan digunakan untuk proses dekripsi. Jika tidak ada citra sidik jari yang tersimpan didalam database yang memiliki tingkat kecocokkan sama atau diatas ambang batas toleransi dengan citra sidik jari yang diberikan, maka proses dekripsi tidak dapat dilakukan.

Setelah mendapatkan kunci dari proses pencocokkan, kunci tersebut kemudian di uji coba dengan *cipher text* yang akan di dekripsi. Jika kunci yang didapat tidak sesuai dengan *cipher text*, maka dekripsi tidak dapat dilakukan. Jika sesuai, maka dekripsi dapat dilakukan. Proses dekripsi akan menghasilkan data yang sama dari data yang digunakan untuk menghasilkan *cipher text* yang digunakan dalam proses dekripsi, dan yang dihasilkan dari proses enkripsi.

### 3.3 Perancangan Antarmuka Sistem

Tahap ini membahas tentang perancangan antarmuka sistem (*User Interface*) untuk aplikasi biometrik sidik jari untuk pengamanan folder yang diberi nama C-FRA. Sistem akan dibuat dalam bentuk aplikasi desktop sistem operasi Windows. Berikut merupakan rancangan tampilan-tampilan yang ada pada aplikasi C-FRA.

#### 3.3.1 Splash Screen

*Splash screen* merupakan tampilan halaman yang paling awal muncul di saat membuka aplikasi. *Splash screen* digunakan sebagai layar dimana aplikasi memuat komponen pada saat dijalankan. Umumnya, *splash screen* dapat berisi logo, proses memuat, dan informasi lainnya. Rancangan tampilan *splash screen* yang akan digunakan untuk aplikasi C-FRA dapat dilihat pada Gambar 3.19.



**Gambar 3.19** Rancangan tampilan *splash screen*

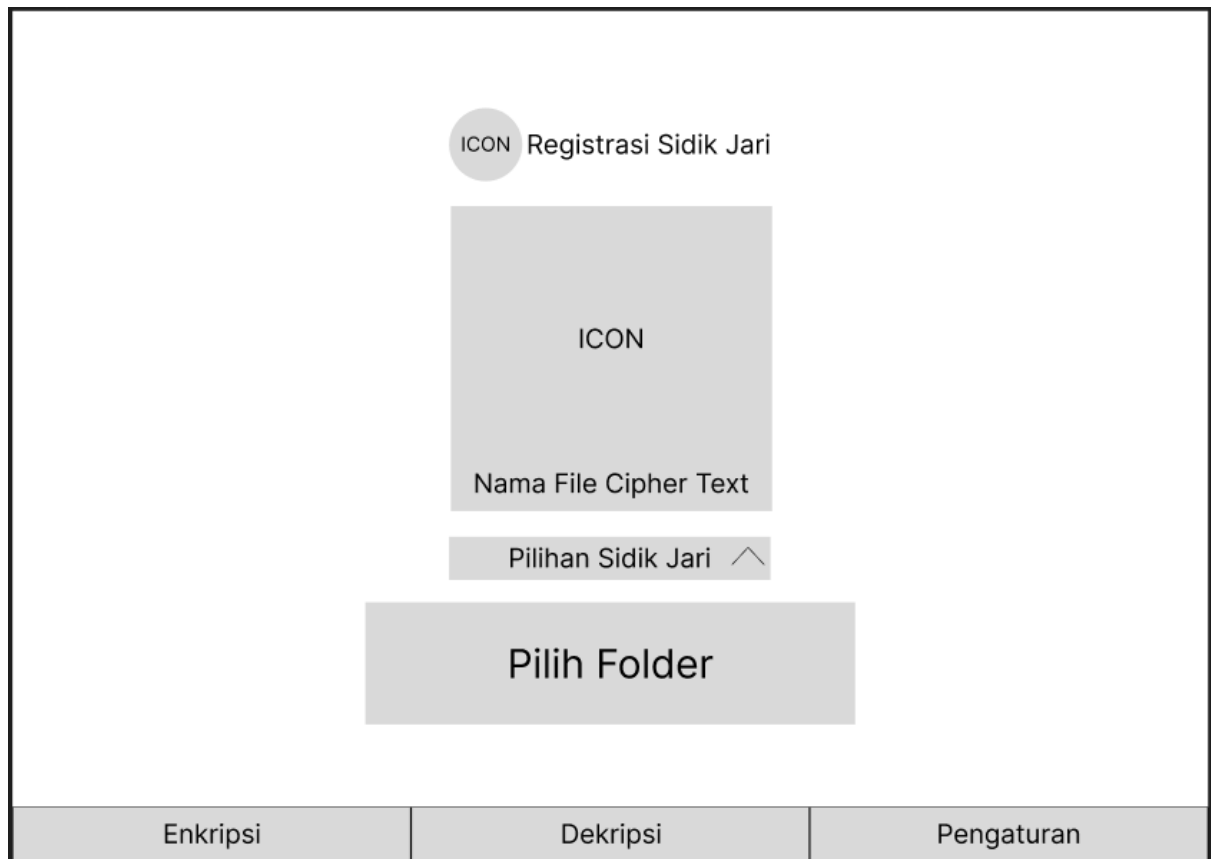
### 3.3.2 Tab Enkripsi

Setelah aplikasi selesai memuat, pengguna kemudian akan melihat halaman tab enkripsi. Aplikasi C-FRA tidak memiliki tampilan utama, dikarenakan aplikasi ini dirancang agar tidak menutupi atau memblokir akses kepada pengguna ketika sebuah proses berjalan. Untuk itu, rancangan antarmuka sistem untuk aplikasi C-FRA menggunakan pendekatan navigasi tab (*Tabbing Navigation*), dimana rancangan ini menggunakan tab dan tidak memiliki proses *step*.

Pada tab enkripsi, terdapat 2 aktivitas yang dapat dilakukan oleh pengguna, yaitu pendaftaran sidik jari, dan enkripsi folder. Sebelum dapat melakukan enkripsi, pertama pengguna harus mendaftarkan sidik jarinya kedalam database lokal aplikasi C-FRA.

Untuk itu, terdapat tombol untuk mendaftarkan sidik jari pada bagian atas halaman. Ketika menekan tombol untuk mendaftarkan sidik jari, sebuah dialog akan muncul untuk melakukan proses tersebut. Setelah mendaftarkan sidik jari, pengguna kemudian dapat memilih sidik jari yang telah didaftar sebelumnya, dan melakukan enkripsi folder dengan memilih tombol pilih pada bagian bawah tampilan. Pada awalnya, tombol untuk memilih folder di non aktifkan, dan dapat di aktifkan dengan memilih sidik jari yang akan digunakan untuk melakukan enkripsi. Setelah memilih folder untuk di enkripsi, pengguna akan dapat melihat 1 elemen tambahan di bagian tengah tampilan. Elemen ini menunjukkan *path* dari folder yang akan di enkripsi. Rancangan tampilan tab enkripsi dapat dilihat pada Gambar 3.20.





**Gambar 3.20** Rancangan tampilan tab enkripsi

### 3.3.3 Dialog Registrasi Sidik Jari

Ketika menekan tombol registrasi sidik jari yang ada pada tab enkripsi, sebuah dialog akan muncul untuk melakukan proses tersebut. Pada tampilan dialog, terdapat 4 elemen yang dapat dilihat pengguna, yaitu judul dialog pada bagian atas, *textbox* dibawah judul dialog untuk menamai sidik jari, kumpulan citra sidik jari yang telah terekam, dan tombol batal dibagian bawah kanan dialog. Pada dialog ini, pengguna akan diminta untuk menempelkan sidik jarinya ke permukaan perangkat keras pemindai sidik jari yang telah terhubung dengan aplikasi. Rancangan tampilan dialog registrasi sidik jari dapat dilihat pada Gambar 3.21.



