

**IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* DAN *TRANSFER
LEARNING* UNTUK MENGIDENTIFIKASI ALAT RUMAH TANGGA**

SKRIPSI

T.M. REZHA TAUFIQURRAHMAN, MX

201401019



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
2024**

IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* DAN *TRANSFER LEARNING* UNTUK MENGIDENTIFIKASI ALAT RUMAH TANGGA

SKRIPSI

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

T.M. REZHA TAUFIQURRAHMAN, MX

201401019



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
2024**

PERSETUJUAN

Judul : IMPLEMENTASI CONVOLUTIONAL NEURAL
NETWORK DAN TRANSFER LEARNING
UNTUK MENGIDENTIFIKASI ALAT
RUMAH TANGGA

Kategori : SKRIPSI

Nama : T.M. REZHA TAUFIQURRAHMAN, MX

Nomor Induk Mahasiswa : 201401019

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI
INFORMASI UNIVERSITAS SUMATERA
UTARA

Medan, 1 Februari 2024

Komisi Pembimbing :

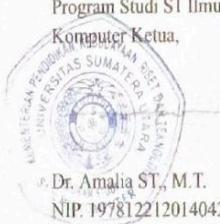
Dosen Pembimbing I

Fuzy Yustika Manik S.Kom., M.Kom
NIP. 198710152019032010

Dosen Pembimbing II

Herriyance S.T., M.Kom.
NIP. 198010242010121002

Diketahui/disetujui oleh
Program Studi S1 Ilmu
Komputer Ketua,



PERNYATAAN**IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* DAN *TRANSFER
LEARNING* UNTUK MENGIDENTIFIKASI ALAT RUMAH TANGGA****SKRIPSI**

Selain beberapa kutipan dan ringkasan yang telah dicantumkan sumbernya, saya mengakui bahwa skripsi ini adalah hasil penelitian saya sendiri.

Medan, 1 Oktober 2023



T.M. Rezha Taufiqurrahman, MX
201401019

PENGHARGAAN

Segala puji bagi Allah SWT yang memberi penulis kemampuan demi menyelesaikan skripsi ini, yang diperlukan untuk memperoleh gelar Sarjana Komputer di Program Studi S-1 Ilmu Komputer Universitas Sumatera Utara. Shalawat tidak lupa diucapkan kepada baginda Nabi Muhammad SAW, karena baginda rasulullah telah menjadi perantara kita untuk dapat memanfaatkan ilmu yang diberikan Allah SWT.

Penulis hendak menyatakan rasa terima kasih dan penghargaan kepada:

1. Bapak Dr. Muryanto Amin, S.Sos, M.Si. sebagai Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia, B.Sc., M.Sc. sebagai Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia ST., M.T. sebagai Ketua Program Studi S1 Ilmu Komputer Universitas Sumatera Utara.
4. Ibu Sri Melvani Hardi S.Kom., M.Kom sebagai Sekretaris Program Studi S1 Ilmu Komputer Universitas Sumatera Utara
5. Ibu Fuzy Yustika Manik S.Kom., M.Kom. sebagai dosen pembimbing I yang membantu penulis dalam menyelesaikan skripsi dengan memberikan bimbingan, kritik, motivasi, dan saran.
6. Bapak Herriyance S.T., M.Kom. sebagai dosen pembimbing II yang telah membantu penulis dalam menyelesaikan skripsi dengan memberikan bimbingan, kritik, motivasi, dan saran.
7. Seluruh Dosen Program Studi S-1 Ilmu Komputer yang dengan senang hati memberikan waktu dan tenaga dalam mengajar dan membimbing sehingga penulis dapat sampai kepada tahap penyusunan skripsi ini.
8. Seluruh Staf Pegawai FASILKOM-TI Universitas Sumatera Utara yang dengan baik hati memberikan bantuan kepada penulis selama periode perkuliahan sampai kepada tahap penyusunan skripsi ini.

9. Orang tua yang tidak pernah hentinya memberikan motivasi, dukungan, doa serta saran dan kasih sayang penuh kepada penulis dalam menuntaskan pendidikan.
10. Adik kandung tercinta Rafli Andriansyah yang selalu mendukung serta mendoakan penulis dalam menjalankan aktivitas kuliah hingga akhirnya menyelesaikan tugas akhir.
11. Abang-Kakak senior tercinta Altaha, Aditya Bagaskara, Alvin Adam Nasution yang selalu sabar dalam mengingatkan dan membantu penulis selama penggerjaan skripsi.
12. Sahabat penulis Atikha Azmila, dan Anggi Ester yang mendukung juga memberikan masukan positif.
13. Teman-teman perkuliahan penulis Wilson, Ariel, Imamul, Avin, Andrew, Yoga, Zelda, Syaripa, Sawaliyah, Ayu, Puan, Daniel, Abel, Ferdi, Sitti, Sonia, Zefania, Hajarani, Iqbal, Chalil, Ewaldo, Sukiya, Difanie, Tamir, John, Yunisa, Andrian yang memberikan semangat dalam menyelesaikan skripsi.
14. Teman-teman Putri, Ayu Valerie, Ani, Indah, Wulan, Widiana, Irpan, Balqis, Lula, Wanda, Juan, Alfin, Vistzal, Fazhan, Nanda yang selalu sabar dalam membantu penulis dalam pengembangan diri.
15. Adik-adik Ilmu Komputer Eunike, Elisa, Angela, Rizky, Rani, Stephen, Elin, Husein, Lorenzo, Johanes, serta adik-adik baik yang tidak dapat disebutkan namanya dan terus mendoakan penulis serta memberikan dukungan, semangat, saran dan motivasi kepada penulis.
16. Keluarga Besar IKLC USU yang tiada hentinya memberi dukungan, keceriaan dan pengalaman mengajar bersama yang berharga.
17. Dan pihak lainnya yang sangat membantu dan menyemangati dalam penyelesaian skripsi.

Harapnya rahmat Allah selalu melindungi langkah-langkah penulis serta diberikan keberkahan kepada seluruh orang yang menyalurkan motivasi, arahan, dan tindakan yang mendukung penulis demi penyelesaian skripsi. Semoga pribadi, keluarga, masyarakat, organisasi, dan negara mendapatkan manfaat dari skripsi ini.

Medan, 1 Oktober 2023

A handwritten signature in black ink, appearing to read "Rezha".

