

**KLASIFIKASI PENYAKIT GINJAL MENGGUNAKAN
CONVOLUTIONAL NEURAL NETWORK**

SKRIPSI

JEREMIA FELIX RAJAGUKGUK

201401140



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA**

2025

UNIVERSITAS SUMATERA UTARA

**KLASIFIKASI PENYAKIT GINJAL MENGGUNAKAN CONVOLUTIONAL
NEURAL NETWORK**

SKRIPSI

**Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh
ijazah Sarjana Ilmu Komputer**

JEREMIA FELIX RAJAGUKGUK

201401140



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA**

2025

UNIVERSITAS SUMATERA UTARA

PERSETUJUAN

Judul : KLASIFIKASI PENYAKIT GINJAL
MENGGUNAKAN CONVOLUTIONAL
NEURAL NETWORK

Kategori : SKRIPSI

Nama : JEREMIA FELIX RAJAGUKGUK

Nomor Induk Mahasiswa : 201401140

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI
INFORMASI UNIVERSITAS SUMATERA
UTARA

Telah diuji dan dinyatakan lulus di Medan, 10 Januari 2025

Komisi Pembimbing :

Dosen Pembimbing II



Fauzan Nur Ahmadi, S.Kom, M.Cs

NIP. 198512292018051001

Dosen Pembimbing I



Amer Sharif S.Si., M.Kom.

NIP. 196910212021011001

Diketahui/Disetujui Oleh

Ketua Program Studi S-1 Ilmu Komputer



Dr. Amalia, S.T., M.T

NIP. 197812212014042001

PERNYATAAN**KLASIFIKASI PENYAKIT GINJAL MENGGUNAKAN
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, 10 Januari 2025



Jeremia Felix Rajagukguk

201401140

PENGHARGAAN

Segala puji syukur dipanjatkan kepada Tuhan Yang Maha Esa atas kasih dan karunia-Nya sehingga penulis mampi melakukan penyusunan skripsi ini sebagai syarat untuk mendapatkan gelar Sarjana Komputer di Program Studi S-1 Ilmu Komputer, Universitas Sumatera Utara.

Penyusunan skripsi ini tidak terlepas dari bantuan, dukungan, dan bimbingan dari banyak pihak. Oleh karena itu, penulis mengucapkan banyak terima kasih kepada:

1. Bapak Prof. Dr. Muryanto Amin S.Sos., M.Si. selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Bapak Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M. selaku Wakil Dekan I Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Ibu Dr. Amalia, S.T., M.T. selaku Ketua Program Studi S-1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
5. Bapak Amer Sharif S.Si., M.Kom. selaku dosen Pembimbing I yang telah memberikan banyak masukan, bimbingan dan dukungan yang berharga kepada penulis selama proses penyusunan skripsi ini.
6. Bapak Fauzan Nur Ahmadi, S.Kom, M.Cs selaku dosen Pembimbing II yang telah memberikan masukan dan bimbingan yang berharga kepada penulis selama proses penyusunan skripsi ini.
7. Seluruh Bapak dan Ibu Dosen Program Studi S-1 Ilmu Komputer, yang telah membimbing saya pada masa perkuliahan.
8. Bapak Nasrum Rajagukguk dan Ibu Erda Silalahi, selaku orang tua penulis, Erika Sianipar selaku nenek penulis, Gabriel Rajagukguk selaku adik penulis, dan anggota keluarga lainnya yang telah memberikan dukungan mental dan juga materi sampai menyelesaikan skripsi.

9. Orang terdekat, sahabat, serta teman peneliti yaitu atas nama Reynold, Chalil, Joshua, Iqbal, Ewaldo, Kevin, Gery, Joe dan teman-teman lainnya yang belum dapat disebutkan oleh peneliti.
10. Dan seluruh pihak yang telah memberi dukungan serta doa baik yang tidak dapat penulis sebutkan satu per-satu. Semoga Tuhan melimpahkan berkah dan kebaikan-Nya atas dukungan yang telah diberikan kepada penulis dan hasil penelitian ini dapat memberi manfaat untuk kedepannya.

Medan, 10 Januari 2025

Penulis



Jeremia Felix Rajagukguk



ABSTRAK

KLASIFIKASI PENYAKIT GINJAL MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK

Ginjal adalah organ tubuh manusia di bagian bawah tulang rusuk belakang. Ginjal merupakan organ tubuh yang sangat penting dalam sistem metabolisme tubuh. Penyakit ginjal terkait dengan terjadinya penyakit kardiovaskular yang memiliki angka kematian dan biaya perawatan tinggi. Gagal ginjal juga dapat terjadi jika kelainan ginjal tidak ditemukan dan tidak diobati segera, karenanya diagnosis dini merupakan langkah yang penting. Oleh karena itu dibutuhkan metode untuk menghadapi permasalahan ini dengan lebih cepat dan akurat, penyakit ginjal dapat diklasifikasikan melalui citra *CT scan*. Salah satu solusi nya adalah *Convolutional Neural Network* (CNN) yaitu metode yang menggunakan input gambar yang dapat menentukan objek yang terdapat dalam suatu citra agar mesin dapat mengenali dan membedakan antara satu citra dengan yang lainnya. Berbagai arsitektur CNN telah dikembangkan. Faktor-faktor yang membedakan arsitektur-arsitektur tersebut seperti jumlah lapisan jaringan dan cara lapisan konvolusi diletakkan. Salah satunya adalah model CNN yang ringan atau *lightweight* CNN dengan kinerjanya yang cepat dan ukurannya yang kecil namun tidak mengorbankan kinerja akurasi yang sangat dibutuhkan. *Lightweight* CNN digunakan untuk mengklasifikasi kondisi yang dimiliki ginjal melalui citra *CT scan* yang diberikan. Hasil akurasi yang didapatkan adalah 98% untuk data latih, 99% untuk data validasi dan 99% untuk data uji, implementasi yang digunakan berupa sebuah aplikasi berbasis *website*.

Kata Kunci: Penyakit Ginjal, *Deep Learning*, *Convolutional Neural Network*, *lightweight* CNN

ABSTRACT

KIDNEY DISEASE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK

The kidney is an organ located at the bottom of the back rib cage. Kidney is a very important organ in the body's metabolic system. Kidney disease is associated with a risk of cardiovascular disease, which has a high treatment cost and mortality rate. Kidney failure can also occur if kidney abnormalities are not discovered and treated promptly, hence early diagnosis is an important step. Therefore, a method is needed to deal with this problem more quickly and accurately, kidney disease can be classified through CT scan images. One solution is Convolutional Neural Network (CNN), a method that uses image input that can determine the objects contained in an image so that the machine can recognize and distinguish between one image and another. Various CNN architectures have been developed. Factors that distinguish these architectures such as the number of network layers and the way the convolution layer is placed. One of them is the lightweight CNN model with its fast performance and small size but not sacrificing the much-needed accuracy performance. Lightweight CNN was used to classify the condition of the kidney through the given CT scan images. The accuracy results obtained were 98% for training data, 99% for validation data and 99% for testing data, implemented as a web-based application.

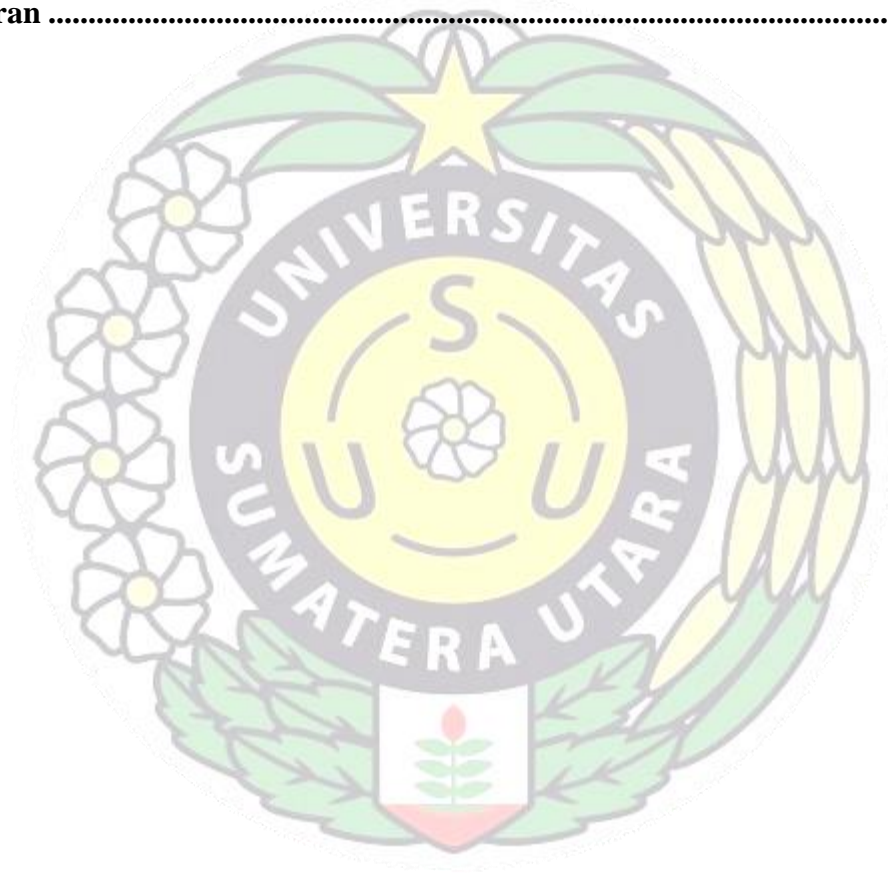
Keywords: Kidney Disease, Deep Learning, Convolutional Neural Network, lightweight CNN

Daftar Isi

PERSETUJUAN.....	i
PERNYATAAN.....	ii
PENGHARGAAN.....	iii
ABSTRAK	v
ABSTRACT	vi
Daftar Isi	vii
Daftar Tabel.....	x
Daftar Gambar.....	xi
Daftar Lampiran	xii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah	3
1.4 Tujuan Penelitian	4
1.5 Manfaat Penelitian	4
1.6 Metodologi Penelitian.....	4
1.7 Sistematika Penulisan	5
BAB II LANDASAN TEORI	6
2.1 Machine Learning	6
2.2 Deep Learning.....	6
2.3 Median filter.....	7
2.4 Convolutional Neural Network (CNN).....	8
2.5 Arsitektur CNN	8

2.6	Lightweight CNN.....	10
2.7	Ginjal.....	10
2.8	<i>CT Scan</i>	11
2.9	Penelitian Sebelumnya.....	12
BAB III ANALISIS DAN PERANCANGAN		13
3.1	Analisis Sistem.....	13
3.1.1	Analisis Masalah.....	13
3.1.2	Analisis Kebutuhan.....	13
3.2	Perancangan Sistem	14
3.3	Perhitungan	18
3.4	Pemodelan Sistem.....	22
3.4.1	Diagram Umum Sistem.....	22
3.4.2	Use Case Diagram.....	23
3.4.3	Activity Diagram.....	24
3.5	Flowchart	25
3.5.1	Flowchart Pelatihan lightweight CNN.....	25
3.5.2	Flowchart Sistem.....	27
3.6	Perancangan <i>Interface</i>	28
3.6.1	Halaman Prediksi dengan Gambar.....	28
BAB IV IMPLEMENTASI DAN PENGUJIAN		29
4.1	Implementasi.....	29
4.1.1	Halaman Utama.....	29
4.1.2	Pelatihan Model	29
4.1.3	Halaman Input dan Prediksi.....	33
4.2	Sumber Data.....	34
4.3	Pengujian sistem	34

4.4	Hasil Pengujian Sistem	43
4.5	User Acceptance Testing	47
BAB V KESIMPULAN DAN SARAN		48
5.1	Kesimpulan	48
5.2	Saran	48
Daftar Pustaka.....		49
Lampiran		53



Daftar Tabel

Tabel 3. 1 Model lightweight CNN.....	15
Tabel 3. 2 Fungsi untuk Pelatihan Model.....	17
Tabel 4. 1 Data Uji	40
Tabel 4. 2 Hasil Pengujian Sistem.....	43
Tabel 4. 3 User Acceptance Testing.....	47



Daftar Gambar

Gambar 2. 1 Citra Sebelum dan Sesudah Median Filter	7
Gambar 2. 2 Arsitektur CNN.....	8
Gambar 2. 3 Convolutional Layer	8
Gambar 2. 4 Pooling Layer	9
Gambar 2. 5 Arsitektur Lightweight CNN	10
Gambar 2. 6 Citra CT Scan	11
Gambar 3. 1 Diagram Umum Sistem	23
Gambar 3. 2 Diagram Use Case	24
Gambar 3. 3 Diagram Activity	25
Gambar 3. 4 Pelatihan Lighweight CNN.....	26
Gambar 3. 5 Sistem	27
Gambar 3. 6 Halaman Prediksi dengan Gambar	28
Gambar 4. 1 Halaman Utama	29
Gambar 4. 2 Import Library	30
Gambar 4. 3 Pembagian Dataset.....	30
Gambar 4. 4 Preprocessing.....	31
Gambar 4. 5 Pembangunan Model	32
Gambar 4. 6 Kompilasi dan Pelatihan Model	32
Gambar 4. 7 Proses Pelatihan	33
Gambar 4. 8 Halaman Input dan Prediksi.....	34
Gambar 4. 9 Citra CT Scan	34
Gambar 4. 10 Grafik Traning dan Validation Accuracy	36
Gambar 4. 11 Grafik Traning dan Validation Loss	36
Gambar 4. 12 Confusion Matrix.....	38

Daftar Lampiran

Lampiran 1 Kode Program.....	53
Lampiran 2 User Acceptance Testing.....	63



BAB I

PENDAHULUAN

1.1 Latar Belakang

Ginjal adalah salah satu organ tubuh manusia yang letaknya di bagian bawah tulang rusuk belakang. Ginjal adalah organ penting dalam sistem metabolisme (Alamsyah, 2019). Fungsi utama ginjal adalah membersihkan darah dari senyawa yang beracun sebelum didistribusikan ke seluruh bagian tubuh.