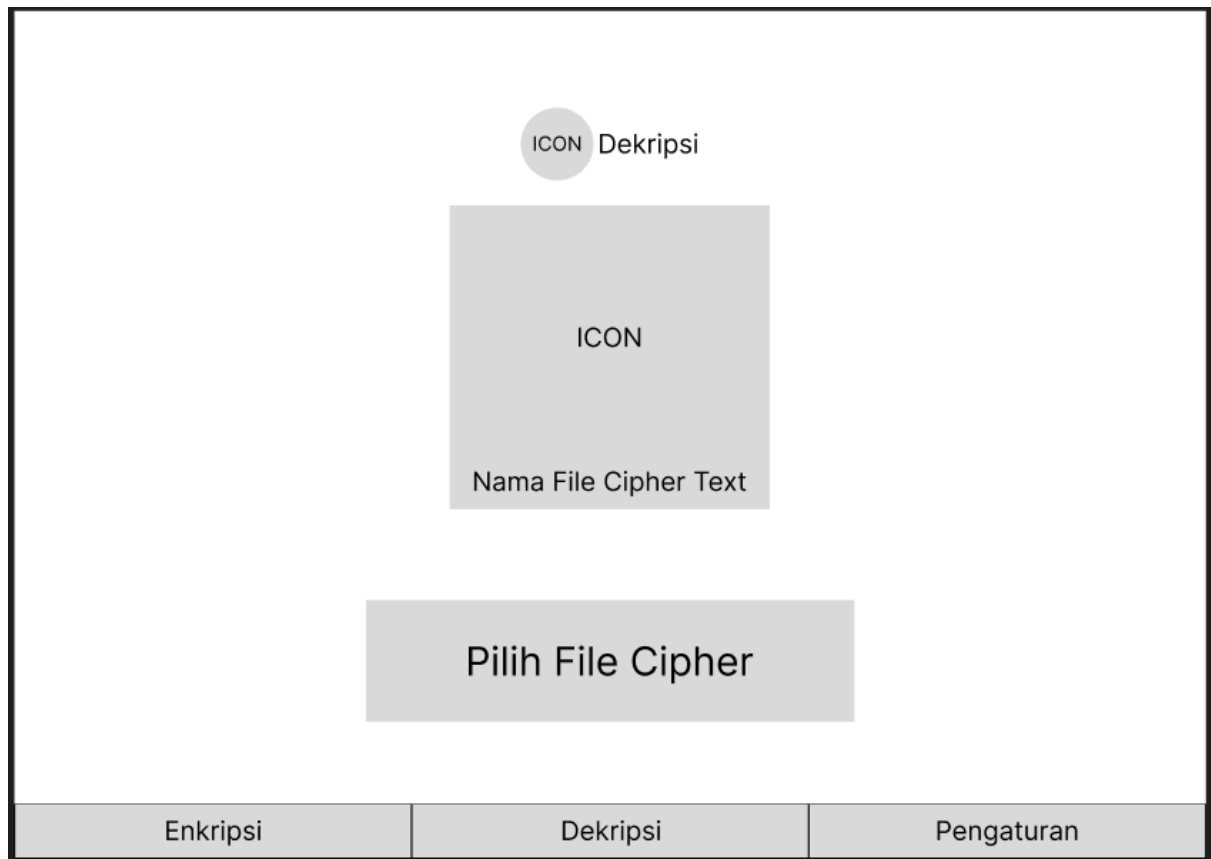
The image shows a registration dialog box titled "Registrasi Sidik Jari". It features a text input field labeled "Nama sidik jari" at the top. Below this, there are four square boxes arranged horizontally, each labeled "Citra 1 Sidik Jari", "Citra 2 Sidik Jari", "Citra 3 Sidik Jari", and "Citra 4 Sidik Jari" respectively. These boxes are intended for capturing fingerprint images. At the bottom right of the dialog, there is a button labeled "Batal" (Cancel).

**Gambar 3.21** Rancangan tampilan dialog registrasi sidik jari

#### 3.3.4 Tab dekripsi

Tab dekripsi memiliki tampilan yang serupa dengan tab enkripsi. Pada tab ini, pengguna dapat melihat 2 elemen, yaitu tombol untuk memilih file *cipher text* pada bagian bawah, dan tombol untuk melakukan dekripsi pada bagian atas.

Pada awalnya, tombol untuk melakukan dekripsi di non aktifkan, dan dapat di aktifkan dengan memilih file *cipher text*. Setelah memilih file *cipher text*, tombol dekripsi dapat di pilih, dan pengguna dapat melihat 1 elemen tambahan di bagian tengah tampilan. Elemen tambahan ini menunjukkan file *cipher text* yang akan di dekripsi. Ketika pengguna menekan tombol dekripsi, dialog dekripsi akan muncul untuk melanjutkan proses tersebut. Rancangan tampilan tab dekripsi dapat dilihat pada Gambar 3.22.



**Gambar 3.22** Rancangan tampilan tab dekripsi

### 3.3.5 Dialog dekripsi

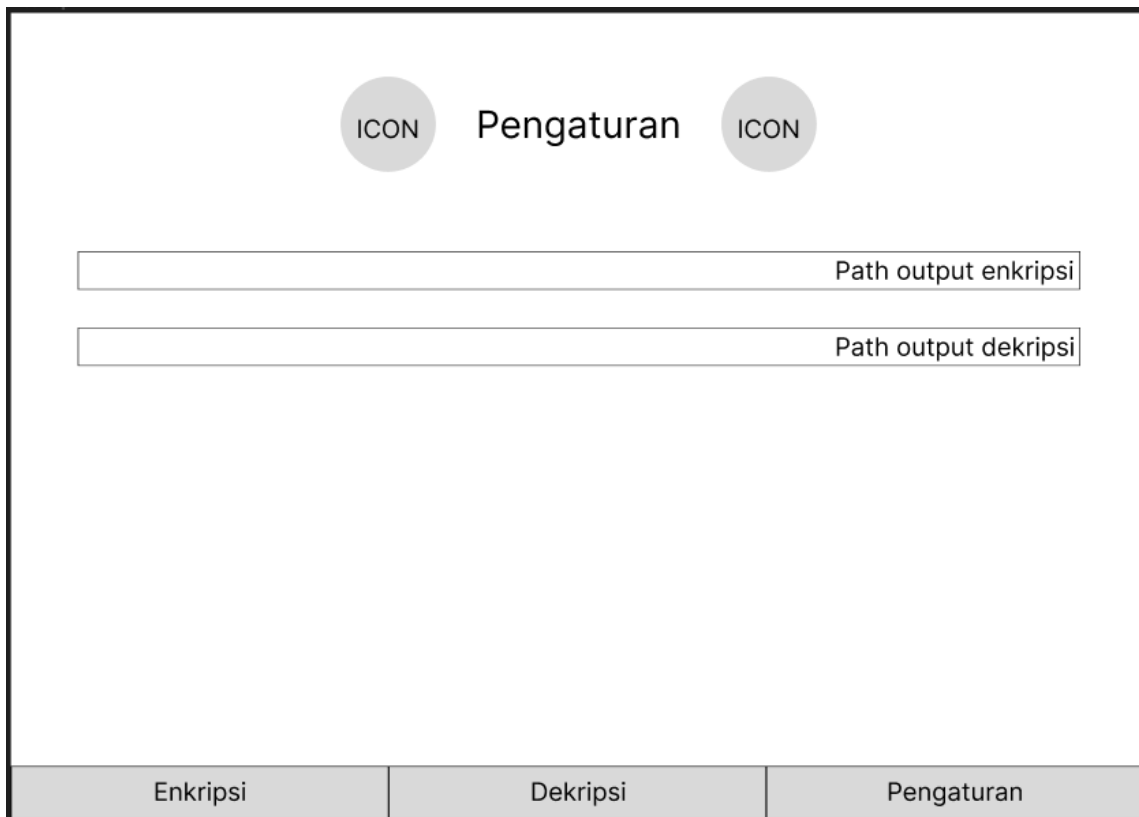
Pada dialog dekripsi, pengguna hanya akan diminta untuk menempelkan sidik jarinya ke permukaan perangkat keras pemindai sidik jari yang telah terhubung dengan aplikasi, dan informasi apakah sidik jari tersebut cocok dengan file *cipher text* yang akan di dekripsi. Sama seperti dialog registrasi sidik jari, pada dialog dekripsi juga terdapat tombol batal pada bagian bawah kanan dialog. Rancangan dialog dekripsi dapat dilihat pada Gambar 3.23.



**Gambar 3.23** Rancangan tampilan dialog dekripsi

### 3.3.6 Tab Pengaturan

Pada tab pengaturan, pengguna dapat melihat 4 elemen, yaitu tombol simpan untuk menyimpan pengaturan yang telah diubah, *textbox path* untuk hasil enkripsi, *textbox path* untuk hasil dekripsi, dan tombol informasi. Tombol simpan hanya akan mengandung ikon saya, dan tidak menggunakan label. Ikon yang digunakan adalah ikon yang telah dikenal secara umum sebagai ikon untuk menyimpan. Begitu juga dengan tombol informasi, tombol tersebut hanya akan mengandung ikon dan tidak menggunakan label. Ketika menekan tombol informasi, akan muncul sebuah dialog, dimana kontennya sama seperti pada Gambar 3.19, hanya saja kali ini konten tersebut ada didalam dialog. Rancangan tampilan tab pengaturan dapat dilihat pada Gambar 3.24, dan rancangan dialog informasi dapat dilihat pada Gambar 3.25.