T.M. Rezha Taufiqurrahman, MX
201401019

ABSTRAK

Peralatan rumah tangga, seperti lampu, oven, dan kulkas, menjadi bagian penting dalam kehidupan sehari-hari. Namun, penggunaan yang tidak efisien dapat menyebabkan pemborosan energi dan merugikan lingkungan. Solusi untuk mengatasi masalah ini adalah memanfaatkan kecerdasan buatan (AI), khususnya algoritma *Convolutional Neural Network* (CNN) dalam implementasi *Deep Learning*. CNN, sebagai algoritma *machine learning*, memiliki keahlian dalam memproses data gambar dan mengidentifikasi pola. Pengembangan lebih lanjut menggunakan metode *Transfer Learning*, yang memanfaatkan fitur-fitur yang telah dipelajari oleh suatu jaringan untuk memecahkan masalah lain dalam domain yang sama. Proses pengembangan melibatkan pengumpulan data, *preprocessing*, pembangunan model, dan pengujian. Tujuan utama dari proyek ini adalah menciptakan model yang dapat memprediksi penggunaan energi peralatan rumah tangga dengan tingkat akurasi mencapai 97%. Implementasi model tersebut akan diintegrasikan ke dalam sebuah website dan dokumentasikan sebagai bagian dari skripsi. Dengan demikian, upaya ini bertujuan untuk mengurangi pemborosan energi dan dampak negatifnya pada lingkungan melalui pemanfaatan teknologi AI.

Kata Kunci : *Convolutional Neural Network, Deep Learning, Transfer Learning, Alat Rumah Tangga, Teknologi, Kecerdasan Buatan.*

ABSTRACT

Household appliances, such as lights, ovens, and refrigerators, play a crucial role in daily life. However, inefficient usage often leads to energy wastage and environmental harm. The solution to address this issue involves leveraging artificial intelligence (AI), specifically the Convolutional Neural Network (CNN) algorithm in Deep Learning implementations. CNN, as a machine learning algorithm, excels in processing image data and identifying patterns. Further development incorporates Transfer Learning, utilizing features learned by a network to solve other problems within the same domain. The development process involves data collection, preprocessing, model construction, and testing. The primary goal of this project is to create a model capable of predicting energy consumption by household appliances with an accuracy level of 97%. The implementation of this model will be integrated into a website and documented as part of a thesis. Thus, this effort aims to reduce energy wastage and its adverse environmental impacts through the application of AI technology.

Keywords : Convolutional Neural Networks, Deep Learning, Transfer Learning, Household Appliances, Technology, Artificial Intelligence.

DAFTAR ISI

PERSETUJUAN.....	iii
PERNYATAAN.....	iv
PENGHARGAAN.....	v
ABSTRAK.....	viii
ABSTRACT.....	ix
DAFTAR ISI.....	x
DAFTAR TABEL.....	xii
DAFTAR GAMBAR.....	xiii
BAB I PENDAHULUAN.....	15
1.1. Latar Belakang.....	15
1.2. Rumusan Masalah.....	17
1.3. Batasan Masalah.....	17
1.4. Tujuan Penelitian.....	18
1.5. Manfaat Penelitian.....	18
1.6. Metodologi Penelitian.....	18
1.7. Sistematika Penulisan.....	21
BAB II LANDASAN TEORI.....	22
2.1. Machine Learning.....	22
2.2. Convolutional Neural Network (CNN).....	22
2.2.1. Pengolahan Gambar Menggunakan CNN.....	23
2.3. Transfer Learning.....	25
2.4. Application Programming Interface (API).....	26
2.5. Penggunaan Teknologi dalam Estimasi Konsumsi Energi.....	27
2.6. Penelitian Relevan.....	28
BAB III ANALISIS DAN PERANCANGAN SISTEM.....	30
3.1. Analisis Sistem.....	30
3.1.1 Analisis Masalah.....	30
3.1.2 Analisis Kebutuhan.....	33
3.1.2.1 Kebutuhan Fungsional.....	33
3.1.2.1 Kebutuhan Non-Fungsional.....	33
3.1.3 Diagram Umum Sistem.....	33
3.2. Pemodelan Sistem.....	35
3.2.1 Use Case Diagram.....	35
3.2.2 Activity Diagram.....	36
3.2.2 Sequence Diagram.....	38

3.3. Flowchart.....	39
3.3.1 Flowchart Sistem.....	39
3.3.2 Diagram Arsitektur Algoritma Convolutional Neural Network (CNN).....	40
3.3.3 Diagram Arsitektur Transfer Learning.....	41
3.4. Perancangan Interface.....	41
3.4.1 Halaman Prediksi dengan Gambar.....	42
3.4.2 Halaman Prediksi dengan Form.....	43
BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM.....	44
4.1. Implementasi.....	44
4.1.1 Landing Page.....	44
4.1.2 Halaman Prediksi dengan Gambar.....	45
4.1.3 Halaman Hasil Prediksi dengan Gambar.....	46
4.1.4 Halaman Prediksi dengan Form.....	47
4.1.5 Halaman Hasil Prediksi dengan Form.....	47
4.2. Pengujian Sistem.....	48
4.3. Hasil Pengujian Sistem.....	61
4.4. Faktor Pengaruh Algoritma CNN.....	67
4.5. Faktor Pengaruh Arsitektur Transfer Learning.....	68
BAB V PENUTUP.....	69
5.1. Kesimpulan.....	69
5.2. Saran.....	70
DAFTAR PUSTAKA.....	71
LAMPIRAN.....	74

DAFTAR TABEL

Tabel 4.1 Daftar Nama dan Tipe Alat	48
Tabel 4.2 Daftar Hasil Percobaan	61

DAFTAR GAMBAR

Gambar 1.1 Arsitektur Sistem	19
Gambar 2.1 Ilustrasi Machine Learning	22
Gambar 2.2 Ilustrasi CNN	23
Gambar 2.3 Ilustrasi Pengolahan Gambar	24
Gambar 2.4 Ilustrasi API	26
Gambar 3.1 Diagram Umum Sistem	34
Gambar 3.2 Use Case Diagram	35
Gambar 3.3 Activity Diagram	37
Gambar 3.4 Sequence Diagram	38
Gambar 3.5 Flowchart Sistem	39
Gambar 3.6 Diagram Arsitektur Algoritma CNN	40
Gambar 3.7 Diagram Arsitektur Transfer Learning	41
Gambar 3.8 Halaman Prediksi Dengan Gambar	42
Gambar 3.9 Halaman Prediksi Dengan Form	43
Gambar 4.1 Landing Page Sistem	44
Gambar 4.2 Halaman Prediksi dengan Gambar	45
Gambar 4.3 Halaman Hasil Prediksi dengan Gambar	46
Gambar 4.4 Halaman Prediksi dengan Form	47
Gambar 4.5 Halaman Hasil Prediksi dengan Form	47
Gambar 4.6 Import Library	50
Gambar 4.7 Persiapan Training Model	51
Gambar 4.8 Pemberian Label dan Load Pre-trained Model	51
Gambar 4.9 Proses Compile Model dan Melatih Model	52
Gambar 4.10 Proses Pelatihan Model dengan 50 Iterasi	53
Gambar 4.11 Menampilkan data hasil pelatihan dengan library matplotlib	54
Gambar 4.12 Grafik Training dan Validation Accuracy	55
Gambar 4.13 Grafik Training dan Validation Loss	56
Gambar 4.14 Penyimpanan model dengan ekstensi h5	56
Gambar 4.15 Pemberian label kelas dan pengujian model dengan data testing	57
Gambar 4.16 Melakukan testing terhadap model	58
Gambar 4.17 Tampilan keluaran model	59
Gambar 4.18 Percobaan pada Website	60
Gambar 4.19 Diagram Arsitektur CNN dengan Transfer Learning	63