Penyakit ginjal memiliki keterkaitan yang erat dengan kondisi kardiovaskular, dan keduanya berisiko tinggi menyebabkan angka kematian serta memerlukan biaya perawatan yang signifikan (Putra, 2019). Di antara berbagai jenis penyakit ginjal, kista, batu ginjal, dan tumor ginjal adalah yang paling umum dan dapat mengganggu fungsi ginjal. Salah satu alat yang digunakan untuk mendiagnosis penyakit ginjal adalah *CT scan* (Islam, 2022). *CT scan* memanfaatkan sinar *X-ray* untuk memindai bagian-bagian anatomi tubuh yang diperlukan, sehingga dapat menghasilkan gambar yang sangat informatif. Metode ini sangat ideal untuk pemeriksaan ginjal karena memberikan informasi yang detail dalam bentuk potongan-potongan gambar.

Gagal ginjal dapat terjadi jika kelainan ginjal tidak ditemukan dan tidak diobati segera, karenanya diagnosis dini merupakan langkah yang penting (Gunasekara, 2022). Namun, interpretasi manual dari gambar *CT scan* oleh ahli dapat memakan waktu dan juga tidak mungkin lepas dari tingkat kesalahan yang disebabkan oleh mata manusia. Karena itu dibutuhkan suatu cara yang lebih baik dalam menghadapi permasalahan ini.

Deep learning merupakan satu dari beberapa kelas algoritma *machine learning* yaitu *neural network*, yaitu model matematika yang terinspirasi oleh struktur otak. *Deep learning* membuat algoritma *neural network* untuk dapat bekerja dengan sangat baik dalam membangun model prediksi di sekitar

masalah yang kompleks seperti *computer vision* (Bisong, 2019). Salah satu metode dari *deep learning* adalah *Convolutional Neural Network* (CNN).

CNN merupakan metode yang digunakan untuk mengenali atau mengklasifikasi penyakit maupun objek (Marcella, 2022). Metode ini memanfaatkan gambar sebagai input, memungkinkan sistem untuk mengidentifikasi dan membedakan berbagai objek dalam suatu gambar. Ini umumnya diterapkan agar mesin dapat mengenali perbedaan antara satu gambar dengan yang lainnya. Arsitektur *Convolutional Neural Network* (CNN) memiliki beberapa lapisan, yaitu *convolutional*, *pooling*, dan *fully connected layer*. Agar dapat berfungsi dengan baik dan mencegah overfitting, CNN memerlukan dataset yang besar.

Berbagai arsitektur CNN telah dikembangkan. Faktor-faktor yang membedakan arsitektur-arsitektur tersebut seperti jumlah lapisan jaringan dan cara lapisan konvolusi diletakkan. Namun, karena keterbatasan memori, arsitektur CNN biasa mahal secara komputasi dengan banyaknya parameter yang dapat dipelajari dan operasi aritmatika (Shuvo, 2020). Karena alasan ini, model CNN yang ringan atau *lightweight CNN* semakin populer di kalangan para peneliti karena kinerjanya yang lebih cepat dan ukurannya yang kecil tanpa mengorbankan kinerja akurasi yang sangat dibutuhkan dibandingkan dengan model atau arsitektur yang terkenal.

Pada penelitian yang dilakukan oleh (Shuvo dkk., 2020) menggunakan *lightweight CNN* untuk mengklasifikasi penyakit pernapasan dengan skalogram hibrida mendapatkan akurasi sebesar 98,92% untuk klasifikasi kronis tiga kelas dan 98,70% untuk klasifikasi patologis enam kelas. Pada penelitian yang dilakukan oleh (Suta dkk., 2019) menggunakan CNN arsitektur Efficientnet-B3 untuk mengklasifikasi tumor otak dengan *CT scan* hasilnya akurasi 99.7%. Pada penelitian yang dilakukan oleh (Widodo, 2022) dalam mengklasifikasi kanker paru-paru dengan citra CT scan mendapatkan akurasi sebesar 91.4%. Pada penelitian yang dilakukan oleh (Fadlurrahman

dkk., 2023) menggunakan CNN dengan untuk mengklasifikasi COVID-19 dengan citra *X-Ray* mendapat akurasi sebesar 90%.

Berdasarkan penelitian-penelitian sebelumnya, dapat disimpulkan bahwa Convolutional Neural Network (CNN) terbukti efektif dalam mengklasifikasikan citra penyakit. Oleh karena itu, penulis berencana untuk melakukan penelitian mengenai penerapan CNN dalam mengklasifikasikan penyakit ginjal. Penyakit yang akan diklasifikasikan meliputi kista, tumor, dan batu ginjal dengan menggunakan model CNN lightweight.

1.2 Rumusan Masalah

Convolutional Neural Network (CNN) merupakan salah satu metode deep learning yang unggul dalam pengolahan citra karena kemampuannya untuk secara otomatis mengenali gambar tanpa memerlukan intervensi manual. Keunggulan ini menjadikan CNN sangat cocok digunakan untuk klasifikasi penyakit ginjal melalui citra CT scan, menggantikan metode interpretasi manual yang cenderung memakan waktu dan berisiko terhadap kesalahan. Oleh karena itu, diperlukan penerapan CNN untuk membangun sistem klasifikasi yang lebih cepat dan akurat dalam membantu diagnosis penyakit ginjal.

1.3 Batasan Masalah

Beberapa batasan masalah dalam penelitian ini adalah:

1. Metode yang digunakan adalah CNN dengan model yang digunakan yaitu *lightweight CNN*
2. Data yang digunakan adalah citra *CT scan* dengan format .jpg yang dikumpulkan dari situs dataset Kaggle.
3. Hasil klasifikasi ada 4 yaitu normal, kista, tumor dan batu ginjal.
4. Jumlah keseluruhan data gambar yaitu berjumlah 5360 gambar yang akan dibagi menjadi data latih, data validasi dan data uji.
5. Hasil akhir pengembangan akan dibuat dalam bentuk web.

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah membuat sistem klasifikasi penyakit ginjal menggunakan *lightweight Convolutional Neural Network* untuk mengidentifikasi dini.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah:

1. Memberi bantuan untuk ahli dalam diagnosis penyakit ginjal menggunakan citra *CT scan*.
2. Mengetahui kinerja CNN dalam mengklasifikasikan penyakit ginjal dengan *CT scan*.
3. Dapat menjadi referensi untuk penelitian yang berhubungan dengan metode CNN.

1.6 Metodologi Penelitian

Metode penelitian yang digunakan pada penelitian ini adalah:

1. Studi Pustaka

Pada tahap ini digunakan metode studi untuk meninjau dan mengumpulkan berbagai referensi yang memiliki hubungan dengan penelitian.

2. Pengumpulan Data

Pada tahap ini dikumpulkan data yang diperlukan untuk penelitian, yaitu gambar *CT scan*.

3. Analisis dan Perancangan Sistem

Tahap ini meliputi analisis dan kebutuhan penelitian, juga merancang alur kerja sistem secara umum dalam bentuk diagram.

4. Implementasi Sistem

Implementasi dari model yang akan dilakukan dibangun sesuai dengan perancangan.

5. Pengujian Sistem

Sistem akan di uji untuk memastikan kinerjanya.

6. Dokumentasi

Penelitian akan dilaporkan dalam bentuk skripsi.

1.7 Sistematika Penulisan

Sistematika penulisan yang dilakukan adalah sebagai berikut:

BAB 1 PENDAHULUAN

Membahas tentang latar belakang penelitian judul skripsi, rumusan dan batasan masalah, tujuan dari penelitian yang dilakukan, manfaat dan metode dari penelitian, serta sistematika dari penulisan skripsi.

BAB 2 LANDASAN TEORI

Membahas penjelasan *Convolutional Neural Network* dan *lightweight CNN*.

BAB 3 ANALISIS DAN PERANCANGAN

Membahas analisis dari sistem. Selanjutnya merancang sistem dengan menggunakan Algoritma CNN dan *lightweight CNN*.

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Membahas tentang implementasi juga pengujian sistem berdasarkan tahapan analisis dan perancangan yang telah dilakukan.

BAB 5 KESIMPULAN DAN SARAN

Memuat kesimpulan beserta saran-saran diharapkan nantinya dikemudian hari dapat bermanfaat terhadap penelitian lain.

BAB II

LANDASAN TEORI

2.1 Machine Learning

Machine learning adalah salah satu cabang dari *kecerdasan buatan* yang berfokus pada pemrograman komputer agar dapat berperilaku cerdas layaknya manusia. Melalui proses ini, komputer dapat meningkatkan pemahaman mereka secara otomatis melalui pengalaman (Kusuma, 2020). Tujuan utama *machine learning* adalah untuk mengembangkan sistem berkemampuan belajar mandiri dan mengambil keputusan tanpa perlu diprogram ulang oleh manusia. Dengan kemampuan itu, mesin bisa beradaptasi dan membuat keputusan yang sesuai dengan terjadinya perubahan.

Machine learning berfungsi ketika data tersedia sebagai input, yang kemudian dianalisis untuk mengidentifikasi pola-pola tertentu dalam kumpulan data besar. Data ini menjadi bahan baku penting untuk pembelajaran, memungkinkan mesin untuk menghasilkan analisis yang akurat. Dalam proses *machine learning*, terdapat dua jenis data yang memainkan peran penting. Data training digunakan untuk pelatihan algoritma, sedangkan data testing berfungsi untuk mengukur performanya setelah dilatih, terutama saat dihadapkan pada data yang belum dilihat sebelumnya. (Retnoningsih, 2020).

2.2 Deep Learning

Cabang dari *machine learning* yaitu *Deep Learning* yang algoritmanya terinspirasi oleh struktur otak manusia, yang dikenal sebagai Jaringan Saraf Buatan. Jaringan ini terdiri dari sejumlah lapisan simpul yang saling terhubung berfungsi untuk memproses data. Semakin banyak lapisan jaringannya maka semakin dalam yang memungkinkan *deep learning* melakukan tugas dan mempelajari fitur yang lebih sulit (Syahputra, 2024). Jaringan saraf berusaha memodelkan pembelajaran manusia dengan mengolah dan menganalisis *big data*. Jaringan saraf akan meningkatkan akurasi setiap kali mereka melakukan

tugas dengan data itu. Hal ini mirip dengan cara manusia meningkatkan kemampuannya.

2.3 Median filter

Median filter adalah suatu metode yang digunakan untuk mengurangi atau menghapus noise seperti *noise salt & pepper* (Kosasih, 2021). Median filter bekerja dengan cara membuat jendela (*window*) yang memuat sejumlah *pixel* yang berjumlah ganjil. Nilai median kemudian didapat dengan cara mengurutkan semua nilai *pixel* dari jendela itu, lalu mengganti setiap nilai *pixel* di citra dengan nilai median dari *pixel* yang berdekatan. Formula median filter adalah

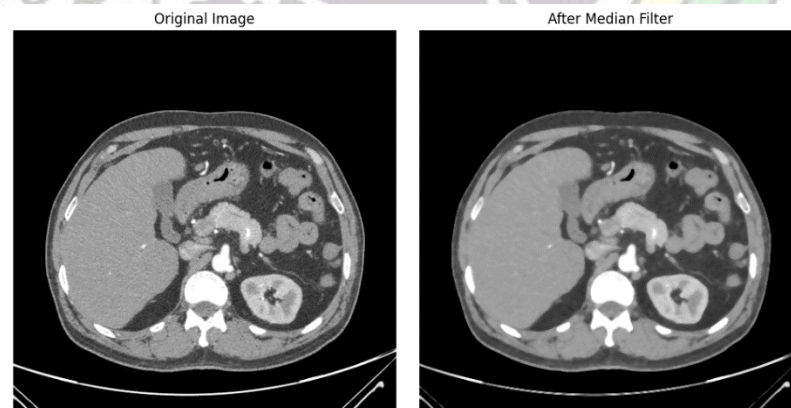
$$f(x,y) = \text{median } \Sigma_{(s,t) \in S_{x,y}} g(s,t)$$

Dimana:

- $f(x,y)$ adalah citra hasil dari median filter.
- $g(s,t)$ adalah jendela yang dicari nilai mediannya.

Jendela digeser tiap *pixel* pada semua daerah citra. Untuk setiap pergeseran dibuat jendela baru dengan mengganti titik tengah pada jendela dengan nilai median pada jendela tersebut.