ICON Pengaturan ICON

Path output enkripsi

Path output dekripsi

Enkripsi Dekripsi Pengaturan

**Gambar 3.24** Rancangan tampilan tab pengaturan



Judul Penelitian

LOGO

Nama Peneliti

NIM Peneliti

Tutup

**Gambar 3.25** Rancangan tampilan dialog informasi

## **BAB IV**

### **IMPLEMENTASI DAN PENGUJIAN SISTEM**

#### **4.1 Implementasi Sistem**

Implementasi *fingerprint recognition* pada keamanan folder menggunakan algoritma *Convolutional Neural Network* dibangun dan di implementasikan menjadi sebuah sistem, dimana proses ini membutuhkan perangkat keras, perangkat lunak, dan data pendukung. Adapun spesifikasi dari perangkat dan data yang akan digunakan didalam penelitian ini adalah sebagai berikut.

##### **4.1.1 Perangkat Keras**

Penelitian ini melakukan proses pengembangan model untuk dapat membedakan sidik jari. Untuk melakukan proses tersebut, perangkat keras yang dibutuhkan membutuhkan spesifikasi yang cukup mumpuni untuk dapat mempercepat proses pelatihan, sehingga mempermudah uji coba pelatihan model dengan berbagai parameter dalam waktu yang singkat. Adapun spesifikasi dari perangkat keras yang digunakan untuk mengembangkan dan melatih model, serta mengembangkan sistem pada penelitian ini adalah:

- Processor AMD Ryzen™ 7 5800X @ 3.8GHz
- GPU Nvidia GeForce RTX™ 3080 Ti @ 1.67GHz 12Gb VRAM
- RAM DDR4 Corsair Vengeance® @ 32Gb 3600MHz
- SSD M.2 NVMe @ 256Gb
- Fingerprint Scanner DigitalPersona U.are.U 4500

##### **4.1.2 Perangkat Lunak**

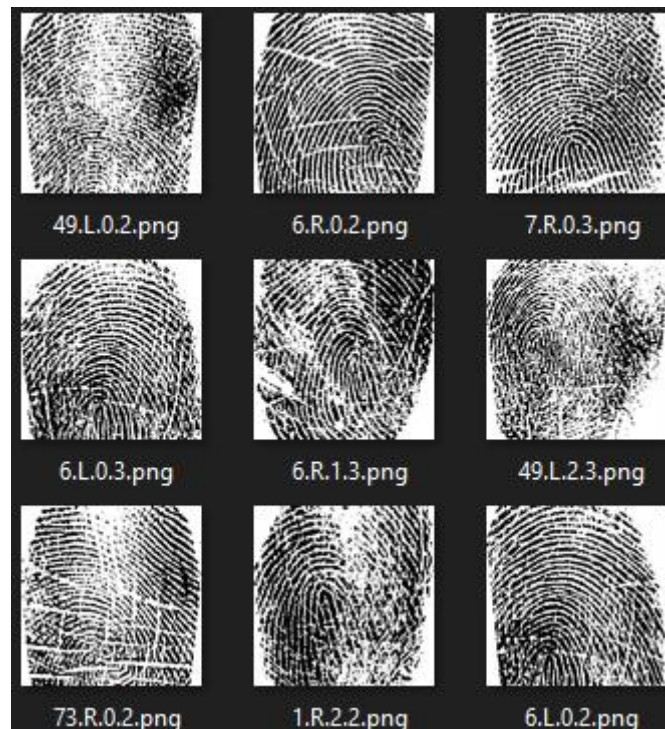
Untuk mengembangkan model dan sistem, dibutuhkan alat berupa perangkat lunak yang akan digunakan dengan perangkat keras yang telah disebutkan. Adapun spesifikasi dari perangkat lunak tersebut yang akan digunakan untuk mengembangkan model dan sistem didalam penelitian ini adalah:

- Visual Studio Code
- TensorFlow v2.14.0
- TensorFlow.js v4.11.0
- Node.js v18.10.0
- TypeScript v4.9.3
- Electron.js v24.1.2
- Angular v15.0.0

- Windows 10 64-bit Pro Build 19045
- Linux Ubuntu 64-bit 22.04.3

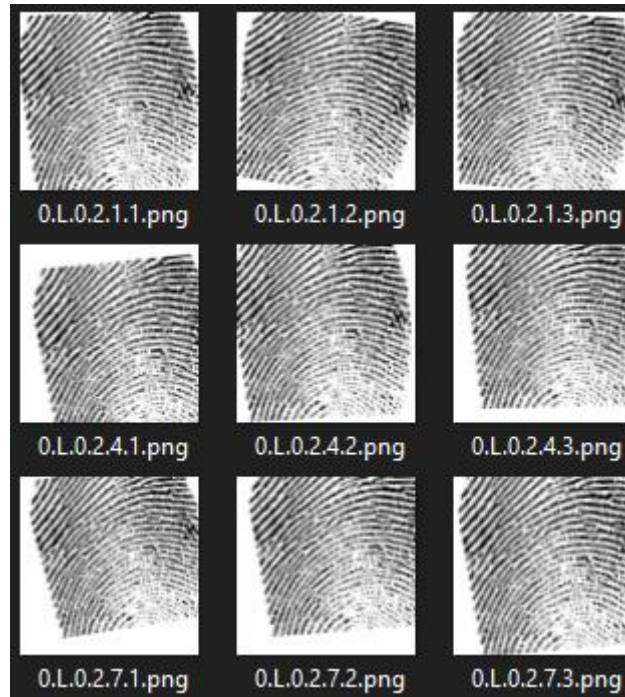
#### 4.1.3 Data Pendukung

Seperti yang telah disebutkan pada subbab 3.2.1 dan subbab 3.2.2, data yang akan digunakan didalam penelitian ini didapatkan dari 74 responden laki-laki dan perempuan dengan jarak usia antara 12 tahun sampai dengan 71 tahun. Setiap responden memiliki 10 jari yang di ambil 3 kali, menghasilkan citra sidik jari sejumlah 2220 citra. Citra tersebut kemudian dilakukan pemrosesan awal untuk dapat memperjelas fitur yang ingin dipelajari oleh model. Adapun contoh dari kumpulan citra asli yang telah di proses dapat dilihat pada Gambar 4.1.



**Gambar 4.1** Contoh dari kumpulan citra sidik jari yang telah dilakukan pemrosesan awal

Kumpulan dari citra yang telah di proses tersebut kemudian di augmentasikan lebih lanjut. Setiap citra di augmentasikan sebanyak 35 kali, menghasilkan data citra teraugmentasi sejumlah 77700 citra. Adapun contoh dari kumpulan citra teraugmentasi dapat dilihat pada Gambar 4.2.



**Gambar 4.2** Contoh dari kumpulan citra sidik teraugmentasi

Secara keseluruhan, citra sidik jari yang digunakan didalam penelitian ini berjumlah 79920 citra. Setiap jari adalah kelas, maka kelas yang digunakan didalam penelitian ini berjumlah 740 kelas. Format penamaan kelas dapat dilihat pada subbab 3.2.3. Setiap kelas memiliki citra sebanyak 108, dengan jumlah 3 citra asli, dan 105 citra teraugmentasi.

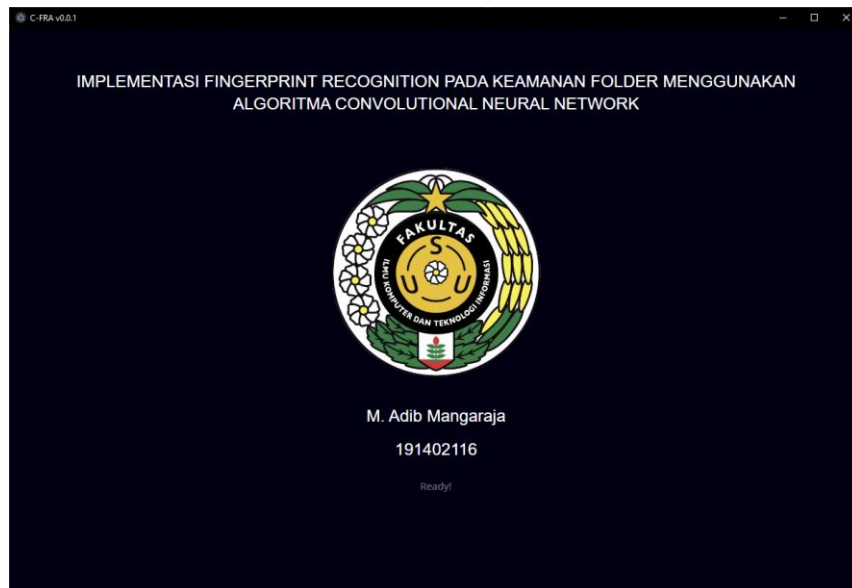
## **4.2 Implementasi Rancangan Antarmuka**

Subbab ini akan membahas tentang implementasi dari rancangan antarmuka yang telah dibahas pada subbab 3.3. Implementasi rancangan didalam penelitian ini menggunakan *framework* Electron.js dan Angular, dimana kedua *framework* tersebut menggunakan bahasa pemrograman JavaScript dan TypeScript. Hasil akhir dari implementasi rancangan antarmuka berupa aplikasi desktop pada platform sistem operasi Windows.