Gambar 4.20 Hasil Model CNN dilengkapi Transfer Learning	64
Gambar 4.21 Diagram Arsitektur CNN dengan Transfer Learning	64
Gambar 4.22 Hasil Model CNN tanpa Transfer Learning	65
Gambar 4.23 Grafik hasil Model CNN Tanpa Transfer Learning	65
Gambar 4.24 Hasil Pengujian Model CNN Tanpa Transfer Learning	66

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dalam periode sekarang, peralatan rumah tangga merupakan inti dari konsumsi energi di negara-negara maju maupun berkembang. Peningkatan populasi dan kesejahteraan menunjukkan bahwa jumlah orang yang menggunakan peralatan rumah tangga semakin meningkat di masa depan. Perkembangan ini membawa dampak signifikan terhadap konsumsi energi, karena berbagai kebutuhan mulai dari yang dasar hingga mewah harus dipenuhi. (Hischier, R., et al., 2020).

Semakin tingginya permintaan akan peralatan rumah tangga, dan dengan demikian produksi dan konsumsinya, menyebabkan bidang ini dianggap sebagai salah satu area yang relevan untuk campur tangan guna memastikan produksi dan konsumsi yang berkelanjutan (Hischier, R., et al., 2020). Dalam beberapa dekade mendatang, diekspektasikan hal serupa terjadi di Indonesia seiring pertumbuhan ekonomi negara tersebut. Pertumbuhan ini kemungkinan akan melibatkan penggunaan bahan bakar fosil yang terus berlanjut, dan akibatnya dampak negatif terhadap lingkungan semakin meningkat.

Dengan kebutuhan listrik yang terus meningkat, menjadikan masyarakat sangat bergantung kepada listrik. Hal ini terjadi bukan hanya pada sektor rumah tangga tetapi juga untuk kegiatan ekonomi di sektor industri, kantor pemerintahan, dan bisnis. Hal ini tentu menjadi catatan untuk pengembangan dan penggunaan energi kedepannya (Pangestu, 2022).

Penelitian ini bertujuan untuk mengatasi masalah membantu konsumen menemukan alternatif untuk perangkat rumah tangga. Dengan memberikan informasi yang diperlukan untuk dipertimbangkan, seperti konsumsi energi, dampak lingkungan dari produksi, konsumsi, dan pembuangan suatu produk, serta memberikan saran-saran alternatif kepada mereka. Dengan begitu, konsumen lebih menyadari dampak konsumsi energi berlebihan yang mereka lakukan.

Convolutional Neural Network (CNN) dalam hal ini akan digunakan sebagai alat untuk mengidentifikasi penggunaan energi dari peralatan rumah tangga nantinya. Perlu diketahui bahwa CNN adalah sebuah algoritma *Machine Learning* yang handal dalam hal pengolahan data dalam bentuk gambar dan mencari pola dari gambar-gambar yang diberikan. Dengan mengidentifikasi pola atau fitur yang terdapat pada sebuah gambar, CNN dapat membantu dalam identifikasi penggunaan energi yang berlebihan.

Transfer Learning adalah Teknik yang memakai fitur-fitur yang telah dipelajari oleh sebuah *network* dalam memecahkan masalah tertentu untuk memecahkan masalah lain namun masih berada pada domain yang sama. *Transfer Learning* merupakan sebuah metode yang dapat diterapkan pada arsitektur CNN. *Transfer Learning* membuat model mempunyai pengetahuan yang lebih luas karena dapat menggunakan informasi yang telah tersedia dan mengembangkannya (Kandel, 2020).

Arsitektur CNN yang digunakan adalah *DenseNet 121*. *DenseNet 121* adalah sebuah arsitektur jaringan saraf konvolusi yang memiliki hubungan yang padat atau Dense antara setiap lapisan. *DenseNet 121* diimplementasikan menggunakan *Transfer Learning* dari *Tensorflow* untuk pembuatan model *Machine Learning*. *DenseNet 121* merupakan cara agar model dilatih secara lebih akurat dan model akan memiliki hubungan yang lebih pendek antar lapisan. Dengan *DenseNet 121* model dapat melakukan ekstraksi fitur dan pola yang lebih akurat demi menghasilkan prediksi yang lebih akurat pula (Huang et al., 2018).

Proses pengolahan gambar dengan CNN akan dimulai dengan pengguna memasukkan gambar yang akan dilakukan pengecekan. Kemudian gambar akan dikirim melalui API kepada model untuk dilakukan percobaan dan prediksi. CNN akan melakukan ekstraksi fitur terhadap gambar yang dilakukan dan akan memprediksi alat rumah tangga apa yang terdapat pada gambar berdasarkan fitur-fitur yang telah diekstrak tadi menggunakan metode Transfer Learning. Model kemudian akan mengembalikan hasil yang sesuai beserta informasi yang relevan terhadap alat rumah tangga yang diprediksi sebelumnya. Hasil dan

informasi akan dikirim melalui API kembali dan akan ditampilkan kepada pengguna tentang alat yang mereka gunakan.

1.2. Rumusan Masalah

Penggunaan *Convolutional Neural Network* (CNN) dan *Transfer Learning* dalam implementasi untuk mengidentifikasi alat rumah tangga bertujuan untuk membantu membatasi penggunaan energi berlebihan. Dengan memanfaatkan teknologi ini, diharapkan dapat dikembangkan sebuah sistem yang mampu secara otomatis mengenali dan memonitor penggunaan energi pada peralatan rumah tangga. Hal ini diharapkan dapat menjadi langkah proaktif dalam meningkatkan efisiensi energi rumah tangga dan mendukung upaya untuk mengurangi dampak lingkungan.

1.3. Batasan Masalah

Batasan masalah penelitian, yakni:

- a. Khusus membahas peralatan-peralatan rumah tangga seperti *Air Conditioner* (AC), pengering rambut, setrika, lampu, laptop, oven, kulkas, penanak nasi, televisi, penyedot debu, dan mesin cuci.
- b. Tidak akan membahas peralatan rumah tangga dengan merek atau *brand* tertentu serta tipe-tipe khusus yang mungkin dimiliki oleh sebuah alat rumah tangga.
- c. *Framework* yang digunakan dalam membuat API adalah *framework Flask*.
- d. Bahasa yang digunakan untuk membangun program dan model adalah bahasa pemrograman *Python*.
- e. Tidak akan membahas implementasi atau pengendalian langsung pada alat-alat rumah tangga pendingin ruangan, pengering rambut, setrika, lampu, laptop, oven, kulkas, penanak nasi, televisi, penyedot debu, dan mesin cuci.
- f. Khusus membahas pengembangan model estimasi konsumsi energi berbasis CNN dan *Transfer Learning* berdasarkan alat-alat rumah tangga

seperti *Air Conditioner* (AC), pengering rambut, setrika, lampu, laptop, oven, kulkas, penanak nasi, televisi, penyedot debu, dan mesin cuci.