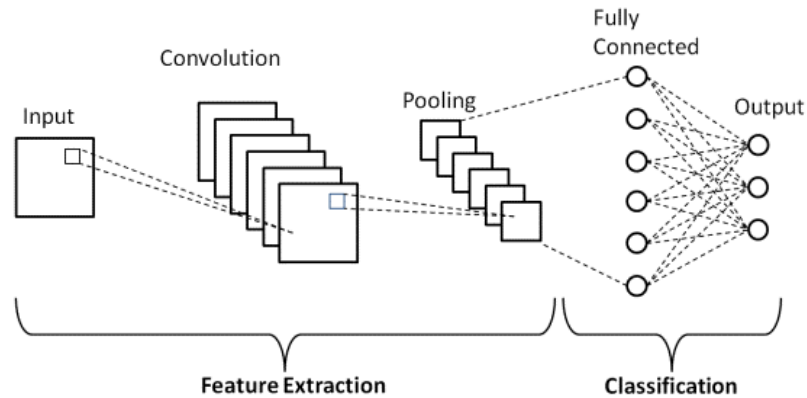
Hasil median filter dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Citra Sebelum dan Sesudah Median Filter

2.4 Convolutional Neural Network (CNN)

Metode CNN sering dipakai untuk mengenali atau mengklasifikasi suatu objek (Marcella, 2022). Metode CNN menggunakan input gambar yang dapat menentukan objek yang terdapat dalam suatu citra agar mesin dapat mengenali dan membedakan antara satu citra dengan yang lainnya. Arsitektur CNN dapat dilihat pada Gambar 2.2.



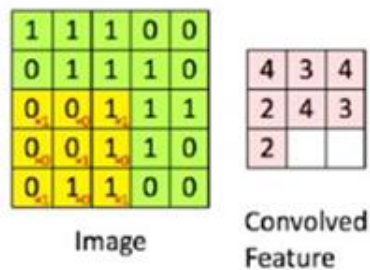
Gambar 2. 2 Arsitektur CNN

(Sumber: Ray, 2020)

2.5 Arsitektur CNN

1. Convolution Layer

Convolutional Layer yaitu lapisan dimana dilakukan operasi untuk ekstraksi fitur. Operasi ekstraksi fitur merupakan hasil akhir titik antara bobot dalam filter dengan *pixel* citra (Kurniawan, 2023). *Convolutional Layer* dapat dilihat pada Gambar 2.3.

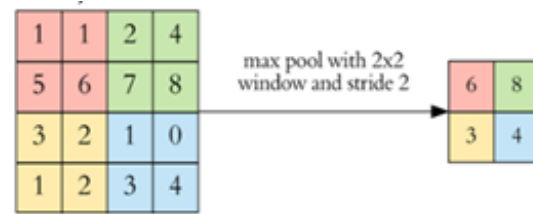


Gambar 2. 3 Convolutional Layer

(Sumber: Riyadi, 2021)

2. *Pooling Layer*

Pooling Layer adalah langkah yang bertujuan untuk mengurangi ukuran citra, sekaligus meningkatkan ketahanan posisi fitur. Proses ini dilakukan melalui metode *max pooling*, dimana hasil dari *layer* konvolusi dipisah menjadi beberapa bagian kecil. Dari setiap bagian ini, diambil nilai tertinggi diambil untuk membentuk matriks citra yang telah direduksi (Riyadi, 2021). Pooling Layer dapat dilihat pada gambar 2. 4.

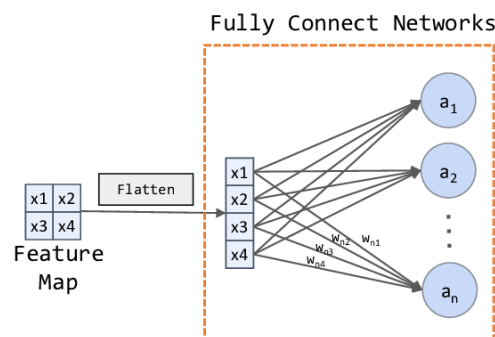


Gambar 2. 4 Pooling Layer

(Sumber: Tama, 2023)

3. *Fully connected layer*

Setelah mengalami proses ekstraksi fitur dengan bantuan convolutional layer dan pooling layer, nilai-nilai ekstraksi tersebut kemudian dirubah dimensinya menggunakan fungsi flattening agar bisa di proses ke fully connected layer. Dalam fully connected layer dilakukan proses prediksi yang menempatkan suatu citra masukan ke dalam kategori kelas tertentu (Nashrullah, 2020).

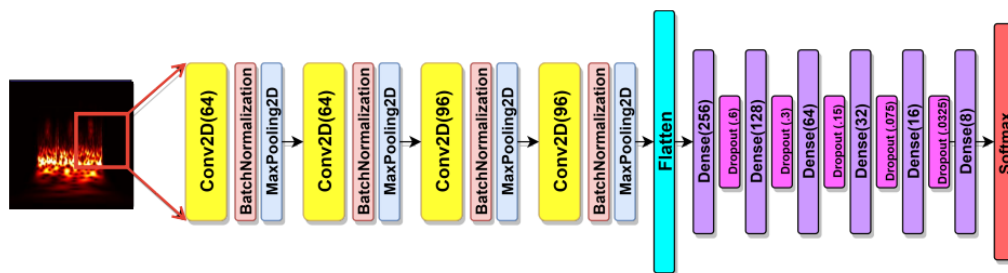


Gambar 2. 5 3. Fully connected layer

(Wang, 2024)

2.6 Lightweight CNN

Akibat keterbatasan memori, arsitektur CNN biasa mahal secara komputasi dengan banyaknya parameter yang dapat dipelajari dan operasi aritmatika (Shuvo, 2020). Karena alasan ini, model CNN yang ringan atau *lightweight* CNN semakin populer di kalangan para peneliti karena kinerjanya yang lebih cepat dan ukurannya yang kecil tanpa mengorbankan kinerja akurasi yang sangat dibutuhkan. Pada peneli Pada penelitian ini akan digunakan *lightweight CNN* yang dibuat oleh Shuvo dkk., pada penelitiannya dalam mengklasifikasi penyakit pernapasan pada tahun 2020 yang mendapatkan akurasi sebesar 98,92% dalam klasifikasi tiga kelas penyakit kronis, serta 98,70% dalam klasifikasi enam kelas penyakit patologis. Penggunaan *lightweight CNN* juga telah digunakan dalam mengklasifikasi penyakit pada tanaman yang dilakukan Thakur dkk., pada tahun 2023 mendapat akurasi 99.16% dan pada penelitian yang dilakukan Sunija dkk. Pada tahun 2021 untuk mengklasifikasi penyakit mata mendapat hasil akurasi 99.69%. Arsitektur *Lightweight CNN* dapat dilihat pada Gambar 2.5.



Gambar 2. 6 Arsitektur *Lightweight CNN*

(Sumber: Shuvo, 2020)

2.7 Ginjal

Ginjal adalah organ yang letaknya di bagian bawah tulang rusuk tepatnya di bagian belakang. Ginjal merupakan organ yang memiliki tugas penting dalam sistem metabolisme (Alamsyah, 2019). Fungsi utamanya yaitu untuk

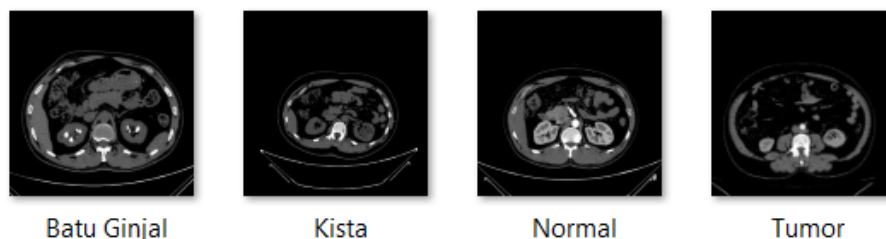
dalam penyaringan darah dan pengeluaran zat sisa dan berbahaya melalui urin (Amani, 2022).

Jika organ ini mengalami gangguan maka akan berdampak pada kesehatan (Sumber Website Halodoc). Tanda dan gejala awal yang terkait dengan penyakit ginjal adalah sebagai berikut.

1. Suhu tubuh menjadi lebih panas.
2. Rasa nyeri di area pinggul.
3. Nyeri saat buang air kecil.
4. Warna urine menjadi lebih pekat.
5. Frekuensi buang air kecil meningkat.
6. Berkurangnya nafsu makan.
7. Rasa mual dan muntah.
8. Kelelahan dan sulit untuk berkonsentrasi.
9. Kulit kering dan gatal.
10. Urine berbusa dan terdapat darah.

2.8 CT Scan

CT Scan adalah alat yang dapat digunakan dalam mengidentifikasi penyakit yang terdapat di dalam tubuh manusia (Siregar, 2019). Alat ini memungkinkan dokter untuk mendeteksi adanya kelainan tanpa harus melakukan tindakan bedah. Dengan kemampuan untuk menghasilkan gambaran yang jelas dari berbagai bagian tubuh, CT scan sering digunakan untuk memeriksa area-area seperti kepala, rongga dada, dan rongga perut. Hasil CT scan dapat dilihat pada Gambar 2.6.



Gambar 2. 7 Citra CT Scan

2.9 Penelitian Sebelumnya

1. Menurut penelitian yang telah dilakukan oleh Shuvo dan timnya, yang berjudul “Model CNN Ringan untuk Mendeteksi Penyakit Pernapasan dari Suara Auskultasi Paru Menggunakan Skalogram Hibrid Berbasis EMD-CWT,” mereka berhasil meraih akurasi sebesar 98,92% dalam klasifikasi tiga kelas penyakit kronis, serta 98,70% dalam klasifikasi enam kelas penyakit patologis.
2. Sebuah penelitian yang dilakukan oleh Suta dan rekan-rekannya yaitu mendiagnosa tumor otak berdasarkan citra MRI menunjukkan hasil yang sangat tinggi dengan tingkat akurasi mencapai 99,7%.
3. Penelitian yang dilakukan oleh Widodo yaitu mengklasifikasi kanker dan arteri pada gambar *CT scan* menggunakan CNN berhasil mencapai tingkat akurasi sebesar 91.4%.
4. Penelitian yang dipimpin oleh Fadlurrahman dan rekan-rekannya berjudul “Klasifikasi Covid-19 Menggunakan Algoritma CNN” memperoleh akurasi sebesar 90% melalui analisis citra X-Ray.
5. Penelitian yang dilakukan oleh Almuayqil dan rekan-rekannya untuk mengklasifikasi penyakit ginjal melalui CT scan memperoleh akurasi pada data uji sebesar 98,10% dengan model Xception dan 98,77% dengan model VGG19.

BAB III

ANALISIS DAN PERANCANGAN

3.1 Analisis Sistem

Pada tahap ini dilakukan proses evaluasi serta mengidentifikasi semua hal yang diperlukan untuk sistem. Tujuannya untuk memfasilitasi perbaikan sistem nantinya sehingga sistem yang dirancang mampu beroperasi sebagaimana mestinya.

3.1.1 Analisis Masalah

Analisis ini merupakan langkah memahami juga merumuskan masalah yang timbul dalam pengembangan sistem, dengan fokus pada penciptaan sistem yang optimal dalam kinerjanya. Dalam penelitian ini, fokus analisis masalah adalah bagaimana mengklasifikasi penyakit ginjal. Pendekatan yang diusulkan adalah pengembangan model klasifikasi menggunakan *lightweight* CNN, model ini diharapkan mampu mengklasifikasi penyakit ginjal dari citra *CT Scan* dengan akurat sehingga memberikan informasi yang tepat.

3.1.2 Analisis Kebutuhan

Tahap ini untuk mengenali data dan cara yang diperlukan untuk merancang sistem. Dalam proses perancangan, analisis kebutuhan meliputi persyaratan fungsional dan non fungsional yang harus diwujudkan agar sistem yang dikembangkan dapat mencapai tujuannya.

1. Fungsional

Kebutuhan fungsional merujuk pada keperluan yang mencakup berbagai proses yang dijalankan, diantaranya:

- Sistem mampu membaca input dari pengguna.
- Sistem dapat mengenali gambar CT Scan saat proses prediksi oleh model.
- Sistem dapat memberikan hasil berupa prediksi kondisi ginjal
- Sistem dapat memberikan nilai keakuratan atas prediksi yang dikeluarkan.

2. Non Fungsional

Kebutuhan non fungsional berkaitan dengan fitur maupun batasan sistem, diantaranya:

- Ekstensi gambar yang dapat diterima sisten adalah .JPG, .PNG, dan .JPEG.
- Batasan kondisi ginjal yang dapat diprediksi sistem adalah kista, batu ginjal, tumor dan normal.
- Memiliki koneksi internet agar dapat terhubung ke sistem.
- Kondisi ginjal dapat dideteksi kurang dari 10 detik.

3.2 Perancangan Sistem

Model akan dibangun dengan proses sebagai berikut:

1. Persiapan data

Persiapan data pada penelitian ini yaitu:

- *Resize* atau pengubahan ukuran gambar pada dataset menjadi 224x224 *pixel* untuk mempercepat proses klasifikasi.
- *Grayscale* untuk mempercepat proses pelatihan.
- Penerapan median filter pada gambar untuk mengurangi noise.
- Normalisasi, yaitu pengubahan nilai pixel menjadi 0-1 untuk mempercepat konvergensi selama pelatihan model.