### **4.2.1 Splash Screen**

Halaman *splash screen* adalah halaman yang pertama kali muncul ketika aplikasi diluncurkan. Sesuai dengan rancangan pada subbab 3.3.1, implementasi halaman *splash screen* dapat dilihat pada Gambar 4.3.

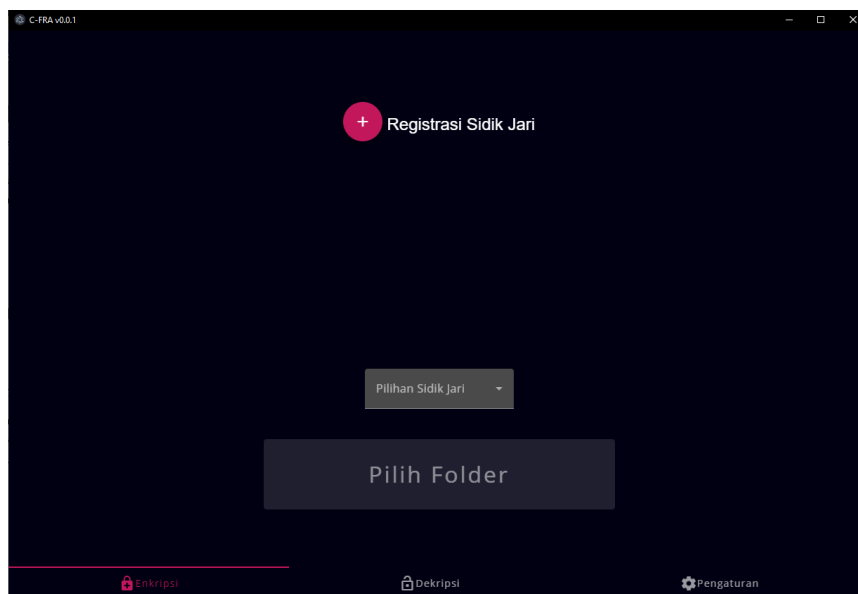




**Gambar 4.3** Tampilan dari halaman *splash screen*

#### 4.2.2 Tab Enkripsi

Sesuai dengan apa yang telah dijelaskan pada subbab 3.3.2, setelah aplikasi telah selesai memuat semua komponen yang diperlukan dan menjalankan semua proses, pengguna kemudian akan melihat tampilan tab enkripsi.



**Gambar 4.4** Tampilan dari halaman tab enkripsi

Dalam halaman ini, pengguna akan melihat sebuah tombol besar berwarna abu-abu kehitaman. Sesuai dengan penjelasan pada subbab 3.3.2, tombol ini di non aktifkan, dan akan aktif ketika pengguna sudah memilih sidik jari yang akan digunakan untuk meng enkripsi folder yang akan dipilih nantinya. Pengguna dapat mengetahui informasi ini dengan mengarahkan kursor ke atas

tombol tersebut. Ketika kursor berada di atas tombol, maka sebuah *tooltip* akan muncul, seperti pada Gambar 4.5.



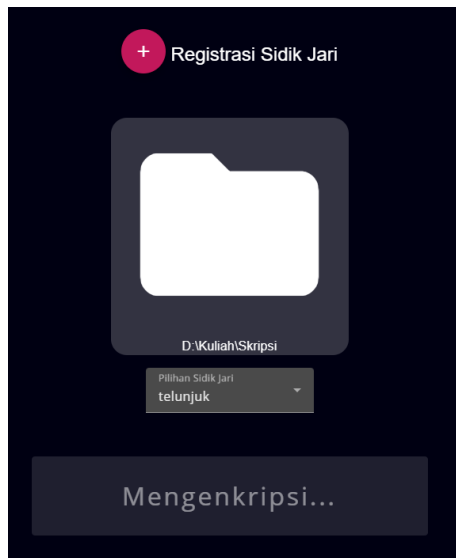
**Gambar 4.5** *Tooltip* ketika kursor diarahkan pada tombol non aktif “Pilih Folder”

Jika tidak ada sidik jari yang terdaftar didalam database lokal, maka ketika pengguna membuka *list* pilihan sidik jari, pengguna akan melihat sebuah pilihan non aktif untuk menginformasikan akan hal tersebut, seperti yang dapat dilihat pada Gambar 4.6.



**Gambar 4.6** *List* pilihan sidik jari ketika tidak ada sidik jari terdaftar didalam database

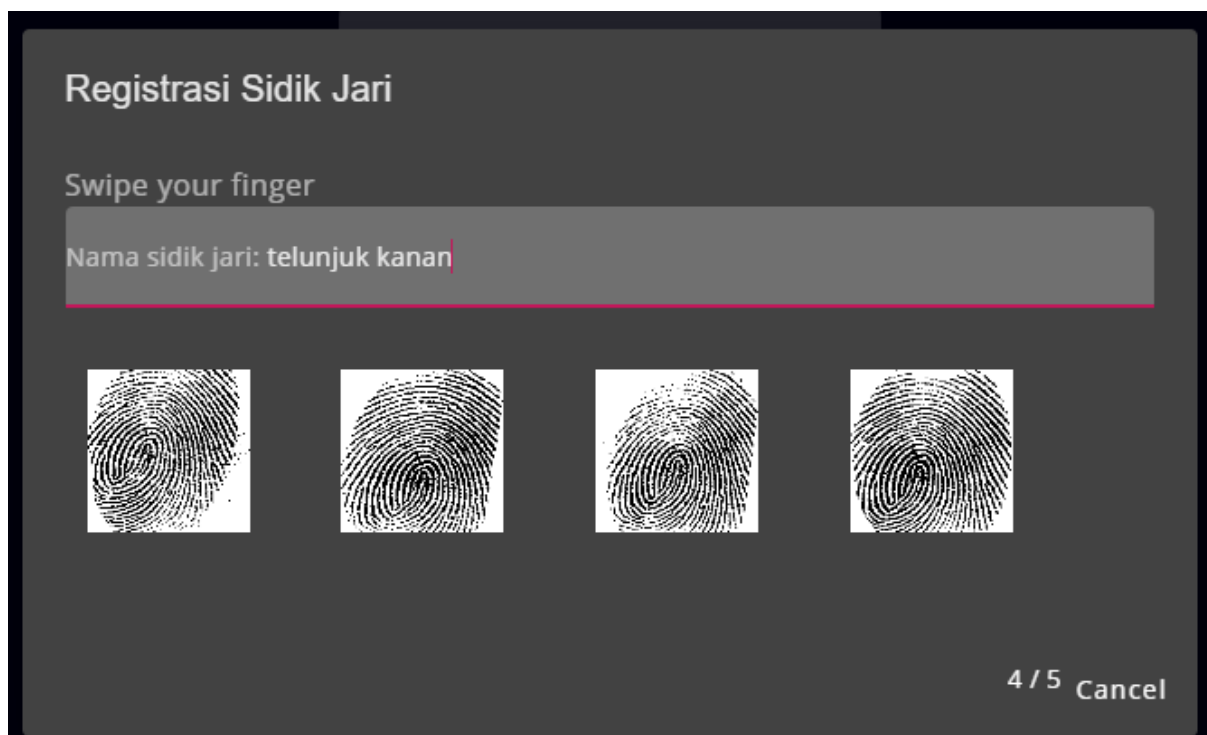
Untuk mendaftarkan sidik jari baru kedalam database, pengguna dapat mengklik tombol dengan ikon tambah dan label “Registrasi Sidik Jari” pada bagian atas halaman tab enkripsi. Setelah mendaftarkan sebuah sidik jari, pengguna dapat memilih sidik jari yang telah terdaftar dari *list* pilihan sidik jari, dan tombol “Pilih Folder” akan menjadi aktif. Setelah memilih sebuah folder, pengguna dapat melihat elemen baru di bagian tengah halaman tab enkripsi, dan proses enkripsi dimulai.