- g. Arsitektur khusus yang digunakan untuk CNN adalah *DenseNet 121*.
- h. Data API khusus digunakan untuk memberikan informasi terhadap penggunaan alat-alat tersebut.
- i. Satuan energi khusus menggunakan kWh.

1.4. Tujuan Penelitian

Menurut rumusan masalah, dengan *Convolutional Neural Network*, *Transfer Learning*, dan API, dapat digunakan untuk membuat model yang berfungsi untuk melakukan estimasi konsumsi energi pada alat-alat rumah tangga.

1.5. Manfaat Penelitian

Penelitian ini harapnya akan menghasilkan keuntungan bagi yang menggunakan seperti mengetahui konsumsi energi yang digunakan pada kehidupan sehari-hari, meningkatkan kewaspadaan terhadap dampak konsumsi energi berlebihan, serta menjaga lingkungan sekitar dengan mengetahui estimasi energi pada alat-alat rumah tangga yang digunakan pada kehidupan sehari-hari.

1.6. Metodologi Penelitian

Dalam penelitian, metode berikut akan dilakukan:

1. Identifikasi Tujuan Penelitian

Pada bagian ini studi akan dimulai dengan menilai efektivitas penggunaan *Convolutional Neural Network* (CNN) dan *Transfer Learning* dalam pengelolaan energi pada alat-alat kebutuhan rumah tangga.

2. Kajian Pustaka

Selanjutnya akan dilakukan pengkajian pustaka demi pengumpulan informasi tentang pengelolaan energi pada alat-alat rumah tangga dengan bantuan CNN dan *Transfer Learning*.

3. Implementasi

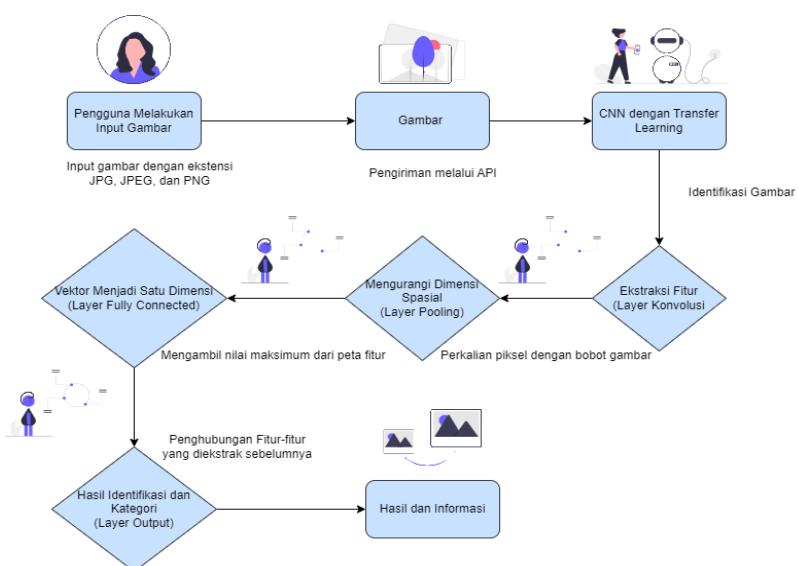
Pada tahap ini akan dilakukan pengembangan serta pembersihan data yang akan digunakan untuk pembuatan model. Pada tahap ini juga akan dimulai pengembangan dari sisi API menggunakan *Framework Flask* dengan bahasa *Python*.

4. Pengujian

Pada bagian ini, dilaksanakan uji coba untuk memastikan bahwa model *Machine Learning* dengan *Convolutional Neural Network* (CNN) untuk pengelolaan energi pada peralatan rumah tangga berfungsi sesuai dengan apa yang diharapkan.

5. Arsitektur Umum

Berikut adalah arsitektur sederhana yang akan digunakan:



Gambar 1.1 Arsitektur Sistem

Pada gambar 1.1 dapat dilihat bagaimana arsitektur sistem bekerja. Dimulai dari pengguna melakukan *input* gambar kedalam sistem dan kemudian gambar diidentifikasi oleh model dan diproses melalui 3 lapisan yaitu lapisan konvolusi dimana gambar akan dilakukan ekstraksi fitur, kemudian pada lapisan *pooling* dilakukan pengurangan dimensi spasial untuk fitur-fitur yang tadi di ekstrak. Kemudian dijadikan *input* agar dapat diterima oleh lapisan *fully connected* dengan bentuk vektor satu dimensi. Akhirnya, didapatkan hasil pada lapisan keluaran dan ditampilkan hasil proses gambar tersebut.

6. Dokumentasi

Berikutnya, seluruh proses penelitian dimulai dari awal hingga tahap terakhir akan dicatat sebagai skripsi.

1.7. Sistematika Penulisan

Skripsi ini memiliki lima bab sistematika penulisan, yaitu:

BAB I PENDAHULUAN

Semua yang berkaitan tentang konteks masalah termasuk kumpulan masalah, tujuan penelitian, limitasi masalah, tujuan dan fungsi serta metodologi , juga proses penulisan dijabarkan pada bagian berikut.

BAB II LANDASAN TEORI

Analisis teoritis mengenai hal *Convolutional Neural Network* dan *Transfer Learning* akan dibicarakan disini.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Melaksanakan pengamatan yang berkaitan dan sistem yang diciptakan dibahas disini. Kemudian dilanjutkan dengan perancangan sistem dengan menggunakan Algoritma CNN dan Metode *Transfer Learning*.

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Dalam bab berikut mencakup pelaksanaan dan *testing* sistem dimana didasarkan pada langkah-langkah analisis perancangan dan sistem yang telah disebutkan diatas.

BAB V PENUTUP

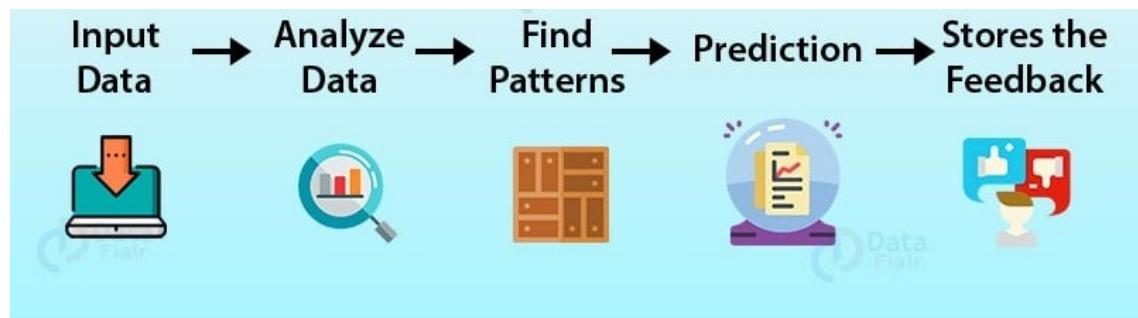
Bab berikut akan mencakup hasil dan rekomendasi untuk penelitian selanjutnya.