2. Pemodelan

Tahap selanjutnya adalah membangun model berdasarkan data yang telah diproses. Model CNN yang dibangun dimulai dengan lapisan input yang dirancang untuk gambar *grayscale* berukuran 224 x 224 *pixel*. Lapisan konvolusi (Conv) pertama menggunakan 64 filter dengan kernel 5x5, diikuti oleh lapisan *max-pooling* 2x2. Tiga blok konvolusi tambahan disusun bertingkat, masing-masing dengan kernel 3x3 dengan 64, 96, serta 96 filter secara berurutan. Blok Conv ini dilengkapi dengan *Batch normalization* dan lapisan *max-pooling* 2x2. Batch Normalization diterapkan untuk mempercepat kemampuan selama pelatihan.

Output dari semua lapisan tadi diratakan (*flattened*) dan dihubungkan dengan lima pasang lapisan *fully connected* dan *dropout*. Fungsi aktivasi *Rectified Linear*

Unit (ReLU) diterapkan pada semua blok Conv dan lapisan *fully connected*. Lapisan *dropout*, yang ditempatkan setelah lapisan *fully connected*, berfungsi sebagai teknik regularisasi untuk mencegah *overfitting* dengan cara secara acak menonaktifkan sebagian neuron selama pelatihan. Model diakhiri dengan lapisan output *SoftMax* yang menghitung probabilitas untuk setiap kelas. Model dapat dilihat pada Tabel 3.1.

Tabel 3. 1 Model lightweight CNN

Blok	Lapisan/Langkah	Penjelasan
Input Layer	img_input = layers.Input(shape=(224, 224, 1))	Membentuk input dengan ukuran 224x224 piksel dan 1 saluran (grayscale).
Layer 1	Conv2D(64, 5, activation="relu")	Konvolusi dengan 64 filter berukuran 5x5.
	BatchNormalization()	Menormalkan output dari layer konvolusi untuk mempercepat proses pelatihan dan meningkatkan stabilitas model.
	MaxPooling2D(2)	Mengurangi ukuran fitur map.
Layer 2	Conv2D(64, 3, activation="relu")	Konvolusi dengan 64 filter berukuran 3x3.
	BatchNormalization()	Menormalkan output untuk menjaga stabilitas pelatihan.
	MaxPooling2D(2)	Mengurangi ukuran fitur map.
Layer 3	Conv2D(96, 3, activation="relu")	Konvolusi dengan 96 filter berukuran 3x3.
	BatchNormalization()	Menormalkan output untuk menjaga stabilitas pelatihan.
	MaxPooling2D(2)	Mengurangi ukuran fitur map.

Layer 4	Conv2D(96, 3, activation="relu")	Konvolusi dengan 96 filter berukuran 3x3.
	BatchNormalization()	Menormalkan output untuk menjaga stabilitas pelatihan.
	MaxPooling2D(2)	Mengurangi ukuran fitur map menjadi 14x14 untuk efisiensi proses.
Fully Connected (FC)	Flatten()	Mengubah fitur map 3D menjadi 1D agar dapat diproses oleh lapisan Dense.
	Dense(256, activation='relu')	Lapisan fully connected dengan 256 unit, mempelajari hubungan non-linear dalam data.
	Dropout(0.6)	Mengurangi overfitting dengan menonaktifkan 60% unit secara acak selama pelatihan.
	Dense(128, activation='relu')	Lapisan fully connected dengan 128 unit.
	Dropout(0.3)	Mengurangi overfitting dengan menonaktifkan 30% unit.
	Dense(64, activation='relu')	Lapisan fully connected dengan 64 unit untuk pemrosesan pola lebih mendalam.
	Dropout(0.15)	Mengurangi overfitting dengan menonaktifkan 15% unit.
	Dense(32, activation='relu')	Lapisan fully connected dengan 32 unit untuk penyederhanaan pola lebih lanjut.

	Dropout(0.075)	Mengurangi overfitting dengan menonaktifkan 7.5% unit.
	Dense(16, activation='relu')	Lapisan fully connected dengan 16 unit untuk menyederhanakan pola sebelum prediksi akhir.
	Dropout(0.0325)	Mengurangi overfitting dengan menonaktifkan 3.25% unit.
	Dense(8, activation='relu')	Lapisan fully connected dengan 8 unit sebagai tahap akhir pemrosesan sebelum output.
Output Layer	Dense(len(class_names), activation='softmax')	Lapisan output dengan softmax, menghasilkan probabilitas untuk setiap kelas, di mana jumlah unit sama dengan jumlah kelas

Setelah model selesai dibuat, model akan di-*compile* dengan fungsi yang dapat dilihat pada Tabel 3.2.

Tabel 3. 2 Fungsi untuk Pelatihan Model

No.	Fungsi	Value
1	<i>Optimizer</i>	Adam
2	<i>Learning Rate (Optimizer)</i>	0,001
3	<i>Loss</i>	<i>Categorical Crossentropy</i>
4	<i>Metric</i>	<i>Accuracy</i>
5	Epochs	15

3. *Confusion Matrix*

Confusion matrix digunakan pada tahap evaluasi model untuk mengetahui performanya dalam memprediksi data validasi. *Confusion matrix* terdiri dari nilai *predicted*, yaitu nilai prediksi dan nilai *actual*, yaitu nilai asli dari data. Dari dua nilai ini akan didapatkan empat nilai yang dapat digunakan untuk memastikan apakah

model memprediksi dengan baik, nilai tersebut adalah *True Positive*, *True Negative*, *False Positive* dan *False Negative*.

4. *Evaluation metric*

Evaluasi model merupakan proses penilaian pada model dengan memperhatikan performanya selama pembangunan dan *evaluation metric* adalah pengukuran yang digunakan pada model. Pada tahap evaluasi ini, pengukuran yang digunakan pada sistem adalah akurasi.

3.3 Perhitungan

Pada tahap ini dilakukan analisis perhitungan terhadap proses median filtering, lapisan konvolusi, pooling dan *fully connected*.

1. Median Filtering



1	2	0	4	3	0
0	1	1	3	2	3
4	1	3	0	2	1
1	1	1	4	2	0
0	1	0	1	2	4
3	4	2	2	0	1

Proses median filter dilakukan dengan membuat jendela berukuran ganjil, pada ilustrasi ini ukurannya 3x3 yang akan dicari nilai mediannya dan akan menggantikan nilai pixel yang berada di tengah, dimulai dari posisi (1,1) yaitu pixel bernilai 1 hingga posisi (4,4) yaitu pixel bernilai 2:

$$(1,1) = \text{median}(1,2,0,0,1,1,4,1,3) = \text{median}(0,0,1,1,1,1,2,3,4) = 1$$

$$(1,2) = \text{median}(2,0,4,1,1,3,1,3,0) = \text{median}(0,0,1,1,1,2,3,3,4) = 1$$

$$(1,3) = \text{median}(0,4,3,1,3,2,3,0,2) = \text{median}(0,0,1,2,2,3,3,3,4) = 2$$

$$(1,4) = \text{median}(4,3,0,3,2,3,0,2,1) = \text{median}(0,0,1,2,2,3,3,3,4) = 2$$

$$(2,1) = \text{median}(0,1,1,4,1,3,1,1,1) = \text{median}(0,1,1,1,1,1,1,1,3,4) = 1$$

Proses ini dilanjutkan sampai posisi (4,4) hingga didapat hasil akhir:

1	2	0	4	3	0
---	---	---	---	---	---

0	1	1	2	2	3
4	1	1	2	2	1
1	1	1	2	2	0
0	2	1	2	2	4
3	4	2	2	0	1

2. Konvolusi

1	2	0	4	3	0
0	1	1	2	2	3
4	1	1	2	2	1
1	1	1	2	2	0
0	2	1	2	2	4
3	4	2	2	0	1

Pada lapisan konvolusi dilakukan proses penjumlahan antara pixel awal dengan filter. Untuk analisis ini digunakan filter seperti berikut:

1	0	1
0	1	0
1	0	1

Pixel awal dijumlahkan dengan filter mulai dari posisi (1,1) yaitu pixel bernilai 1 sampai posisi (4,4) yaitu pixel bernilai 2:

$$(1,1) = ((1 \times 1) + (2 \times 0) + (0 \times 1) + (0 \times 0) + (1 \times 1) + (1 \times 0) + (4 \times 1) + (1 \times 0) + (1 \times 1)) = 7$$

$$(1,2) = ((2 \times 1) + (0 \times 0) + (4 \times 1) + (1 \times 0) + (1 \times 1) + (2 \times 0) + (1 \times 1) + (1 \times 0) + (2 \times 1)) = 10$$

$$(1,3) = ((0 \times 1) + (4 \times 0) + (3 \times 1) + (1 \times 0) + (2 \times 1) + (2 \times 0) + (1 \times 1) + (2 \times 0) + (2 \times 1)) = 8$$

$$(1,4) = ((4 \times 1) + (3 \times 0) + (0 \times 1) + (2 \times 0) + (2 \times 1) + (3 \times 0) + (2 \times 1) + (2 \times 0) + (1 \times 1)) = 9$$

$$(2,1) = ((0 \times 1) + (1 \times 0) + (1 \times 1) + (4 \times 0) + (1 \times 1) + (1 \times 0) + (1 \times 1) + (1 \times 0) + (1 \times 1)) = 9$$

Proses ini dilanjutkan sampai posisi (4,4) hingga didapat hasil akhir:

1	2	0	4	3	0
0	7	10	8	9	3
4	7	6	8	9	1
1	7	8	8	11	0
0	9	10	7	7	4
3	4	2	2	0	1

3. Pooling

1	2	0	4	3	0
0	7	10	8	9	3
4	7	6	8	9	1
1	7	8	8	11	0
0	9	10	7	7	4
3	4	2	2	0	1

Pada pooling, pixel dibagi menjadi beberapa bagian kecil yang pada ilustrasi ini ditandai dalam beberapa kelompok warna, pada penelitian ini digunakan max pooling sehingga nilai tertinggi yang akan diambil untuk membentuk pixel baru.

- *Bagian 1* = $\max(0,1,2,7) = 7$
- *Bagian 2* = $\max(0,4,10,8) = 10$
- *Bagian 3* = $\max(3,0,9,3) = 9$
- *Bagian 4* = $\max(4,7,1,7) = 7$
- *Bagian 5* = $\max(6,8,8,8) = 8$
- *Bagian 6* = $\max(9,1,11,0) = 11$
- *Bagian 7* = $\max(0,9,3,4) = 9$
- *Bagian 8* = $\max(10,7,2,2) = 7$
- *Bagian 9* = $\max(7,4,0,1) = 7$

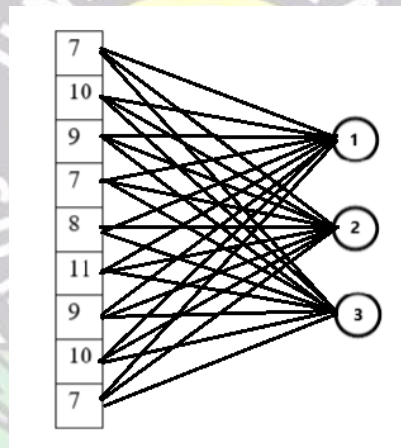
Proses dilakukan untuk semua bagian hingga didapat hasil akhir:

7	10	9
7	8	11
9	10	7

4. Fully Connected

7	10	9
7	8	11
9	10	7

Pada fully connected, output dari lapisan sebelumnya diflatten menjadi (7, 10, 9, 7, 8, 11, 9, 10, 7).



Untuk analisa ini diandaikan ada 3 neuron pada di lapisan fully connected dengan bobot berikut:

- Neuron 1: $W_1 = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$
- Neuron 2: $W_2 = [0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]$
- Neuron 3: $W_3 = [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]$

Dengan bias untuk masing-masing neuron adalah $b_1=1$, $b_2=2$, dan $b_3=0.5$.