**Gambar 4.7** Konten dari tab enkripsi ketika enkripsi dimulai

#### 4.2.3 Dialog Registrasi Sidik Jari

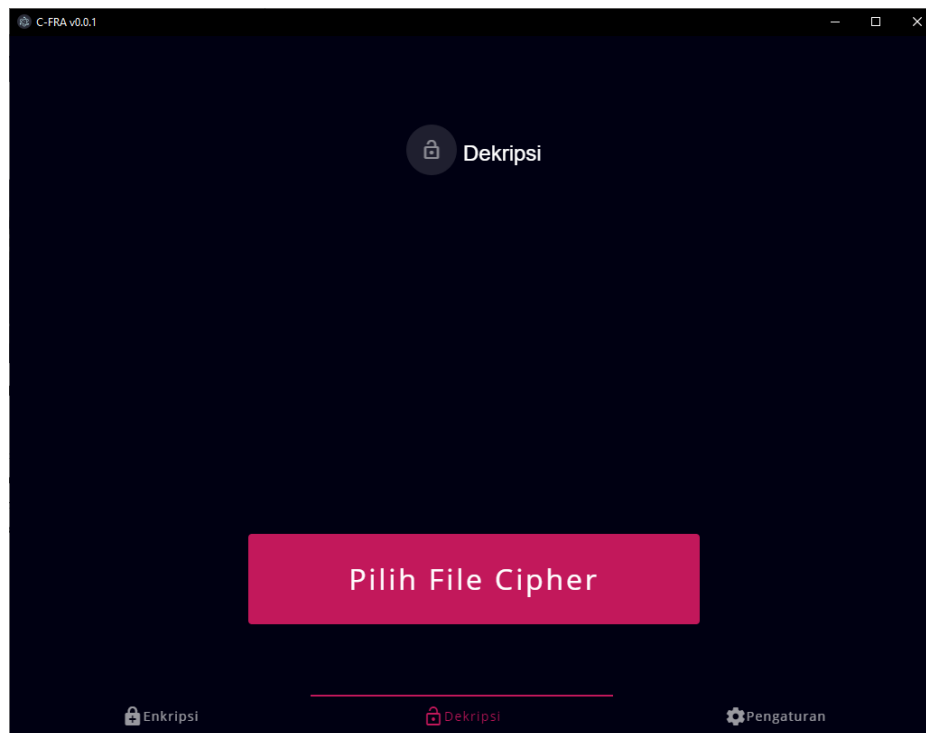
Pada dialog ini, pengguna akan diminta untuk menempelkan sidik jarinya ke permukaan perangkat keras pemindai sidik jari untuk mendapatkan citra sidik jari yang akan didaftarkan kedalam database sistem. Sesuai dengan rancangan yang telah dibahas pada subbab 3.3.3, hasil implementasi dari dialog registrasi sidik jari dapat dilihat pada Gambar 4.8.



**Gambar 4.8** Tampilan dialog registrasi sidik jari

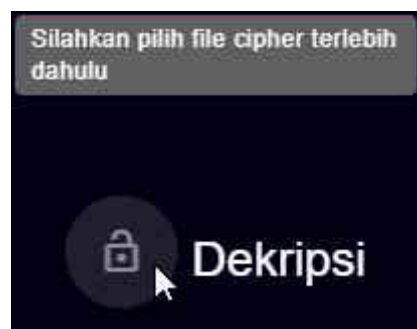
#### 4.2.4 Tab Dekripsi

Tab dekripsi memiliki tampilan yang serupa dengan tab enkripsi, namun memiliki posisi alur kerja yang berkebalikan. Implementasi rancangan tab dekripsi dapat dilihat pada Gambar 4.9.



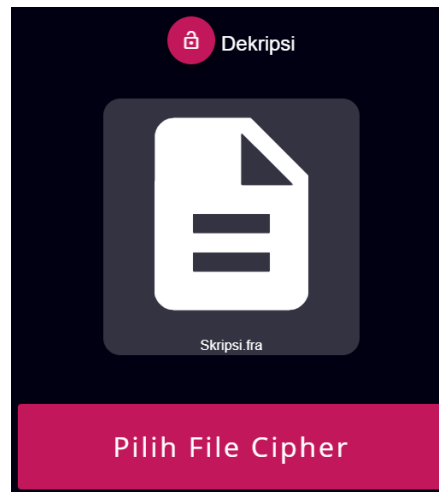
**Gambar 4.9** Tampilan dari halaman tab dekripsi

Didalam tab dekripsi, pengguna dapat melihat 2 elemen pada bagian atas dan bawah halaman, yaitu tombol untuk memilih file *cipher*, dan tombol untuk memulai dekripsi. Sebelum dekripsi dapat dimulai, file *cipher* harus dipilih terlebih dahulu. Oleh karena itu, tombol untuk memulai dekripsi di non aktifkan sampai file *cipher* dipilih terlebih dahulu. Sama seperti tombol “Pilih Folder” pada tab enkripsi, pengguna dapat mengetahui informasi ini dengan mengarahkan kursor ke atas tombol tersebut. Ketika kursor berada di atas tombol, maka sebuah *tooltip* akan muncul, seperti pada Gambar 4.10.



**Gambar 4.10** *Tooltip* ketika kursor diarahkan pada tombol non aktif “Dekripsi”

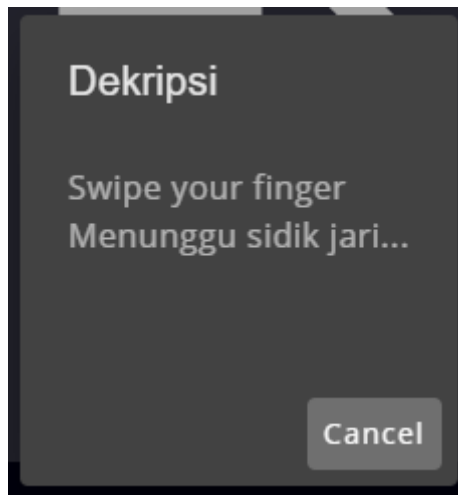
Setelah file *cipher* dipilih, tombol dekripsi akan menjadi aktif, dan pengguna dapat melihat elemen baru di bagian tengah halaman tab dekripsi. Berbeda dengan tab enkripsi, pada tab dekripsi proses tidak akan langsung dimulai setelah memilih file *cipher*. Untuk memulai proses tersebut, pengguna harus mengklik tombol dekripsi yang telah aktif setelah memilih file *cipher*. Rancangan ini dibuat dengan alasan ketika dekripsi, sebuah dialog akan muncul, membuat akses komponen lainnya terblokir sampai proses dari dialog dekripsi selesai atau dibatalkan.



**Gambar 4.11** Konten dari tab dekripsi ketika file *cipher* telah dipilih

#### 4.2.5 Dialog Dekripsi

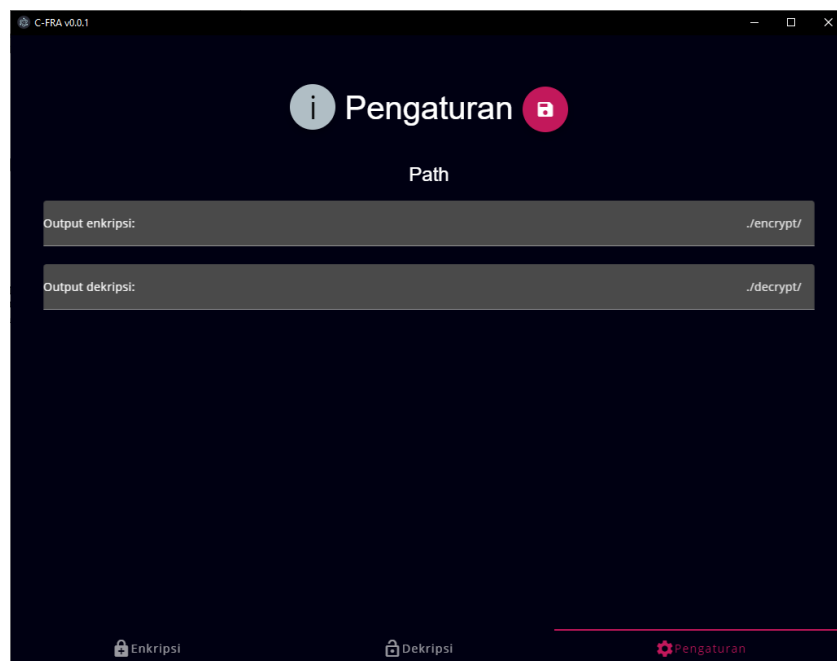
Ketika pengguna telah memilih file *cipher* pada tab dekripsi, untuk melakukan dekripsi pengguna harus mengklik tombol dekripsi yang telah aktif pada bagian atas halaman tab dekripsi. Ketika pengguna mengklik tombol dekripsi tersebut, dialog dekripsi akan muncul untuk membandingkan sidik jari yang diberikan oleh pengguna dengan sidik jari yang tersimpan dan digunakan untuk meng enkripsi data menjadi file *cipher* yang telah dipilih. Sesuai dengan rancangan pada subbab 3.3.5, implementasi rancangan dialog dekripsi dapat dilihat pada Gambar 4.12.