BAB II

LANDASAN TEORI

2.1. Machine Learning

Machine Learning adalah subdisiplin dari ilmu komputer yang memiliki keterkaitan dengan pengembangan algoritma dan model dimana memungkinkan sebuah komputer mempunyai kemampuan dalam mempelajari *pattern*, membuat prediksi, dan menarik keputusan berdasarkan *pattern* yang diberikan. *Machine Learning* secara luas bertujuan memungkinkan komputer untuk “belajar” tanpa harus di program langsung. Komputer meningkatkan kinerjanya untuk menyelesaikan tugas yang ada berdasarkan “pengalaman” yakni data yang diberikan kepadanya (Bi et al., 2019).



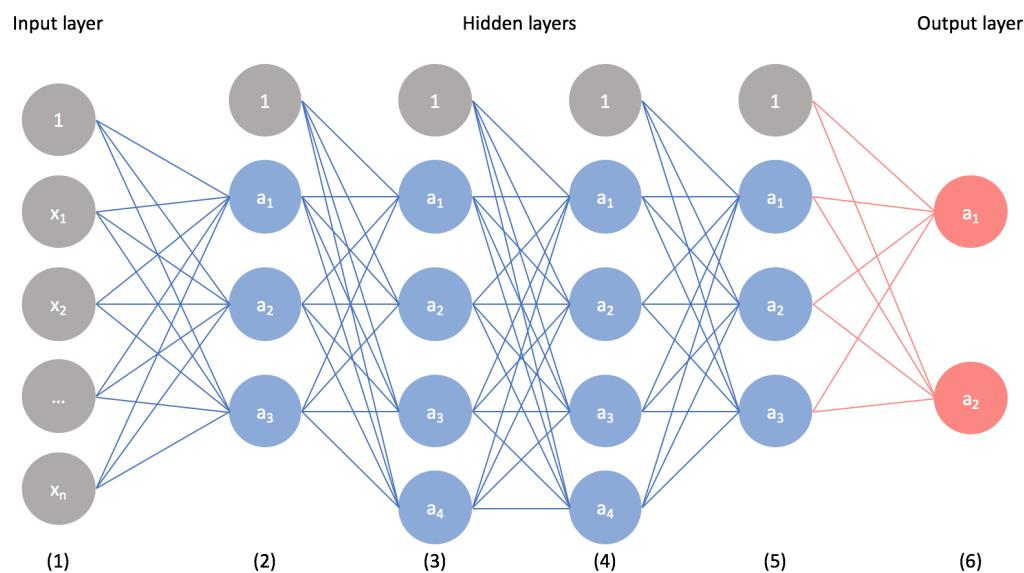
Gambar 2.1 Ilustrasi Machine Learning

Pada gambar 2.1 dapat dilihat ilustrasi *machine learning*. Dijelaskan bagaimana data dimasukkan kemudian model akan menganalisis data dan menemukan pola dari data tersebut hingga dapat menarik sebuah kesimpulan atau prediksi dan menerima umpan balik hingga model dapat memprediksi dengan baik.

2.2. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) disebutkan sebuah arsitektur *neural network* tiruan yang dirancang demi mengolah data terstruktur seperti gambar ataupun data lainnya yang berdimensi spasial. CNN menggunakan operasi konvolusional untuk mengambil atau mengekstrak fitur-fitur yang ada pada data. Dengan begitu, komputer dapat mengenal pola yang lebih kompleks

dan mempertajam hasil yang didapat pada model. CNN terdiri atas beberapa lapisan seperti normalisasi, aktivasi, dan *pooling* yang digunakan agar komputer dapat menerima fitur-fitur dari data yang diberikan. CNN dapat melakukan transformasi data menjadi representasi lebih tinggi untuk mengerjakan tugas-tugas yang kompleks dan rumit (Messina et al., 2021).



Gambar 2.2 Ilustrasi CNN

Pada gambar 2.2 dapat dijelaskan bagaimana algoritma CNN bekerja. Dimulai dengan memasukkan *input* pada lapisan *input* dan kemudian diproses oleh lapisan-lapisan tersembunyi dimana data akan diekstrak, dianalisis dan sebagainya. Terakhir setelah semua analisa dilakukan, ditarik sebuah kesimpulan yaitu prediksi terhadap data tersebut pada bagian lapisan keluaran.

2.2.1. Pengolahan Gambar Menggunakan CNN

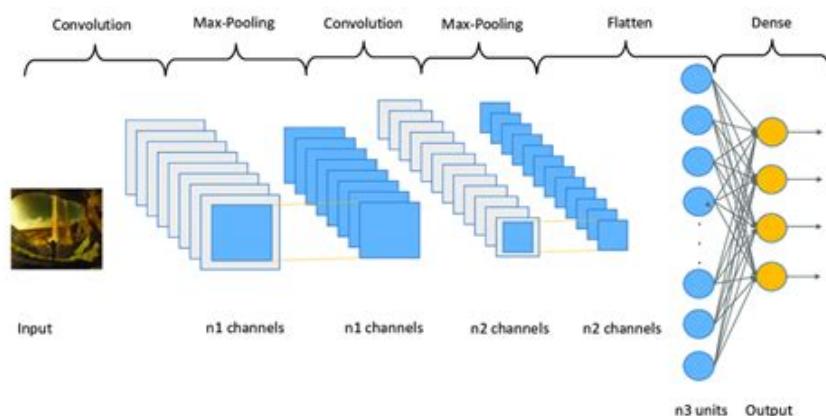
Convolutional Neural Network tentu saja dapat digunakan dalam pengolahan citra. Secara sederhana, CNN melakukan ekstraksi terhadap fitur-fitur yang ada dari sebuah gambar. Proses ini disebut juga dengan operasi konvolusi. Hal ini dilakukan dengan melibatkan kernel dan *filter*

untuk mendeteksi pola dan fitur-fitur yang ada pada gambar. Setiap *filter* akan digeser secara berulang kali untuk menghasilkan sebuah peta fitur.

CNN juga menggunakan lapisan *pooling* untuk mengurangi dimensi dari peta fitur yang dihasilkan. Hal ini dilakukan agar parameter yang digunakan tidak terlalu banyak agar dapat mempercepat proses pelatihan dan menghindari *overfitting*.

Ketika proses diatas telah selesai, hasil akan dimasukkan ke dalam lapisan *fully connected* yang bertugas untuk mengklasifikasikan gambar ke dalam kategori yang tersedia.

Dengan penjelasan diatas, dapat disimpulkan bahwa CNN dapat membantu dalam proses pengolahan gambar serta konsumsi energi. Peralatan rumah tangga tentu memiliki citra yang memiliki fitur-fitur tersendiri yang dapat dideteksi oleh CNN. Dengan begitu, pengguna dapat mengetahui informasi lebih lanjut mengenai konsumsi energi mereka masing-masing.