Aktivasi dihitung dengan formula:

$$a_i = \text{dot}(W_i, \text{input}) + b_i$$

- Perhitungan Neuron 1

$$a_1 = (0.1 \times 7 + 0.2 \times 10 + 0.3 \times 9 + 0.4 \times 7 + 0.5 \times 8 + 0.6 \times 11 + 0.7 \times 9 + 0.8 \times 10 + 0.9 \times 7) + 1 = 40.4$$

- Perhitungan Neuron 2

$$a_2 = (0.9 \times 7 + 0.8 \times 10 + 0.7 \times 9 + 0.6 \times 7 + 0.5 \times 8 + 0.4 \times 11 + 0.3 \times 9 + 0.2 \times 10 + 0.1 \times 7) + 2 = 40.6$$

- Perhitungan Neuron 3

$$a_3 = (0.5 \times 7 + 0.5 \times 10 + 0.5 \times 9 + 0.5 \times 7 + 0.5 \times 8 + 0.5 \times 11 + 0.5 \times 9 + 0.5 \times 10 + 0.5 \times 7) + 0.5 = 39.5$$

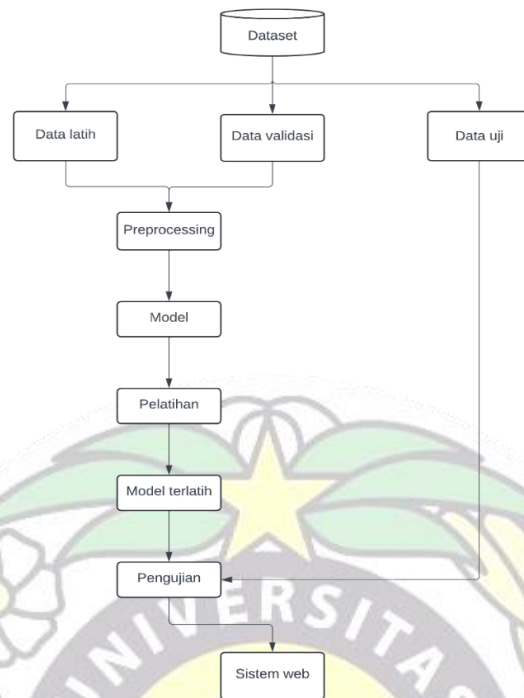
Didapat output dari lapisan fully connected adalah 40.4, 40.6 dan 39.5.

3.4 Pemodelan Sistem

Tahap ini adalah proses analisis langkah-langkah interaksi antara pengguna dan aplikasi yang dikembangkan, dengan tujuan memastikan sistem dapat beroperasi secara efisien. Untuk menggambarkan pemodelan sistem, digunakan *Unified Modeling Language* (UML). UML yang akan diimplementasikan meliputi Diagram Umum Sistem, Use Case dan Activity Diagram.

3.4.1 Diagram Umum Sistem

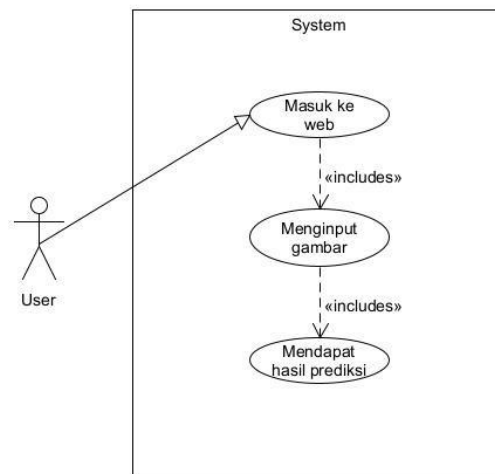
Pada diagram ini dibuat gambaran mengenai cara sistem berjalan dengan menunjukkan interaksi, aliran, dan proses setiap komponen dalam sistem berlangsung. Pertama dataset akan dibagi menjadi data latih, uji dan validasi. Kemudian data latih dan validasi akan dipreprosesing dan digunakan untuk melatih model. Setelah model dilatih akan menghasilkan model terlatih dan dilakukan pengujian menggunakan data uji yang akhirnya model akan digunakan untuk membuat sistem web. Desain sistem dapat dilihat pada Gambar 3.1 berikut:



Gambar 3. 1 Diagram Umum Sistem

3.4.2 Use Case Diagram

Pada diagram ini diberikan gambaran mengenai interaksi pelaku dengan sistem. Pelaku merupakan entitas yang berperan dalam interaksi tersebut, yang bisa berupa manusia, perangkat keras, dan berbagai elemen lainnya. Berikut diagramnya:

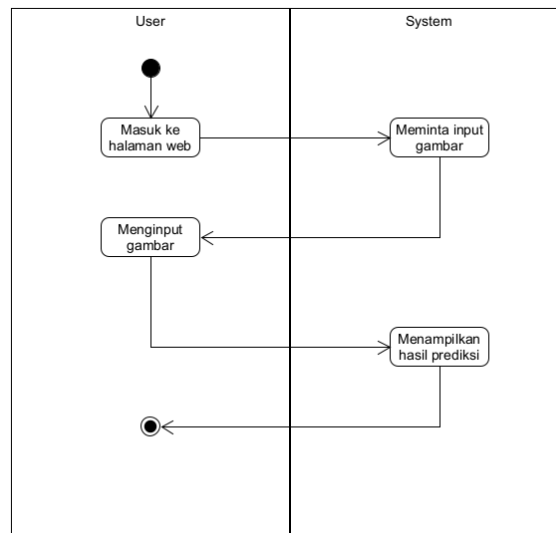


Gambar 3. 2 Diagram Use Case

Pada Gambar 3.2 di atas memperlihatkan, pertama *user* akan masuk ke aplikasi sistem. *User* kemudian menginput gambar untuk diprediksi kondisi ginjal dari gambar tersebut. Setelah itu, *user* akan menerima hasil prediksi dari data yang dimasukkan.

3.4.3 Activity Diagram

Diagram ini menguraikan kegiatan pada sistem. Diagram ini menggambarkan urutan proses kerja mulai awal hingga akhir. Berikut adalah diagramnya:



Gambar 3. 3 Diagram Activity

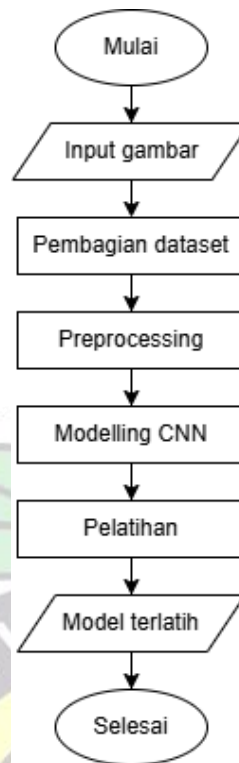
Pada Gambar 3.3 diperlihatkan proses yang awalnya pengguna mengakses halaman beranda. Selanjutnya, sistem meminta pengguna untuk mengunggah gambar. Setelah pengguna menginput gambar, sistem akan menampilkan hasil prediksi berdasarkan gambar yang telah diunggah.

3.5 Flowchart

Flowchart adalah representasi visual alur proses sebuah sistem.

3.5.1 Flowchart Pelatihan lightweight CNN

Berikut flowchart sederhana dari pelatihan model:

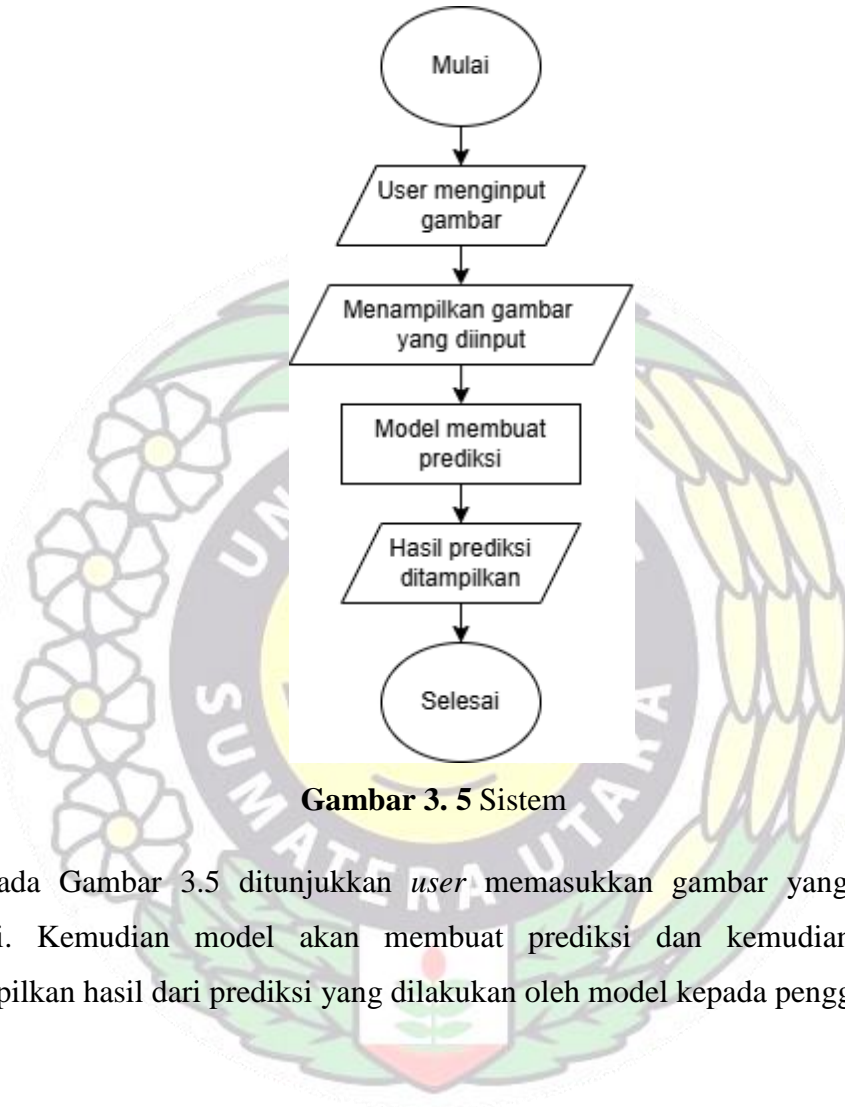


Gambar 3. 4 Pelatihan Lighweight CNN

Pada Gambar 3.4 ditunjukkan *user* menginputkan gambar. Gambar tersebut kemudian akan dibagi. Kemudian gambar di-*preprocess* dengan melakukan *resizing*, *grayscale*, penerapan median filter dan normalisasi. Setelah dataset melalui preprocessing akan dibuat model CNN untuk dilatih sehingga menghasilkan sebuah model CNN yang terlatih.

3.5.2 Flowchart Sistem

Di bawah adalah flowchart sederhana sistem yang digunakan:



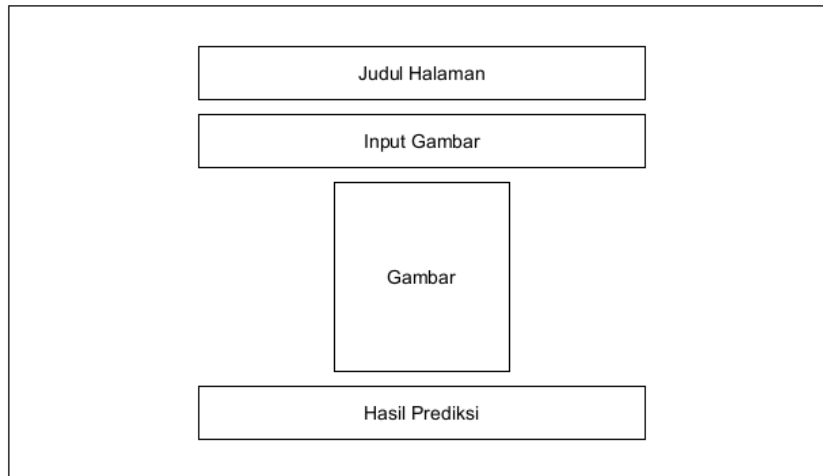
Gambar 3. 5 Sistem

Pada Gambar 3.5 ditunjukkan *user* memasukkan gambar yang ingin di prediksi. Kemudian model akan membuat prediksi dan kemudian website menampilkan hasil dari prediksi yang dilakukan oleh model kepada pengguna.

3.6 Perancangan *Interface*

Perancangan *Interface* adalah proses menciptakan kerangka desain sistem. Perancangan *Interface* diperlukan untuk memastikan proses pembuatan sistem efektif berdasarkan apa yang telah didefinisikan dalam desain antarmuka.

3.6.1 Halaman Prediksi dengan Gambar



Gambar 3. 6 Halaman Prediksi dengan Gambar

Gambar 3.6 diatas menunjukkan rancangan *interface* dari halaman prediksi. Pada halaman ini, *user* akan memasukkan gambar, kemudian gambar tersebut akan ditampilkan dan di bawahnya akan ditampilkan hasil prediksi dari gambar.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Sistem yang dirancang memanfaatkan *lightweight* CNN dan komponen front-endnya dibangun menggunakan Streamlit dan back-end dikembangkan dengan bantuan bahasa pemrograman Python.

4.1.1 Halaman Utama



Gambar 4. 1 Tampilan Halaman Utama

Gambar 4. 1 menampilkan laman utama sistem yang merupakan tampilan yang muncul pada saat sistem diakses. Di sini, pengguna dapat mengunggah gambar yang akan diprediksi.