**Gambar 4.12** Tampilan dialog dekripsi

#### 4.2.6 Tab Pengaturan

Pada tab pengaturan, pengguna dapat mengatur 2 pengaturan, yaitu *path* dari output enkripsi dan dekripsi. Sesuai dengan rancangan pada subbab 3.3.6, implementasi rancangan tab pengaturan dapat dilihat pada Gambar 4.13.



**Gambar 4.13** Tampilan dari halaman tab pengaturan

Ketika pengguna mengubah suatu pengaturan, pengaturan tersebut tidak akan di gunakan atau di simpan sebelum pengguna menyimpan pengaturan. Untuk menyimpan pengaturan, pengguna dapat mengklik tombol simpan. Seperti yang telah dijelaskan pada subbab 3.3.6, tombol simpan dan tombol informasi hanya akan mengandung ikon.

Ikons dipilih berdasarkan pengetahuan umum, dimana ikon dengan gambar diskette adalah ikon simpan, dan ikon dengan gambar huruf “i” adalah ikon informasi.

Pada tab dialog informasi, pengguna dapat melihat konten yang sama seperti konten yang dikandung pada halaman *splash screen*.



**Gambar 4.14** Tampilan dialog informasi

### 4.3 Prosedur Operasional

Aplikasi C-FRA memiliki 4 tampilan, yaitu *splash screen*, tab enkripsi, tab dekripsi, dan tab pengaturan. Aplikasi C-FRA juga memiliki 3 dialog, yaitu dialog registrasi sidik jari, dialog dekripsi, dan dialog informasi. Ketika aplikasi diluncurkan, tampilan pertama yang pengguna akan lihat adalah *splash screen*. Setelah aplikasi selesai memuat, pengguna akan melihat tab enkripsi.

Untuk dapat melakukan dekripsi, pengguna akan membutuhkan file *cipher* yang didapatkan dari proses enkripsi. Untuk melakukan enkripsi, pengguna harus mendaftarkan sidik jarinya terlebih dahulu. Dan untuk mendaftarkan sidik jari, pengguna dapat mengklik tombol registrasi sidik jari dibagian atas halaman tab enkripsi, dimana dialog registrasi sidik jari akan muncul untuk menjalankan proses tersebut.

Setelah pengguna mendaftarkan sidik jarinya, pengguna kemudian dapat memilih sidik jari yang telah didaftar sebelumnya, dan folder yang akan di amankan untuk di enkripsi. Setelah enkripsi selesai, pengguna akan mendapatkan file *cipher* yang di simpan di folder dengan *path* output enkripsi pada tab pengaturan.

Untuk melakukan dekripsi, pengguna dapat beralih ke tab dekripsi, dimana tab ini terletak di bagian bawah aplikasi. Pada tab dekripsi, pengguna dapat memilih file *cipher* yang akan di dekripsi, dimana file *cipher* ini didapatkan dari proses enkripsi sebelumnya. Setelah memilih file *cipher* untuk di dekripsi, pengguna kemudian dapat memulai proses dekripsi dengan mengklik tombol dekripsi di bagian atas halaman tab dekripsi.

Ketika pengguna mengklik tombol dekripsi, dialog dekripsi akan muncul untuk menjalankan proses tersebut. Didalam dialog dekripsi, pengguna akan diminta untuk memberikan sidik jarinya dengan menempelkan sidik jari ke permukaan perangkat keras pemindai yang tersambung dengan sistem. Jika sidik jari yang diberikan tidak sama dengan sidik jari yang dipilih untuk menghasilkan file *cipher* yang akan di dekripsi, maka pengguna akan diberi tahuhan melalui dialog dekripsi. Jika sidik jari sudah sesuai, maka proses dekripsi akan berlangsung, dan hasil dekripsi akan di simpan di folder dengan *path* output dekripsi pada tab pengaturan.

#### **4.4 Pelatihan Sistem**

Pelatihan sistem dilakukan dengan menggunakan algoritma *Convolutional Neural Network* dengan model siamese, atau disebut juga model kembar. Pelatihan sistem dilakukan menggunakan data berjumlah 79920 citra, dimana 2220 citra adalah citra asli, dan 77700 citra adalah citra teraugmentasi. Setiap citra asli di augmentasikan sebanyak 35 kali, dimana setiap 5 augmentasi menggunakan kombinasi teknik augmentasi yang berbeda.

Pelatihan sistem menggunakan 3 jenis data yang dibagi dari 79920 jumlah citra. Jenis data tersebut adalah data training, data validation, dan data testing. Pembagian data memiliki perbandingan 8 : 1 : 1, dimana 8 adalah data training, dan 1 adalah data validation dan data testing. Hasil dari pembagian data dapat dilihat pada Tabel 4.1.



**Tabel 4.1** Hasil pembagian data

No.	Data	Data Training	Data Validation	Data Testing	Jumlah
1.	Citra asli	1776	222	222	2220
2.	Citra teraugmentasi	62160	7770	7770	77700
Jumlah Keseluruhan Data		63936	7992	7992	79920

Penelitian ini berfokus pada 2 parameter pelatihan, yaitu parameter *epoch* dan *batch size*. Kedua parameter tersebut memiliki dampak yang besar terhadap pelatihan dan performa model. Parameter *epoch* menandakan berapa kali model dilatih pada dataset training, menyeimbangkan performa model antara *underfitting* dan *overfitting* dengan membuat model mempelajari pola data secara efektif tanpa harus mengingat fitur yang diberikan. Kemudian dari pada itu, parameter *batch size* mempengaruhi efisiensi dan penyatuan berat dalam proses optimisasi dengan menentukan jumlah contoh pelatihan yang diproses di setiap iterasi *epoch*. Pelatihan sistem dilakukan sebanyak 8 kali, dimana jumlah tersebut didapatkan berdasarkan kombinasi dari jumlah *epoch* dan *batch size*, dengan jumlah *epoch* adalah 15, 20, 25, dan 30, dan jumlah *batch size* adalah 16, dan 32. Sebelumnya, peneliti ingin menggunakan *batch size* lebih besar dari 32, yaitu 64 dan diatasnya. Namun, pelatihan sistem menggunakan *batch size* 64 menimbulkan error *Out Of Memory*. Jumlah *epoch* di batasi sampai dengan 30, dikarenakan model sudah mulai *overfitting* dengan perbedaan 5% pada *training accuracy* (acc) dan *validation accuracy* (val\_acc). Hasil dari kedelapan uji coba pelatihan dapat dilihat pada Tabel 4.2.

**Tabel 4.2** Hasil uji coba pelatihan sistem

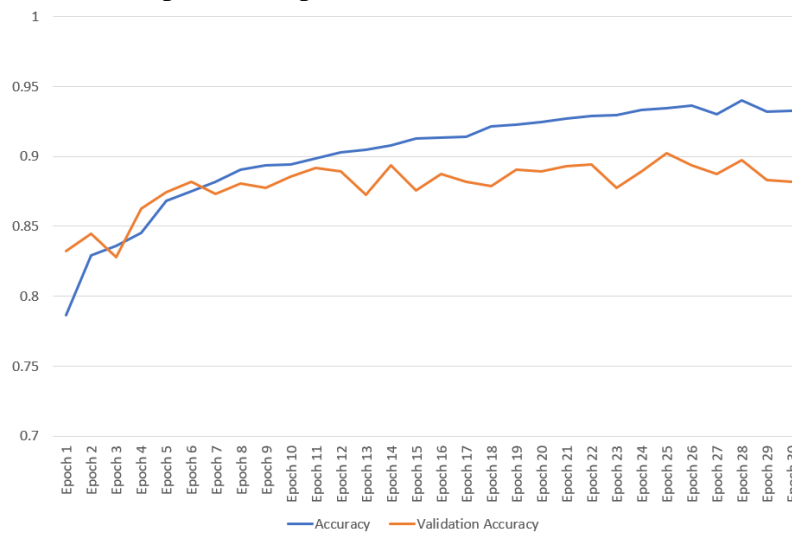
No.	<i>Epoch</i>	<i>Batch Size</i>	acc	val_acc	Best val_Acc
1.	15	16	90.25%	89.06%	89.06% @ epoch 15
2.	15	32	89.75%	88.21%	88.84% @ epoch 13
3.	20	16	91.49%	89.15%	89.15% @ epoch 20
4.	20	32	91.21%	88.56%	89.02% @ epoch 17
5.	25	16	92.48%	87.95%	89.06% @ epoch 24
6.	25	32	93.51%	89.79%	89.94% @ epoch 22
No.	<i>Epoch</i>	<i>Batch Size</i>	acc	val_acc	Best val_Acc
7.	30	16	93.25%	88.15%	90.21% @ epoch 25
8.	30	32	94.72%	89.44%	89.71% @ epoch 28

Berdasarkan hasil dari uji coba pelatihan pada Tabel 4.2, didapatkan hasil val\_acc tertinggi pada percobaan ke-7 *epoch* ke 25, yaitu sebesar 90.21%. Adapun hasil lengkap dari percobaan ke-7 dapat dilihat pada Tabel 4.3.