Gambar 2.3 Ilustrasi Pengolahan Gambar

Pada gambar 2.3 dapat dilihat bagaimana CNN melakukan pengolahan gambar dengan algoritma nya. Dimulai dengan melakukan ekstraksi fitur gambar kemudian melakukan pengurangan dimensi spasial pada lapisan *pooling* dan diterapkan teknik *flatten* untuk mengubah fitur menjadi vektor satu dimensi

agar dapat diterima oleh lapisan *fully connected*. Hingga dikeluarkan hasil pada lapisan keluaran.

2.3. Transfer Learning

Transfer Learning merupakan pendekatan pada pembelajaran mesin dimana model yang telah dilatih akan digunakan untuk memperbaiki atau menyelesaikan masalah pada masalah lainnya namun pada domain yang sama. Dalam *Transfer Learning*, model yang telah dilatih di awal atau disebut model sumber, digunakan sebagai *starting point* untuk melatih model baru atau disebut model target (Zhuang et al., 2020).

Salah satu arsitektur populer untuk CNN yang digunakan untuk *transfer learning* adalah *DenseNet 121*. Dalam *DenseNet 121* setiap lapisan menerima *input* dari seluruh lapisan sebelumnya. Hal ini menyatakan bahwa setiap lapisan terhubung satu sama lain. Dengan hubungan ini, aliran informasi dalam *network* akan menjadi lebih kuat dan akurat (Huang et al., 2018).

Keuntungan dari *DenseNet 121* adalah kemampuannya dalam mengatasi masalah gradien yang melemah (*vanishing gradient*). *DenseNet 121* dapat mempercepat model untuk dilatih, mengurangi jumlah parameter yang perlu diestimasi, dan meningkatkan kinerja serta akurasi dari model dalam pengolahan gambar.

DenseNet 121 memiliki total 121 lapisan, termasuk lapisan konvolusi, lapisan *pooling*, dan lapisan klasifikasi. Dalam *transfer learning*, *DenseNet 121* sering digunakan sebagai model sumber yang telah dilatih dahulu pada dataset yang sangat besar.. Kemudian, bobot model tersebut dilakukan *fine-tuning* untuk tugas klasifikasi khusus dengan dataset yang lebih kecil (Huang et al., 2018).

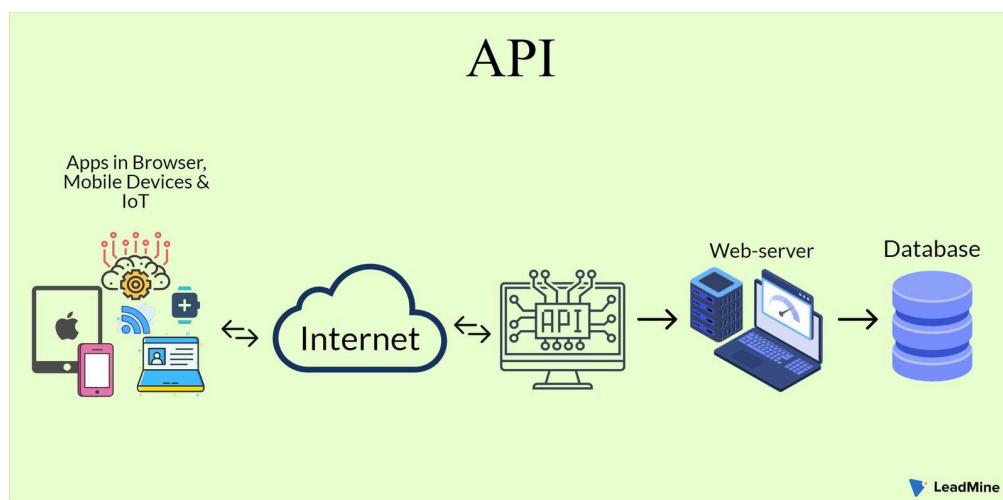
2.4. Application Programming Interface (API)

API adalah koleksi protokol serta perintah dimana memungkinkan *software* mampu saling berinteraksi satu sama lain. API bertindak sebagai penghubung untuk mengirim data antara dua atau lebih aplikasi.

Dengan adanya API, para pengembang tidak perlu repot untuk mempelajari aplikasi lain secara terperinci. Pengembang dapat langsung melakukan integrasi fungsionalitas dari aplikasi lain ke dalam aplikasi yang hendak dibuat.

API dapat terdiri atas beberapa instruksi, protokol komunikasi, atau juga beberapa fungsi dari sebuah *library*. API dapat digunakan untuk berbagai tujuan seperti mengakses halaman *website*, dan integrasi dengan *platform* lainnya.

API dapat digunakan sebagai sarana untuk mendistribusikan *Open Data*. Seperti contoh, apabila seseorang ingin memanipulasi data tanpa harus mengubah halaman *website* yang dimiliki maka dapat digunakan API. Hal ini seakan-akan pengembang menitipkan sebagian potongan situs *website* pada API. Contoh lainnya seperti konversi suhu dari *Fahrenheit* ke *Celcius*. Untuk menampilkan data seperti itu tidak diperlukan manipulasi halaman *website* pengembang. Hanya perlu menyisipkan potongan kecil dari *website* lain untuk meminta data seperti itu. (Sugimoto, G. 2019).



Gambar 2.4 Ilustrasi API

Gambar 2.4 menjelaskan bagaimana API bekerja. Dimulai dari aplikasi yang terhubung ke internet ingin meminta data ke bagian server atau basis data aplikasi. Demi menjaga simplicitas API melakukan permintaan langsung kebagian server dan basis data tanpa perlu memahami komponen program basis data dan server aplikasi. Kemudian server dan basis data memberikan data yang diminta oleh aplikasi dan mengirim data tersebut dalam bentuk *response* kepada aplikasi melalui API. API merupakan perantara antara aplikasi dan server dalam ilustrasi tersebut.

2.5. Penggunaan Teknologi dalam Estimasi Konsumsi Energi

Penggunaan teknologi dalam estimasi konsumsi energi merupakan cara untuk mengoptimalkan konsumsi energi dan pengelolaannya. Teknologi ini mencakup segala hal baik dalam penghematan energi hingga sistem energi terbarukan.

Seperti contoh, Sistem penggalian *Bitcoin* memakan energi yang cukup banyak. Bahkan dapat dikatakan setara dengan penggunaan energi listrik oleh Pemerintah Negara Jerman. Maka untuk mengatasi masalah tersebut diciptakan sebuah metode untuk mengatasi konsumsi berlebihan tersebut menggunakan metode “*Proof-of-Work (PoW)*” yaitu metode menggunakan teknologi *blockchain* dan kriptografi. (Sedlmeir et al., 2019)

Metode ini dianggap efektif terhadap konsumsi energi untuk penggalian *bitcoin* dikarenakan dapat menentukan dengan pasti *computing power* yang digunakan menggunakan *hash rate*.