4.1.2 Pelatihan Model

Berikut tahapan-tahapan yang dilakukan yang dilakukan dalam pelatihan model.

a. *Import Library*

```
!pip install unrar
!pip install split-folders
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import Model
import os
from google.colab import drive
import matplotlib.pyplot as plt
import splitfolders
import cv2
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay
import seaborn as sns
```

Gambar 4. 2 Import Library

Pada Gambar 4.2 menunjukkan proses *import library* yang akan diperlukan. Pada model, diperlukan beberapa library untuk digunakan pada pelatihan model. Library yang diperlukan antara lain yaitu, tensorflow, keras, os, matplotlib, splitfolders, cv2, numpy, sklearn dan seaborn.

b. Pembagian Dataset

```
input_folder = "/content/data"
output_folder = "/content/split"

# Membagi dataset menjadi pelatihan, validasi, dan pengujian
splitfolders.ratio(
    input_folder,
    output=output_folder,
    seed=42,
    ratio=(0.8, 0.1, 0.1), # 80% train, 10% val, 10% test
    group_prefix=None
)

Copying files: 5360 files [00:04, 1130.55 files/s]
```

Gambar 4. 3 Pembagian Dataset

Gambar 4.3 di atas menampilkan tahapan memuat dataset yang masih dalam bentuk arsip terkompresi .rar kemudian diekstrak dan dibagi data latih, validasi dan data uji.

c. Preprocessing

```

train_dir = "/content/split/train"
val_dir = "/content/split/val"
test_dir = "/content/split/test"

#Fungsi median filter
def apply_median_filter(directory, target_size=(224, 224)):
    processed_images = []
    labels = []
    class_names = sorted(os.listdir(directory))
    for class_index, class_name in enumerate(class_names):
        class_folder = os.path.join(directory, class_name)
        for file_name in os.listdir(class_folder):
            file_path = os.path.join(class_folder, file_name)
            image = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
            if image is not None:
                image = cv2.resize(image, target_size)
                image = cv2.medianBlur(image, 5)
                processed_images.append(image)
                labels.append(class_index)
    return np.array(processed_images), np.array(labels), class_names

# Fungsi median filter tetap sama
train_images, train_labels, class_names = apply_median_filter(train_dir)
val_images, val_labels, _ = apply_median_filter(val_dir)
test_images, test_labels, _ = apply_median_filter(test_dir)

# Normalisasi
train_images = train_images / 255.0
val_images = val_images / 255.0
test_images = test_images / 255.0

# Ubah dimensi menjadi (N, H, W, 1) karena gambar grayscale
train_images = np.expand_dims(train_images, axis=-1)
val_images = np.expand_dims(val_images, axis=-1)
test_images = np.expand_dims(test_images, axis=-1)

# Ubah label ke one-hot encoding
train_labels = tf.keras.utils.to_categorical(train_labels, num_classes=len(class_names))
val_labels = tf.keras.utils.to_categorical(val_labels, num_classes=len(class_names))
test_labels = tf.keras.utils.to_categorical(test_labels, num_classes=len(class_names))

```

Gambar 4. 4 Preprocessing

Pada Gambar 4.4 menunjukkan tahapan *preprocessing* yang akan dilakukan pada gambar yang akan digunakan untuk pelatihan, diantaranya adalah *resizing*, *grayscale*, penerapan median filter dan normalisasi.

d. Pembangunan model

Model kemudian dibangun sesuai dengan arsitektur yang dipilih yaitu *lightweight CNN*. Pembangunan model dapat dilihat pada Gambar 4.5.

```

img_input = layers.Input(shape=(224, 224, 1))

#Layer1
x = layers.Conv2D(64, 5, activation="relu")(img_input)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Layer2
x = layers.Conv2D(64, 3, activation="relu")(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Layer3
x = layers.Conv2D(96, 3, activation="relu")(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Layer4
x = layers.Conv2D(96, 3, activation="relu")(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Fully Connected
x = layers.Flatten()(x)
x = layers.Dense(256, activation='relu')(x)
x = layers.Dropout(0.6)(x)
x = layers.Dense(128, activation='relu')(x)
x = layers.Dropout(0.3)(x)
x = layers.Dense(64, activation='relu')(x)
x = layers.Dropout(0.15)(x)
x = layers.Dense(32, activation='relu')(x)
x = layers.Dropout(0.075)(x)
x = layers.Dense(16, activation='relu')(x)
x = layers.Dropout(0.0325)(x)
x = layers.Dense(8, activation='relu')(x)

output = layers.Dense(len(class_names), activation='softmax')(x)
model = Model(img_input, output)

```

Gambar 4. 5 Pembangunan Model

e. Kompilasi dan Pelatihan Model

```

#Kompilasi
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

#Latih
history = model.fit(
    train_images, train_labels,
    epochs=15,
    validation_data=(val_images, val_labels)
)

```

Gambar 4. 6 Kompilasi dan Pelatihan Model

Pada Gambar 4.6 menunjukkan tahap kompilasi model menggunakan *optimizer* Adam, fungsi *loss categorical_crossentropy* dan *metric* akurasi untuk menilai kinerja model selama pelatihan. Model kemudian dilatih sebanyak 15

epochs.

f. Proses Pelatihan

```
Epoch 5/15
134/134 ————— 10s 74ms/step - accuracy: 0.6082 - loss: 0.7898 - val_accuracy: 0.7854 - val_loss: 0.5405
Epoch 6/15
134/134 ————— 10s 76ms/step - accuracy: 0.7550 - loss: 0.5612 - val_accuracy: 0.6978 - val_loss: 0.6854
Epoch 7/15
134/134 ————— 10s 73ms/step - accuracy: 0.8221 - loss: 0.4596 - val_accuracy: 0.7966 - val_loss: 0.5717
Epoch 8/15
134/134 ————— 10s 73ms/step - accuracy: 0.8967 - loss: 0.3149 - val_accuracy: 0.8937 - val_loss: 0.2644
Epoch 9/15
134/134 ————— 10s 74ms/step - accuracy: 0.9263 - loss: 0.2162 - val_accuracy: 0.9720 - val_loss: 0.1136
Epoch 10/15
134/134 ————— 10s 74ms/step - accuracy: 0.9431 - loss: 0.1572 - val_accuracy: 0.9683 - val_loss: 0.1097
Epoch 11/15
134/134 ————— 10s 73ms/step - accuracy: 0.9342 - loss: 0.2172 - val_accuracy: 0.9832 - val_loss: 0.0502
Epoch 12/15
134/134 ————— 10s 74ms/step - accuracy: 0.9724 - loss: 0.0785 - val_accuracy: 0.9944 - val_loss: 0.0214
Epoch 13/15
134/134 ————— 10s 74ms/step - accuracy: 0.9768 - loss: 0.0815 - val_accuracy: 0.9944 - val_loss: 0.0163
Epoch 14/15
134/134 ————— 10s 73ms/step - accuracy: 0.9641 - loss: 0.1332 - val_accuracy: 0.9757 - val_loss: 0.0853
Epoch 15/15
134/134 ————— 10s 75ms/step - accuracy: 0.9875 - loss: 0.0503 - val_accuracy: 0.9944 - val_loss: 0.0247
```

Gambar 4. 7 Pelatihan

Gambar 4.7 menunjukkan proses pelatihan. Hasil akhir dari proses pelatihan menunjukkan akurasi 98,75% untuk data latih dan 99,44% untuk data validasi. Lamanya proses pelatihan model adalah 163 detik.

4.1.3 Halaman Input dan Prediksi

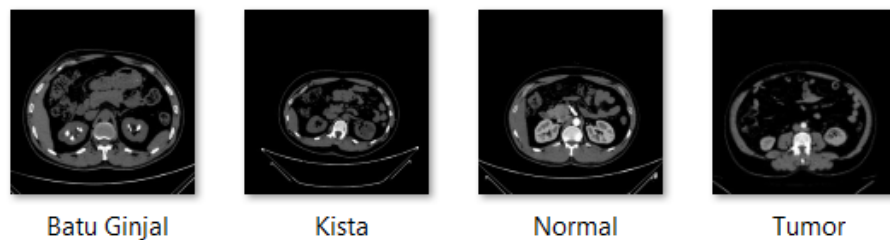
Di halaman ini, *user* akan melakukan pemasukan data berupa gambar dengan cara mengunggah atau mengarahkan gambar yang ingin diprediksi ke dalam area berbentuk persegi panjang dengan warna yang lebih gelap pada website. Setelah *user* menginput gambar, sistem akan memberikan hasil prediksi dan nilai *confidence*. Tampilan halaman ada di Gambar 4.8.



Gambar 4. 8 Tampilan Input dan Prediksi

4.2 Sumber Data

Data citra CT Scan ginjal diperoleh melalui situs Kaggle (Islam, 2022). Data yang digunakan berasal dari *Picture archiving and communication system*) dan *workstations* dari sebuah rumah sakit di Dhaka, Bangladesh. Dataset ini terdiri dari 12446 gambar yang berisi 4 kelas penyakit ginjal. Untuk kepentingan penelitian ini kemudian diambil 1340 gambar untuk setiap kelas dengan total 5360 gambar. Sampel citra dari masing-masing kelas ditampilkan di Gambar 4.9.



Gambar 4. 9 Citra CT Scan

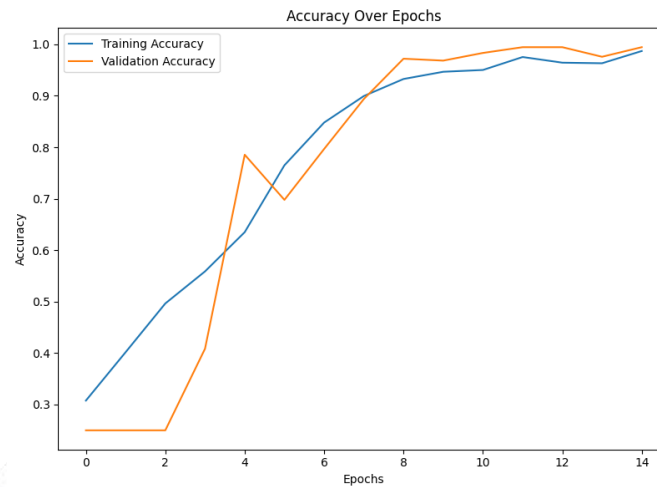
4.3 Pengujian sistem

Pengujian sistem adalah tahapan yang akan dilakukan setelah implementasi arsitektur dilakukan, pengujian sistem untuk menunjukkan sistem bekerja baik dalam memprediksi kondisi ginjal melalui citra *CT Scan*. Pada tahap pengujian, terdapat 4 kondisi yang akan diprediksi, diantaranya ginjal dengan penyakit batu ginjal, kista

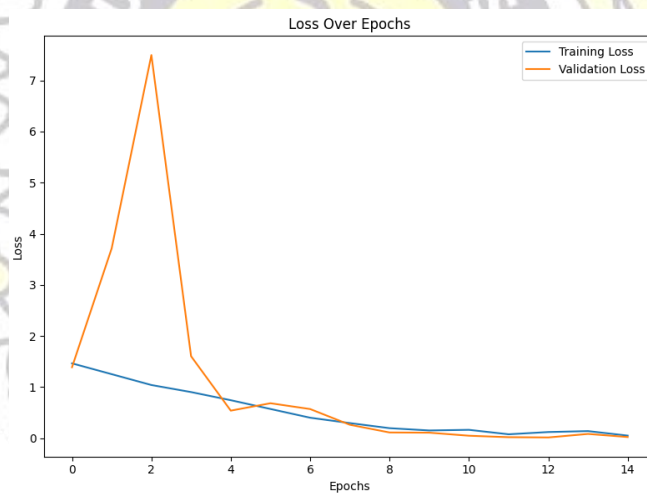
dan tumor, dan ginjal dengan kondisi normal. Berikut adalah langkah-langkah yang dilakukan dalam pengujian sistem.



a. Accuracy Plot



Gambar 4. 10 Grafik Traning dan Validation Accuracy



Gambar 4. 11 Grafik Traning dan Validation Loss

Gambar 4.10 dan 4.11 adalah grafik hasil dari perhitungan terhadap *accuracy* dan *loss* dari proses pelatihan model dan juga validasinya. Pada grafik ini dapat dilihat bahwa kedua garis terlihat saling beriringan yang artinya model berhasil melakukan prediksi dengan baik menggunakan data latih dan data validasi.

b. Pengujian model

Model diuji dengan data yang sudah dibagikan sebanyak 10% dari keseluruhan data untuk digunakan sebagai data uji. Hasil pengujian menunjukkan akurasi sebesar 99,62%. Hasil dari pengujian model dapat dilihat pada Gambar 4.12.

```
# Evaluasi model pada data uji
test_loss, test_accuracy = model.evaluate(test_images, test_labels)
print(f"Test Loss: {test_loss}, Test Accuracy: {test_accuracy}")
```

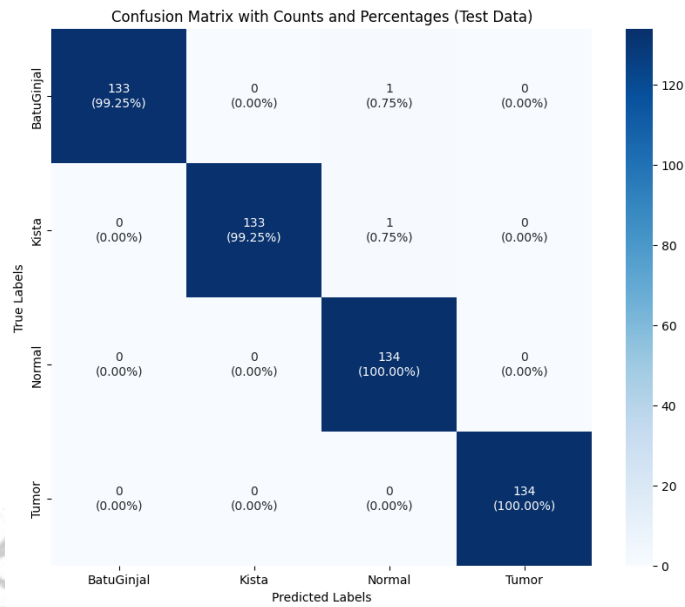
17/17 ————— 0s 21ms/step - accuracy: 0.9955 - loss: 0.0320
Test Loss: 0.024072498083114624, Test Accuracy: 0.996268630027771

Gambar 4. 12 Hasil Pengujian Model

Perbandingan kinerja *lightweight* CNN dengan model lain dalam mengklasifikasi penyakit ginjal.