**Tabel 4.3** Hasil lengkap uji coba pelatihan ke-7

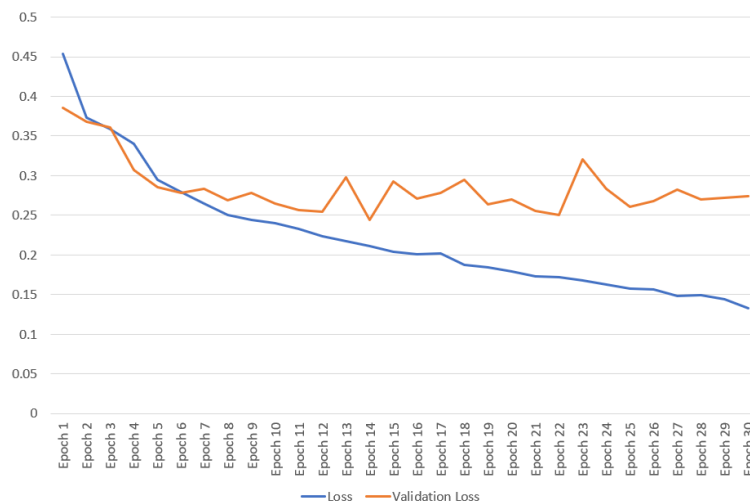
Epoch	Loss	Accuracy	Validation loss	Validation accuracy	Time
1	0.4538	0.7864	0.3851	0.8321	141s
2	0.3733	0.8291	0.3685	0.8444	117s
3	0.3592	0.8362	0.3605	0.8280	108s
4	0.3406	0.8455	0.3070	0.8626	107s
5	0.2947	0.8684	0.2857	0.8746	107s
6	0.2792	0.8748	0.2778	0.8820	112s
7	0.2648	0.8821	0.2835	0.8729	115s
8	0.2499	0.8903	0.2685	0.8803	117s
9	0.2438	0.8936	0.2780	0.8773	119s
10	0.2397	0.8939	0.2652	0.8855	121s
11	0.2333	0.8985	0.2562	0.8918	129s
12	0.2231	0.9028	0.2546	0.8895	131s
13	0.2177	0.9044	0.2982	0.8723	148s
14	0.2116	0.9075	0.2446	0.8937	133s
15	0.2041	0.9130	0.2928	0.8753	140s
16	0.2009	0.9134	0.2705	0.8873	135s
17	0.2023	0.9138	0.2778	0.8820	129s
18	0.1876	0.9212	0.2943	0.8787	118s
19	0.1845	0.9226	0.2641	0.8906	111s
20	0.1792	0.9244	0.2696	0.8891	118s
21	0.1733	0.9268	0.2558	0.8927	123s
22	0.1718	0.9286	0.2501	0.8939	122s
23	0.1679	0.9296	0.3201	0.8777	121s
24	0.1623	0.9329	0.2839	0.8893	129s
25	0.1573	0.9345	0.2608	0.9021	130s
26	0.1563	0.9364	0.2679	0.8935	138s
27	0.1486	0.9303	0.2828	0.8873	138s
28	0.1492	0.9397	0.2704	0.8971	139s
29	0.1442	0.9320	0.2720	0.8832	137s
30	0.1332	0.9325	0.2738	0.8815	134s

Visualisasi tabel tersebut dapat dilihat pada Gambar 4.15 dan Gambar 4.16.



**Gambar 4.15** Visualisasi akurasi dari hasil pelatihan system

Pada Gambar 4.15 dapat dilihat garis akurasi dimulai lebih rendah dari validasinya, yang kemudian validasi akurasi mengalami penurunan dan melewati akurasi pada *epoch* ke-3, dan naik kembali diatas akurasi pada *epoch* ke-4. Namun, akurasi melewati validasi akurasi pada *epoch* ke-7, dan validasi akurasi mulai stagnan dari *epoch* ke-6. Sedangkan akurasi terus menerus menaik sedikit demi sedikit dimulai dari *epoch* ke-7. Ini menandakan bahwa model sudah mulai *overfitting* dengan kenaikan validasi akurasi yang didapat sangat insignifikan di setiap *epoch* setelah *epoch* ke-7.



**Gambar 4.16** Visualisasi loss dari hasil pelatihan system

Pada Gambar 4.16, visualisasi loss dan validasinya serupa dengan visualisasi akurasi dengan validasinya, dimana validasi loss mulai stagnan dari *epoch* ke-6, dan loss terus mengalami penurunan, meninggalkan dan menjauhi validasi loss di setiap *epoch* setelah *epoch* ke-6.

#### 4.5 Pengujian Sistem

















Subbab ini akan menjelaskan hasil dari pengujian sistem yang telah dilatih pada subbab sebelumnya. Pengujian sistem dilakukan dengan memberikan 2 citra sidik jari kepada model, mencatat hasil dari prediksi yang diberikan oleh model, dan membandingkannya dengan nilai yang sebenarnya.

Data yang akan digunakan berjumlah 7992, dimana nilai ini didapat dari 10% dari total keseluruhan data. Jumlah kelas adalah 74, dimana kelas ini tidak digunakan didalam pelatihan model, dengan demikian dapat diartikan bahwa citra sidik jari yang akan digunakan untuk pengujian sistem bersifat unik dan tidak pernah dilihat oleh model sebelumnya.





Sesuai dengan penjelasan pada subbab 3.2.4, hasil akhir yang diberikan oleh model adalah bilangan *float* dengan jarak antara 0.0 dan 1.0. Hasil ini didapatkan dari lapisan terakhir pada model klasifikasi, yaitu lapisan *dense* dengan aktivasi *sigmoid*. Bilangan *float* yang didapat dari model dapat diartikan sebagai persentase kecocokkan dari kedua citra sidik jari yang diberikan ke model, dimana semakin tinggi nilai tersebut, maka semakin cocok kedua citra yang diberikan.

Namun, hasil yang diharapkan didalam penelitian ini adalah nilai boolean, dimana model diharapkan untuk dapat memberikan jawaban antara “cocok” dan “tidak cocok”. Oleh karena itu, dibutuhkan nilai ambang batas, dimana jika nilai yang diberikan oleh model berada dibawah nilai ambang batas, maka dapat dipastikan bahwa model memberikan jawaban “tidak cocok”, dan jika nilai tersebut sama dengan atau diatas nilai ambang batas, maka model memberikan jawaban “cocok”. Nilai dari ambang batas yang digunakan didalam pengujian sistem adalah 0.7. Nilai ini didapat dari analisa sensitivitas model melalui pengujian, dimana didalam pengujian tersebut menunjukkan bahwa model memberikan akurasi terbaik secara keseluruhan ketika menggunakan nilai ambang batas 0.7. Adapun beberapa contoh hasil dari pengujian sistem dapat dilihat didalam Tabel 4.4.

**Tabel 4.4** Beberapa contoh hasil pengujian sistem

No.	Citra input	Citra pembanding	Nilai	Prediksi model	Klasifikasi prediksi	Status
1.			Cocok	0.89	Cocok	Benar
2.			Cocok	0.68	Tidak cocok	Salah
3.			Cocok	0.98	Cocok	Benar
4.			Cocok	0.79	Cocok	Benar
5.			Cocok	0.71	Cocok	Benar
6.			Tidak cocok	0.02	Tidak cocok	Benar
7.			Tidak cocok	0.31	Tidak cocok	Benar
8.			Tidak cocok	0.28	Tidak cocok	Benar

**Tabel 4.4** Beberapa contoh hasil pengujian sistem (Lanjutan)

No.	Citra input	Citra pembanding	Nilai	Prediksi model	Klasifikasi prediksi	Status
9.			Tidak cocok	0.22	Tidak cocok	Benar
10.			Tidak cocok	0.71	Cocok	Salah

Tabel 4.4 merupakan 10 contoh hasil pengujian sistem. Pengujian keseluruhan dilakukan sebanyak 222 kali, dimana nilai ini didapat dari total kelas yang digunakan untuk pengujian sistem dikalikan dengan jumlah pengambilan citra asli, yaitu sebanyak 3 kali. Hasil keseluruhan pengujian dapat dilihat pada tabel *confusion matrix*, yang dapat dilihat pada Tabel 4.5.