Batas atas untuk konsumsi listrik yang disebabkan oleh penambangan juga dapat diestimasi, selama diasumsikan bahwa semua peserta bertindak secara rasional, yang berarti mereka bertujuan untuk memperoleh keuntungan dengan berpartisipasi dalam penambangan. Hal ini mungkin tidak benar untuk semua peserta, tetapi sebagian besar dari kekuatan komputasi untuk *Bitcoin* dan *cryptocurrency PoW* lainnya yang relevan disediakan oleh perusahaan atau kelompok yang mengkhususkan diri dalam penambangan (Romiti et al., 2019).

2.6. Penelitian Relevan

Selanjutnya adalah penelitian yang telah dilakukan sebelumnya yang berkaitan, yakni:

1. Pada jurnal berjudul Sistem Baru Berbasis *Convolutional Neural Network* untuk Aplikasi NILM (Ciancetta, et al., 2021). Penelitian yang dilakukan adalah membangun sebuah model yang disebut NILM yang berbasis *Convolutional Neural Network* untuk mengidentifikasi konsumsi energi yang tersambung ke rumah. Identifikasi konsumsi daya peralatan individu sangat penting, karena memungkinkan adanya perbaikan yang dapat mengurangi konsumsi daya peralatan tersebut. Sistem yang diusulkan mampu mengekstrak permintaan energi dari setiap perangkat individu menggunakan teknik pemantauan beban non intrusif (NILM). Algoritma yang diusulkan memungkinkan deteksi dan klasifikasi peristiwa secara simultan tanpa perlu melakukan pemrosesan ganda.
2. Pada jurnal berjudul Jaringan Neural LSTM Hibrida untuk Peramalan Konsumsi Energi Rumah Tangga Individual (Yan, et al., 2019). Pada penelitian ini digunakan sebuah metode untuk memprediksi penggunaan energi di masa depan berbasis LSTM dengan menggunakan *Historical Data*. Dalam penelitian ini, model *deep learning* hibrida diusulkan yang menggabungkan jaringan saraf *ensemble* LSTM dengan teknik SWT. SWT mengurangi volatilitas dan meningkatkan dimensi data, yang berpotensi membantu meningkatkan akurasi LSTM. Selain itu, jaringan saraf ensemble LSTM juga meningkatkan performa prediksi metode yang diusulkan.
3. Pada jurnal berjudul Meningkatkan Peramalan Konsumsi Energi Listrik Menggunakan CNN dan Bi-LSTM (Le, et al., 2019) Penelitian ini melakukan percobaan menggunakan CNN dan Bi-LSTM untuk mengembangkan model yang dapat digunakan untuk memprediksi penggunaan energi listrik. Dua CNN dalam modul pertama mengekstrak informasi penting dari beberapa variabel dalam dataset IHEPC.

Kemudian, modul dengan dua lapisan Bi-LSTM menggunakan informasi tersebut serta tren deret waktu dalam dua arah, yaitu ke depan dan ke belakang, untuk melakukan prediksi. Nilai yang diperoleh dalam modul Bi-LSTM akan dilewatkan ke modul terakhir yang terdiri dari dua lapisan *fully connected* untuk akhirnya memprediksi konsumsi energi listrik di masa depan.

4. Pada jurnal berjudul DB-Net: Model Peramalan Multi-Langkah Berbasis *Dilated CNN* yang Baru untuk Konsumsi Daya dalam Sistem Energi Lokal Terintegrasi (Khan, et al., 2021) Penelitian ini menjelaskan bagaimana CNN dapat melakukan manajemen energi listrik menggunakan *Integrated Local Energy Systems* yang diimplementasikan melalui *Machine Learning*. Pendekatan yang diusulkan memungkinkan pengendalian energi daya yang efisien dalam ILES antara konsumen dan pemasok saat digunakan untuk ECP jangka panjang dan pendek. Tahap pertama menggabungkan prosedur akuisisi data dan penyempurnaan ke dalam modul pra-pemrosesan di mana tujuan utamanya adalah untuk mengoptimalkan data yang terkumpul dan mengatasi *outliers*. Pada tahap berikutnya, data yang telah disempurnakan dilewatkan ke lapisan DCNN untuk encoding fitur diikuti oleh lapisan Bi-LSTM untuk mempelajari pola sekuensial tersembunyi dan mendekode peta fitur. Pada tahap terakhir, model DB-Net memprediksi *multi-step power consumption* (PC), termasuk output per jam, harian, mingguan, dan bulanan. Pendekatan yang diusulkan mencapai performa prediksi yang lebih baik daripada metode yang ada, dengan demikian mengkonfirmasi keefektifannya.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Sistem

Analisis sistem termasuk bagian dalam penelitian bertujuan demi menguraikan komponen-komponen yang diperlukan untuk dapat bekerja di dalam sistem. Analisis sistem dapat dibagi menjadi dua jenis yakni analisis masalah dan analisis kebutuhan. Analisis masalah berguna demi identifikasi penyebab serta dampak dari sebuah masalah. Sedangkan analisis kebutuhan dibutuhkan agar menguraikan data dan tahap yang diperlukan dalam menciptakan sistem.

3.1.1 Analisis Masalah

Dalam penelitian, analisis masalah mempunyai manfaat demi menelusuri penyebab atas masalah kemudian dilakukan pengkajian lebih dalam pada konflik yang akan ditangani untuk menghasilkan sistem yang efektif. Permasalahan yang dianalisis ialah bagaimana algoritma CNN dapat membantu pengguna dalam mengurangi penggunaan energi berlebih pada alat rumah tangga mereka serta memberikan informasi lebih lanjut mengenai alat tersebut.

Untuk mempermudah analisis masalah, sebuah metode yang dapat digunakan adalah metode *5-Whys*. Metode *5-Whys* merupakan pendekatan yang digunakan dalam analisis masalah untuk menggali akar penyebab masalah dengan bertanya "Mengapa?" secara berulang. Metode ini bertujuan untuk menemukan penyebab mendasar dari suatu masalah dengan mengidentifikasi dan mengeksplorasi berbagai faktor yang berkontribusi terhadap masalah tersebut.

Berikut adalah beberapa pertanyaan yang digunakan untuk mengeksplorasi lebih lanjut masalah penelitian diatas:

1. Mengapa identifikasi energi alat rumah tangga penting?

Jawab: Peralatan rumah tangga telah menjadi inti dari konsumsi energi di negara-negara maju maupun berkembang. Peningkatan populasi dan kesejahteraan menunjukkan bahwa jumlah orang yang menggunakan peralatan rumah tangga semakin meningkat di masa depan. Semakin tingginya permintaan akan peralatan rumah tangga, dan dengan demikian produksi dan konsumsinya, menyebabkan bidang ini dianggap sebagai salah satu area yang relevan untuk campur tangan guna memastikan produksi dan konsumsi yang berkelanjutan.