Model	Jumlah Data	Akurasi
Lightweight CNN	536	99,26%
VGG19	1867	98,77%
Xception	1867	98,10%

c. Confusion Matrix



Gambar 4. 13 Confusion Matrix

Dari hasil *Confusion Matrix* pada Gambar 4.13, dapat diambil kesimpulan bahwa model telah berhasil memprediksi data sesuai dengan nilai aktualnya.

d. Evaluation metrics

Confusion Matrix dapat digunakan untuk mendapatkan *evaluation metrics*.

Kelas	TP	FP	TN	FN
Batu Ginjal	133	0	402	1
Kista	133	0	402	1
Normal	134	2	401	0
Tumor	134	0	402	0

- Akurasi

Akurasi yaitu persentase dari keseluruhan data yang berhasil diklasifikasikan dengan benar.

$$\text{Akurasi} = \frac{\text{TP}}{\text{total data}}$$

$$\text{Akurasi} = \frac{133 + 133 + 134 + 134}{536} = \frac{534}{536} = 0,99,62 = 99,62\%$$

- Precision

Precision adalah metrik yang mengevaluasi tentang seberapa baik model dapat menghasilkan hasil tepat berdasarkan pada informasi yang tidak tepata.

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

$$Precision \text{ Batu Ginjal} = \frac{133}{133} = 100\%$$

$$Precision \text{ Kista} = \frac{133}{133} = 100\%$$

$$Precision \text{ Normal} = \frac{134}{136} = 98,52\%$$

$$Precision \text{ Tumor} = \frac{134}{134} = 100\%$$

$$Mean Precision = \frac{100 + 100 + 98,52 + 100}{4} = 99,63\%$$

- *Recall*

Recall merupakan perbandingan antara jumlah prediksi positif yang benar dan total data aktual yang positif.

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

$$Recall \text{ Batu Ginjal} = \frac{133}{134} = 99,25\%$$

$$Recall \text{ Kista} = \frac{133}{134} = 99,25\%$$

$$Recall \text{ Normal} = \frac{134}{134} = 100\%$$

$$Recall \text{ Tumor} = \frac{134}{134} = 100\%$$

$$Mean Recall = \frac{99,25 + 99,25 + 100 + 100}{4} = 99,62\%$$

- *F-1 Score*

F1-score yaitu nilai mean harmoni presisi dengan recall.

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

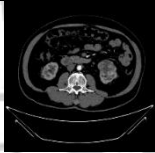
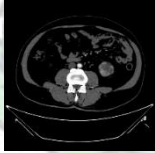
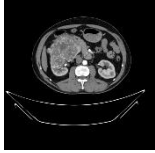

$$F1 - Score = 2 \times \frac{(0,9963 \times 0,9962)}{(0,9963 + 0,9962)} = 2 \times \frac{0,9925}{1,9925} = 0,9962 = 99,62\%$$


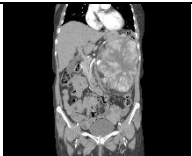
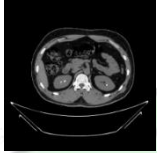
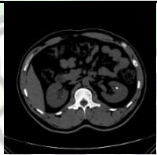





Berdasarkan perhitungan yang telah dilakukan, maka disimpulkan *lightweight* CNN dalam melakukan klasifikasi mendapat akurasi sebesar 99,62%, precision sebesar 99,63%, recall sebesar 99,62%, dan F1-Score sebesar 99,62%.

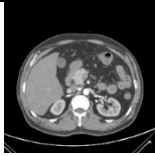
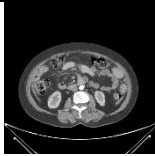



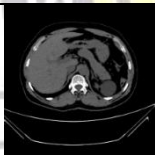
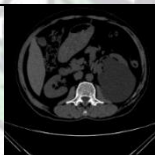


e. Data Uji Sistem

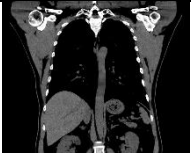

Tahap ini adalah penentuan data uji untuk pengujian sistem web, data ini tidak termasuk dalam data latih maupun validasi dan data uji sebelumnya. Data ujinya dapat dilihat pada Tabel 4.1 berikut.

Tabel 4. 1 Data Uji

No.	Gambar	Kelas
1		Tumor
2		Tumor
3		Tumor
4		Tumor

5		Tumor
6		Tumor
7		Batu ginjal
8		Batu ginjal
9		Batu ginjal
10		Batu ginjal
11		Batu ginjal
12		Batu ginjal
13		Normal

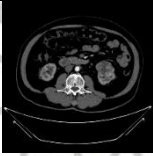
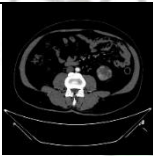
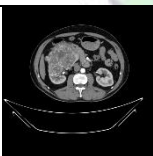

14		Normal
15		Normal
16		Normal
17		Normal
18		Normal
19		Kista
20		Kista
21		Kista
22		Kista


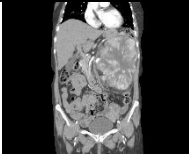
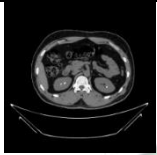
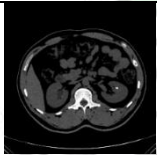





23		Kista
24		Kista

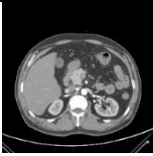




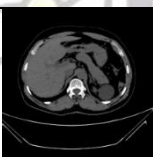
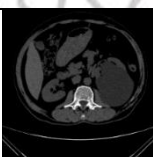


4.4 Hasil Pengujian Sistem

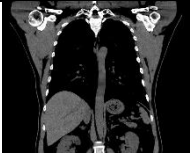

Setelah dilakukan pengujian sistem, hasil yang didapat dengan mencoba semua data uji adalah:

Tabel 4. 2 Hasil Pengujian Sistem

No.	Gambar	Kelas	Hasil Prediksi
1		Tumor	Tumor
2		Tumor	Tumor
3		Tumor	Tumor
4		Tumor	Tumor

5		Tumor	Tumor
6		Tumor	Tumor
7		Batu ginjal	Batu ginjal
8		Batu ginjal	Batu ginjal
9		Batu ginjal	Batu ginjal
10		Batu ginjal	Batu ginjal
11		Batu ginjal	Batu ginjal
12		Batu ginjal	Batu ginjal
13		Normal	Normal

14		Normal	Normal
15		Normal	Normal
16		Normal	Normal
17		Normal	Normal
18		Normal	Normal
19		Kista	Kista
20		Kista	Kista
21		Kista	Kista
22		Kista	Kista

23		Kista	Kista
24		Kista	Kista

Dari hasil Tabel 4.2 di atas maka dapat diambil kesimpulan, yaitu model dapat mengklasifikasi kondisi ginjal sesuai dengan kondisi aslinya.



4.5 User Acceptance Testing

User Acceptance Testing (UAT) yaitu tahap pengujian terakhir sistem untuk memastikan bahwa sistem tersebut dapat diterima dan memenuhi kebutuhan pengguna. Proses UAT disusun dalam bentuk tabel yang berisi pernyataan, yang kemudian diisi oleh pakar, yaitu Dr. M. Feldi Gazaly, M. Ked (PD), Sp. PD, K-GH, yang bertindak sebagai penguji sekaligus evaluator sistem klasifikasi penyakit ginjal yang telah dikembangkan. Hasil dari UAT dapat dilihat di Tabel 4.3.

Tabel 4. 3 User Acceptance Testing

Pernyataan	Sangat Tidak Setuju	Tidak Setuju	Netral	Setuju	Sangat Setuju
Sistem tidak memiliki kesulitan dalam penggunaannya					✓
Sistem klasifikasi menghasilkan diagnosa penyakit dalam waktu yang singkat					✓
Sistem klasifikasi penyakit ginjal menghasilkan diagnosa yang sesuai dengan diagnosa pakar				✓	
Sistem klasifikasi penyakit ginjal layak digunakan sesuai dengan kebutuhan medis					✓
Sistem layak dikembangkan untuk dapat mendiagnosis lebih banyak jenis penyakit				✓	

Dari hasil Tabel 4,3, dapat dinyatakan bahwa sistem klasifikasi penyakit pada ginjal tidak memiliki kesulitan untuk digunakan, menghasilkan diagnosa penyakit yang sesuai dengan diagnosa pakar, layak digunakan sesuai kebutuhan medis, dan layak dikembangkan untuk dapat mendiagnosis lebih banyak jenis penyakit ginjal.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang diperoleh dari penelitian yang telah dilakukan adalah sebagai berikut.

1. Arsitektur *lightweight* CNN dapat digunakan untuk memprediksi penyakit pada ginjal melalui citra *CT Scan* dengan akurasi sebesar 98,75% pada data latih, 99,44% pada data validasi dan 99,26% pada data latih.
2. Jumlah data dan pembagian data yang digunakan pada saat pelatihan mempengaruhi metode CNN dalam mendapatkan model yang stabil.
3. Jumlah epoch, optimizer, learning rate dan runtime dapat mempengaruhi kinerja model.
4. Prediksi penyakit pada ginjal melalui CT Scan dapat menjadi acuan dalam memeriksa kondisi ginjal dengan lebih akurat.

5.2 Saran

Hal yang bisa dijadikan bahan pertimbangan bagi penelitian yang terkait adalah:

1. Memperbanyak kuantitas data agar model menjadi lebih sempurna.
2. Memperbanyak jenis penyakit yang diperlukan untuk diprediksi apabila data yang dikumpulkan dinilai cukup untuk dimasukkan ke dalam model.
3. Diharapkan kedepannya dapat bekerja sama dengan instansi yang berkaitan untuk memperkuat sistem agar berguna bagi petugas medis.

Daftar Pustaka

- Alamsyah, A. P. D. (2019). Sistem Pakar Diagnosa Penyakit Ginjal. *International Journal of Artificial Intelligence*, 6(1), 53-74.
- Almuayqil, S. N., Abd El-Ghany, S., Abd El-Aziz, A. A., & Elmogy, M. (2024). KidneyNet: A Novel CNN-Based Technique for the Automated Diagnosis of Chronic Kidney Diseases from CT Scans. *Electronics*, 13(24), 4981.
- Amani, P., Adriani, D., Putri, M. A., & Imran, Y. (2022). Penyuluhan risiko penyakit ginjal kronis pada pasien hipertensi prolanis putewa Jakarta Timur. *Jurnal Kreativitas Pengabdian Kepada Masyarakat (PKM)*, 5(10), 3287-3295.
- Bisong, E., & Bisong, E. (2019). What Is Deep Learning?. Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners, 327-329.
- Fadlurrahman, M. N., Minarno, A. E., & Azhar, Y. (2023). Klasifikasi Covid19 Menggunakan Algoritma Cnn. *Jurnal Repositor*, 5(2).
- Gunasekara, T. D. K. S. C., De Silva, P. M. C., Ekanayake, E. M. D. V., Thakshila, W.A. K. G., Pinipa, R. A. I., Sandamini, P. M. M. A., ... & Jayasundara, N. (2022). Urinary biomarkers indicate pediatric renal injury among rural farming communities in Sri Lanka. *Scientific Reports*, 12(1), 8040.
- Halodoc. 2021, 4 Oktober. 10 Gejala Awal Penyakit Ginjal Ini Perlu Diwaspadai. Diakses pada 7 Desember 2024, dari <https://www.halodoc.com/artikel/10-gejala-awal-penyakit-ginjal-ini-perlu-diwaspadai>
- Islam, M. N., Hasan, M., Hossain, M. K., Alam, M. G. R., Uddin, M. Z., & Soylyu, A. (2022). Vision transformer and explainable transfer learning