**Tabel 4.5** *Confusion Matrix* dari hasil pengujian sistem

	Nilai Positif	Nilai Negatif
Prediksi Positif	98	9
Prediksi Negatif	13	102

Berdasarkan Tabel 4.5, nilai dari *True Positive* (TP) adalah 98, *True Negative* (TN) adalah 102, *False Positive* (FP) adalah 9, dan *False Negative* (FN) adalah 13. Setelah mendapatkan nilai-nilai tersebut, didapatkan nilai-nilai indikasi dengan rumus yang tertera pada subbab 2.11 sebagai berikut.

1) *Accuracy*

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{98 + 102}{98 + 102 + 9 + 13} = 90.09\%$$

2) *Precision*

$$Precision = \frac{TP}{TP + FP} = \frac{98}{98 + 9} = 91.58\%$$

3) *Recall*

$$Recall = \frac{TP}{TP + FN} = \frac{98}{98 + 13} = 88.28\%$$

4) *Specificity*

$$Specificity = \frac{TN}{TN + FP} = \frac{102}{102 + 9} = 91.89\%$$

5) *F1 Score*

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{91.58 \times 88.28}{91.58 + 88.28} = 89.90\%$$

Persamaan - persamaan tersebut menunjukkan bahwa model kembar dengan algoritma CNN memiliki akurasi sebesar 90.1% dan memiliki nilai *F1 Score* sebesar 89.9% dalam mengenal dan membedakan sidik jari dengan perangkat DigitalPersona U.are.U 4500 menggunakan dataset yang telah disebutkan.

## **BAB V**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Berdasarkan dari penelitian untuk membuat implementasi *fingerprint recognition* untuk keamanan folder menggunakan algoritma *Convolutional Neural Network* yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil yang dapat membantu penelitian-penelitian baru selanjutnya. Adapun kesimpulan-kesimpulan tersebut adalah sebagai berikut:

1. Algoritma *Convolutional Neural Network* dapat digunakan untuk mengamankan folder dengan rekognisi sidik jari dengan akurasi yang tinggi.
2. Arsitektur model siamese, atau dengan nama lain model kembar, cocok untuk digunakan jika model diharapkan untuk dapat mencari persamaan dan perbedaan fitur dari data yang diberi.
3. Model kembar dengan algoritma *Convolutional Neural Network* mampu mencapai akurasi setinggi 90% dalam membedakan 2 citra sidik jari.
4. Algoritma *Convolutional Neural Network* memiliki kinerja yang lebih baik dalam bekerja dengan data citra, apabila dibandingkan dengan algoritma *Euclidean Distance*.
5. Kotornya lensa dari perangkat keras pemindai sidik jari dapat berefek pada hasil prediksi model dalam melakukan pencocokkan.

#### **5.2 Saran**

Adapun beberapa saran yang dapat digunakan dalam penelitian-penelitian baru selanjutnya dengan tema yang sama adalah sebagai berikut:

1. Diharapkan agar sistem dapat membedakan citra sidik jari dengan baik, walaupun terdapat kotoran pada lensa dari perangkat keras pemindai sidik jari.
2. Diharapkan agar sistem dapat bekerja dengan baik dalam membedakan sidik jari dari perangkat keras pemindai sidik jari yang berbeda-beda.
3. Mencoba algoritma yang berbeda dari *Convolutional Neural Network* untuk mengolah citra, seperti algoritma *Recurrent Neural Network* atau *Graph Neural Network*, dan membandingkan hasilnya dengan algoritma *Convolutional Neural Network*.
4. Penelitian ini tidak mencakup akan pemalsuan sidik jari dengan menggunakan bahan seperti latex atau gelatin. Untuk penelitian selanjutnya, diharapkan agar dapat membedakan sidik jari asli dan sidik jari palsu.



5. Akurasi yang didapatkan dalam penelitian ini adalah ~90%, dan dapat dikatakan sebagai hasil yang baik. Namun, menurut peneliti hasil ini sangat jauh dari kata sempurna dalam konteks keamanan biometrik. Untuk penelitian selanjutnya, diharapkan agar dapat mencapai hasil akurasi yang lebih tinggi lagi dengan menggunakan algoritma yang lain, atau dengan arsitektur model yang lebih baik lagi

## DAFTAR PUSTAKA

- Amruth, Y., Gopinatha, M. B., Gowtham, P. M., Kiran, M., & Harshalatha, Y. (2020). *Fingerprint and Signature Authentication System using CNN*.
- Arianda, J., Dirgantoro, B., & Setianingsih, C. (2019). *Detection Of Children With Personality Through Fingerprint Random Forest And Maximum Entropy Method*.
- Arun, T. K., Sowmya, V., Vinayakumar, R., Soman, P. K., & Sajith, V. V. (2019). *Convolutional Neural Networks for Fingerprint Liveness Detection System*.
- Bajahzar, A., & Guedri, H. (2019). Reconstruction of Fingerprint Shape using Fractal. *International Journal of Advanced Computer Science and Applications (IJACSA)*.
- Bakhshi, B., & Veisi, H. (2019). *End to End Fingerprint Verification Based On Convolutional Neural Network*.
- Banoula, M. (2023, February 16). *What is Tensorflow? Deep Learning Libraries and Program Elements Explained*. Dipetik December 20, 2023, dari Simplilearn: <https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-tensorflow>
- Brent, M. (2023, April 25). *File Permissions in Linux / Unix: How to Read, Write & Change?* Dipetik Juni 4, 2023, dari Guru99: <https://www.guru99.com/file-permissions.html>
- Chidi, O. (2020, July 24). *An Introduction to TensorFlow.js. What is TensorFlow.js | by Chidi O | The Startup | Medium*. Dipetik December 20, 2023, dari Medium: <https://medium.com/swlh/an-introduction-to-tensorflow-js-c2e837e70a01>
- de Alencar, M. S. (2022). *Cryptography and Network Security*. doi:10.1515/9781683926900
- Dermawan, D. R. (2016). *Implementasi Fingerprint Recognition Pada Keamanan Folder*.
- Edwards, B. (2021, Oct 18). *What Are Computer Files and Folders?* Dipetik Juni 4, 2023, dari How-To Geek: <https://www.howtogeek.com/757092/what-are-computer-files-and-folders/>
- Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., . . . Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition*. doi:10.1016/j.patcog.2017.10.013
- Holbrook, R., & Cook, A. (t.thn.). *A Single Neuron | Kaggle*. Dipetik 11 3, 2023, dari Kaggle: <https://www.kaggle.com/code/ryanhobbrook/a-single-neuron>
- Khan, A. I. (2018). *LAFIN: A Convolutional Neural Network-based Technique for Singular Point Extraction and Classification of Latent Fingerprints*.

- Kumar, A. (2023, May 12). *Different Types of CNN Architectures Explained: Examples*. Dipetik Juni 4, 2023, dari Data Analytics: <https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/>
- Lee, S. Y., Yap, W. S., Hum, C. Y., Goi, B. M., & Tee, Y. K. (2018). *Investigate the Impact of Colour to Grayscale Conversion on Sound Recovery via Visual Microphone*.
- Mattera, G., Nele, L., & Paoletta, D. (2023). Monitoring and control the Wire Arc Additive Manufacturing process using artificial intelligence techniques: a review. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-023-02085-5
- Narkhede, S. (2018, May 9). *Understanding Confusion Matrix / by Sarang Narkhede / Towards Data Science*. Dipetik December 20, 2023, dari Towards Data Science: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>
- Pal, A. K., Ray, S., & Bhattacharya, J. (2021, August). Comparative analysis of quantitative dermatoglyphic markers in schizophrenia patients and controls attending a superspeciality hospital in West Bengal. *International Journal of Research in Medical Sciences* 9. doi:10.18203/2320-6012.ijrms20213205
- Pandya, B., Alani, A. A., Bharadi, V., Cosma, G., Taherkhani, A., & McGinnity, T. (2018). *Fingerprint Classification using a Deep Convolutional Neural Network*.
- Shawkat, A., Hazim, R., & Mossiah, A. A. (2018). Biometrics Detection and Recognition Based-on Geometrical Features Extraction. *2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA)*. doi:10.1109/ICASEA.2018.8370956
- Shehu, Y. I., Garcia, A. R., & Palade, V. (2018). *Detailed Identification of Fingerprints using Convolutional Neural Networks*.
- Smid, M. E. (2021). Development of the advanced encryption standard. *Journal of Research of the National Institute of Standards and Technology*. doi:10.6028/JRES.126.024
- srinandutta. (2023, Jan 16). *Thresholding-Based Image Segmentation*. Dipetik Juni 4, 2023, dari GeeksforGeeks: <https://www.geeksforgeeks.org/thresholding-based-image-segmentation/>
- What is an AI model? / IBM*. (t.thn.). Dipetik December 20, 2023, dari IBM: <https://www.ibm.com/topics/ai-model>