- models for auto detection of kidney cyst, stone and tumor from CT-radiography. *Scientific Reports*, 12(1), 1-14.
- Kosasih, R. (2021). Pendeteksian Kendaraan Menggunakan Metode Median Filter: Array. *Jurnal Ilmiah Komputasi*, 20(1), 53-58.
- Kurniawan, R., Wintoro, P. B., Mulyani, Y., & Komarudin, M. (2023). Implementasi Arsitektur Xception Pada Model Machine Learning Klasifikasi Sampah Anorganik. *Jurnal Informatika dan Teknik Elektro Terapan*, 11(2).
- Marcella, D., Yohannes, Y., & Devella, S. (2022). Klasifikasi penyakit mata menggunakan Convolutional Neural Network dengan arsitektur VGG-19. *Jurnal Algoritme*, 3(1), 60-70.
- Nashrullah, F., Wibowo, S. A., & Budiman, G. (2020). Investigasi parameter epoch pada arsitektur resnet-50 untuk klasifikasi pornografi. *Journal of Computer, Electronic, and Telecommunication*, 1(1), 1-8.
- P. D. Kusuma, Machine Learning Teori, Program, dan Studi Kasus. Yogyakarta: Deepublish, 2020.
- Putra, H. W. (2019). Sistem Pakar Diagnosis Penyakit Ginjal Dengan Metoda Forward Chaining. *Jurnal Sains dan Informatika: Research of Science and Informatic*, 5(1), 7-12.
- Ray, S., Alshouiliy, K., & Agrawal, D. P. (2020). Dimensionality reduction for human activity recognition using google colab. *Information*, 12(1), 6.
- Retnoningsih, E., & Pramudita, R. (2020). Mengenal machine learning dengan teknik supervised dan unsupervised learning menggunakan python. *Bina Insani Ict Journal*, 7(2), 156-165.
- Rguibi, Z., Hajami, A., Zitouni, D., Elqaraoui, A., & Bedraoui, A. (2022). Cxai: Explaining convolutional neural networks for medical imaging diagnostic. *Electronics*, 11(11), 1775.
- Riyadi, A. S., Wardhani, I. P., & Widayati, S. (2021, September). Klasifikasi citra anjing dan kucing menggunakan metode convolutional neural

- network (CNN). In *Prosiding Seminar SeNTIK* (Vol. 5, No. 1, pp. 307-311).
- Siregar, E. S., Sutapa, N., & Sudarsana, I. W. B. (2019). Penentuan Dosis Efektif Pada Pemeriksaan CT Scan Kepala Anak Dengan Software Indose CT. *Kappa Journal*, 3(2), 113-117.
- Shuvo, S. B., Ali, S. N., Swapnil, S. I., Hasan, T., & Bhuiyan, M. I. H. (2020). A lightweight cnn model for detecting respiratory diseases from lung auscultation sounds using emd-cwt-based hybrid scalogram. *IEEE Journal of Biomedical and Health Informatics*, 25(7), 2595-2603.
- Sunija, A. P., Kar, S., Gayathri, S., Gopi, V. P., & Palanisamy, P. (2021). Octnet: A lightweight cnn for retinal disease classification from optical coherence tomography images. *Computer methods and programs in biomedicine*, 200, 105877.
- Suta, I. B. L. M., Hartati, R. S., & Divayana, Y. (2019). Diagnosa tumor otak berdasarkan citra MRI (Magnetic Resonance Imaging). *Maj. Ilm. Teknol. Elektro*, 18(2), 149-154.
- Syahputra, S. A. F., Azizah, N. M., Aiman, J., Nikmah, D. A., & Rosyani, P. (2024). Identifikasi dan Prediksi Umur Berdasarkan Citra Wajah Menggunakan Deep Learning Algoritma Convolutional Neural Network (CNN). *AI dan SPK: Jurnal Artificial Intelligent dan Sistem Penunjang Keputusan*, 2(1), 87-95.
- Thakur, P. S., Sheorey, T., & Ojha, A. (2023). VGG-ICNN: A Lightweight CNN model for crop disease identification. *Multimedia Tools and Applications*, 82(1), 497-520.
- Trilaksono, B. R., Riza, H., Jarin, A., Darmayanti, N. D. S., & Liawatimena, S. (Eds.). (2023). *Prosiding Use Cases Artificial Intelligence Indonesia: Embracing Collaboration for Research and Industrial Innovation in Artificial Intelligence*. Penerbit BRIN.

- Wang, R., Wang, H., He, Z., Zhu, J., & Zuo, H. (2024). WeldNet: alightweight deep learning model for welding defect recognition. *Welding in the World*, 1-12.
- Widodo, S. (2022). Klasifikasi Kanker dan Artery pada Citra Computed Tomography Menggunakan Deep Learning Convolution Neural Network. *Indonesian of Health Information Management Journal (INOHIM)*, 10(2), 94-103.



Lampiran

Lampiran 1 Kode Program

Model.ipynb

```
!pip install unrar
!pip install split-folders
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import Model
import os
from google.colab import drive
import matplotlib.pyplot as plt
import splitfolders
import cv2
import numpy as np
from sklearn.metrics import classification_report,
confusion_matrix, ConfusionMatrixDisplay
import seaborn as sns

!mkdir -p /content/data
!mkdir -p /content/split

drive.mount("/content/drive")

!unrar x /content/drive/MyDrive/data.rar /content/data

input_folder = "/content/data"
output_folder = "/content/split"

# Membagi dataset menjadi pelatihan, validasi, dan pengujian
splitfolders.ratio(
    input_folder,
    output=output_folder,
```



```

    seed=42,
    ratio=(0.8, 0.1, 0.1), # 80% train, 10% val, 10% test
    group_prefix=None
)

train_dir = "/content/split/train"
val_dir = "/content/split/val"
test_dir = "/content/split/test"

#Fungsi median filter
def apply_median_filter(directory, target_size=(224, 224)):
    processed_images = []
    labels = []
    class_names = sorted(os.listdir(directory))
    for class_index, class_name in enumerate(class_names):
        class_folder = os.path.join(directory, class_name)
        for file_name in os.listdir(class_folder):
            file_path = os.path.join(class_folder, file_name)
            image = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
            if image is not None:
                image = cv2.resize(image, target_size)
                image = cv2.medianBlur(image, 5)
                processed_images.append(image)
                labels.append(class_index)
    return np.array(processed_images), np.array(labels),
class_names

# Fungsi median filter
train_images, train_labels, class_names =
apply_median_filter(train_dir)
val_images, val_labels, _ = apply_median_filter(val_dir)
test_images, test_labels, _ = apply_median_filter(test_dir)

# Normalisasi
train_images = train_images / 255.0

```

```

val_images = val_images / 255.0
test_images = test_images / 255.0

# Ubah dimensi menjadi (N, H, W, 1)
train_images = np.expand_dims(train_images, axis=-1)
val_images = np.expand_dims(val_images, axis=-1)
test_images = np.expand_dims(test_images, axis=-1)

# Ubah label ke one-hot encoding
train_labels = tf.keras.utils.to_categorical(train_labels,
num_classes=len(class_names))
val_labels = tf.keras.utils.to_categorical(val_labels,
num_classes=len(class_names))
test_labels = tf.keras.utils.to_categorical(test_labels,
num_classes=len(class_names))

#model
img_input = layers.Input(shape=(224, 224, 1))

#Layer1
x = layers.Conv2D(64, 5, activation="relu")(img_input)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Layer2
x = layers.Conv2D(64, 3, activation="relu")(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Layer3
x = layers.Conv2D(96, 3, activation="relu")(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Layer4

```

```

x = layers.Conv2D(96, 3, activation="relu")(x)
x = layers.BatchNormalization()(x)
x = layers.MaxPooling2D(2)(x)

#Fully Connected
x = layers.Flatten()(x)
x = layers.Dense(256, activation='relu')(x)
x = layers.Dropout(0.6)(x)
x = layers.Dense(128, activation='relu')(x)
x = layers.Dropout(0.3)(x)
x = layers.Dense(64, activation='relu')(x)
x = layers.Dropout(0.15)(x)
x = layers.Dense(32, activation='relu')(x)
x = layers.Dropout(0.075)(x)
x = layers.Dense(16, activation='relu')(x)
x = layers.Dropout(0.0325)(x)
x = layers.Dense(8, activation='relu')(x)

output = layers.Dense(len(class_names), activation='softmax')(x)

model = Model(img_input, output)

#Kompilasi
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

#Latih
history = model.fit(
    train_images, train_labels,
    epochs=15,
    validation_data=(val_images, val_labels)
)

```

```

# Evaluasi model pada data uji
test_loss, test_accuracy = model.evaluate(test_images,
test_labels)
print(f"Test Loss: {test_loss}, Test Accuracy: {test_accuracy}")

model.save('/content/model.h5')

#Plot loss
plt.figure(figsize=(12, 6))
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.title('Loss Over Epochs')
plt.legend()
plt.show()

#Plot akurasi
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Accuracy Over Epochs')
plt.legend()
plt.show()

# Mendapatkan label sebenarnya dari data uji
test_true_classes = np.argmax(test_labels, axis=1)

# Mendapatkan prediksi model untuk data uji
test_predictions = model.predict(test_images)
test_predictions_classes = np.argmax(test_predictions, axis=1)

```

```

# Hitung confusion matrix
conf_matrix = confusion_matrix(test_true_classes,
test_predictions_classes)

# Hitung persentase untuk setiap elemen dalam confusion matrix
conf_matrix_percentage = conf_matrix / conf_matrix.sum(axis=1,
keepdims=True) * 100

# Menggabungkan jumlah dan persentase dalam satu matriks string
annot = np.empty_like(conf_matrix, dtype=object)
for i in range(conf_matrix.shape[0]):
    for j in range(conf_matrix.shape[1]):
        annot[i, j] = f"{conf_matrix[i,
j]}\n({conf_matrix_percentage[i, j]:.2f}%)"

# Visualisasi dengan seaborn heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(
    conf_matrix,
    annot=annot,
    fmt="",
    cmap="Blues",
    xticklabels=class_names,
    yticklabels=class_names
)

plt.title("Confusion Matrix with Counts and Percentages (Test
Data)")
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.show()

```

App.py


```

import streamlit as st
from tensorflow.keras.models import load_model
import numpy as np
from PIL import Image
import cv2
import base64
from io import BytesIO
import requests
import os

st.set_page_config(
    page_title="KidneyClassify",
    page_icon="🩺",
    layout="centered",
    initial_sidebar_state="auto"
)

#fungsi untuk tampilkan gambar
def image_to_base64(image):
    buffered = BytesIO()
    image.save(buffered, format="PNG")
    return base64.b64encode(buffered.getvalue()).decode("utf-8")

# fungsi preprocessing
def preprocess_image(image, target_size):
    image = image.resize(target_size)
    image = image.convert("L")
    image_array = np.array(image)
    image_array = cv2.medianBlur(image_array, 5)
    image_array = image_array / 255.0
    image_array = np.expand_dims(image_array, axis=-1)
    image_array = np.expand_dims(image_array, axis=0)
    return image_array

#URL model

```

```

MODEL_URL =
"https://raw.githubusercontent.com/jeremyfelix12/KidneyClassify/main/model.h5"
MODEL_PATH = "model.h5"

#download model
if not os.path.exists(MODEL_PATH):
    with open(MODEL_PATH, "wb") as file:
        response = requests.get(MODEL_URL)
        file.write(response.content)

#load model
model = load_model(MODEL_PATH)

class_labels = ['Batu Ginjal', 'Kista', 'Normal', 'Tumor']

st.title("Klasifikasi Penyakit Ginjal")

#upload gambar
uploaded_file = st.file_uploader("Silahkan Upload Gambar",
type=["jpg", "jpeg", "png"])

if uploaded_file is not None:
    # Tampilkan gambar di tengah
    image = Image.open(uploaded_file)
    image_resized = image.resize((224, 224))

    # CSS untuk gambar
    st.markdown(
f"""
<div style="
    display: flex;
    justify-content: center;
    align-items: center;
    margin-top: 20px;

```

```

        border: 4px solid #122525;
        border-radius: 15px;
        padding: 8px;
        width: fit-content;
        margin-left: auto;
        margin-right: auto;
    ">
        
    </div>
    """,
    unsafe_allow_html=True
)

#preprocess
processed_image = preprocess_image(image, target_size=(224,
224))

#prediksi
predictions = model.predict(processed_image)
predicted_class = class_labels[np.argmax(predictions)]
confidence = np.max(predictions)

#CSS untuk hasil prediksi dan confidence
st.markdown(
    f"""
    <div style="display: flex;
    justify-content: center;
    text-align: center;
    margin-top: 20px;
    border: 2px solid #122525;

```

```
border-radius: 10px;
padding: 10px;">
  <h3 style="font-size: 24px; color: #FFFFFF;">Prediksi:
{predicted_class}</h3>
  <h3 style="font-size: 24px; color: #FFFFFF;">Confidence:
{confidence:.2f}</h3>
</div>
""",
unsafe_allow_html=True
)
```



Lampiran 2 User Acceptance Testing

USER ACCEPTANCE TESTING

Klasifikasi Penyakit Ginjal Menggunakan Convolutional Neural Network

Nama : dr. M. Feldi Gazaly M.Ked(PD), Sp.PD, K-GH

Tempat : Rumah Sakit Universitas Sumatera Utara

Pernyataan	Sangat Tidak Setuju	Tidak Setuju	Netral	Setuju	Sangat Setuju
Sistem tidak memiliki kesulitan dalam penggunaannya					✓
Sistem klasifikasi menghasilkan diagnosa penyakit dalam waktu yang singkat					✓
Sistem klasifikasi penyakit ginjal menghasilkan diagnosa yang sesuai dengan diagnosa pakar				✓	
Sistem klasifikasi penyakit ginjal layak digunakan sesuai dengan kebutuhan medis					✓
Sistem layak dikembangkan untuk dapat mendiagnosis lebih banyak jenis penyakit				✓	

Medan, 19 November 2024

Responden



dr. M. Feldi Gazaly M.Ked(PD), Sp.PD, K-GH