2. Mengapa *Convolutional Neural Network* (CNN) dipilih sebagai metode untuk identifikasi energi alat rumah tangga?

Jawab: CNN adalah sebuah algoritma *Machine Learning* yang handal dalam hal pengolahan data dalam bentuk gambar dan mencari pola dari gambar-gambar yang diberikan. Dengan mengidentifikasi pola atau fitur yang terdapat pada sebuah gambar, CNN dapat membantu dalam identifikasi penggunaan energi yang berlebihan. CNN memiliki proses ekstraksi gambar yang baik sehingga dapat melakukan pengolahan fitur-fitur yang terdapat pada sebuah gambar dan sesuai dengan masalah yang ingin diselesaikan yakni identifikasi energi pada alat rumah tangga.

3. Mengapa *Transfer Learning* diintegrasikan dalam studi kasus ini untuk estimasi konsumsi energi berbasis API?

Jawab: *Transfer Learning* merupakan sebuah arsitektur yang dapat diterapkan di dalam algoritma CNN. *Transfer Learning* merupakan sebuah teknik yang digunakan untuk memperluas pengetahuan target model dengan memanfaatkan model yang telah dilatih sebelumnya (*pre-train*). *Transfer Learning* juga

membantu target model untuk menghindari *overfitting*, dan mempercepat waktu latih model. Dalam kasus ini, *Transfer Learning* digunakan untuk mempercepat pelatihan model estimasi konsumsi energi berbasis API dan juga mempertajam akurasi model agar dapat memprediksi dengan baik penggunaan energi oleh alat rumah tangga.

4. Mengapa API digunakan sebagai metode dalam penelitian estimasi konsumsi energi?

Jawab: API merupakan serangkaian protokol yang dapat digunakan untuk menghubungkan dua buah atau lebih aplikasi. API juga berperan untuk mengirimkan *request* dan *response* dari pengguna dan *server*. Dengan API, program dapat dengan mudah saling berinteraksi tanpa harus mengerti bagaimana aplikasi lainnya dijalankan. API dapat mengurangi beban pada pengembang serta aplikasi. Karena pengembang hanya perlu fokus pada API daripada harus mempelajari arsitektur secara detail pada kedua buah atau lebih aplikasi tersebut. Estimasi konsumsi energi akan menggunakan API untuk mempermudah interaksi antara *Back-end* dengan *front-end*.

5. Mengapa diperlukan studi kasus untuk menguji pendekatan ini dalam situasi nyata?

Jawab: Dengan adanya studi kasus, akan memberikan pemahaman lebih mendalam tentang CNN dan *Transfer Learning*. Dengan situasi nyata, pengguna diharapkan mampu mengidentifikasi dan menghemat penggunaan alat rumah tangga mereka dan meningkatkan *awareness* terhadap penggunaan alat rumah tangga berlebihan.

3.1.2 Analisis Kebutuhan

Analisis kebutuhan termasuk langkah penting demi mengenali data dan prosedur yang diperlukan dalam merancang sistem. Dalam proses perancangan, analisis kebutuhan dapat mencakup persyaratan fungsional dan non-fungsional yang perlu dipenuhi sehingga sistem yang dikembangkan dapat mencapai tujuannya.

3.1.2.1 Kebutuhan Fungsional

Kebutuhan Fungsional termasuk pengertian dari tahapan yang akan dikerjakan sisi sistem demi memenuhi tujuannya. Berikut merupakan Kebutuhan Fungsional yang digunakan untuk penelitian dan sistem:

1. Sistem dapat membaca *input* gambar dari pengguna.
2. Sistem dapat mengenali jenis alat rumah tangga ketika dilakukan proses prediksi oleh model.
3. Sistem dapat melakukan estimasi penggunaan energi yang dihasilkan oleh alat rumah tangga tersebut.
4. Sistem dapat memberikan hasil lainnya yang berkaitan dengan alat rumah tangga tersebut untuk pemahaman pengguna lebih lanjut.

3.1.2.2 Kebutuhan Non-Fungsional

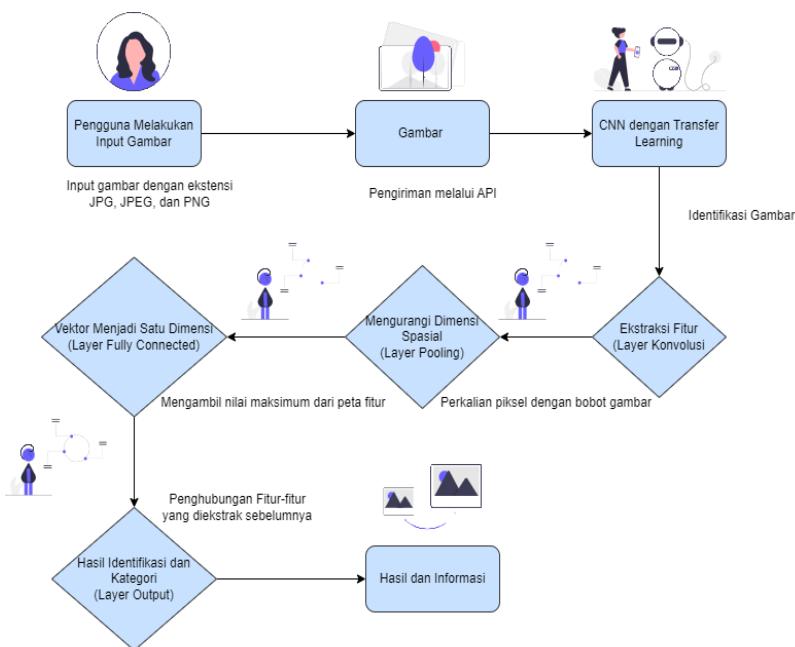
Kebutuhan non-fungsional termasuk persyaratan dimana mencakup fitur, atribut, pembatasan layanan, atau aspek-aspek lain dari sistem seperti batasan waktu, kendala proses pengembangan, dan standar sistem yang berfungsi sebagai pelengkap. Di bawah ini terdapat daftar kebutuhan fungsional yang diperlukan dalam sistem ini:

1. Batasan ekstensi gambar yang dapat digunakan sebagai *input* adalah .JPG, .PNG, dan .JPEG.
2. Batasan alat rumah tangga yang dapat digunakan adalah, *Air Conditioner*, pengering rambut, setrika, lampu, laptop, oven, kulkas, penanak nasi, televisi, penyedot debu, dan mesin cuci.

3. *User Interface* pada sistem akan mudah dipahami oleh pengguna.
4. Memiliki kemampuan untuk mendeteksi kesalahan atau *error* apabila pengguna melakukan *input data* yang tidak sesuai.
5. Tidak membutuhkan biaya yang besar dan tidak memerlukan perangkat tambahan.
6. Internet diperlukan untuk terhubung dengan sistem.

3.1.3 Diagram Umum Sistem

Diagram berikut merupakan tampilan visual tentang jalannya sistem dimana memperlihatkan bagaimana proses, aliran, dan interaksi antara komponen-komponen dalam sistem berlangsung. Seluruh desain aplikasi ini akan dijelaskan secara rinci dalam ilustrasi berikut:



Gambar 3.1 Diagram Umum Sistem

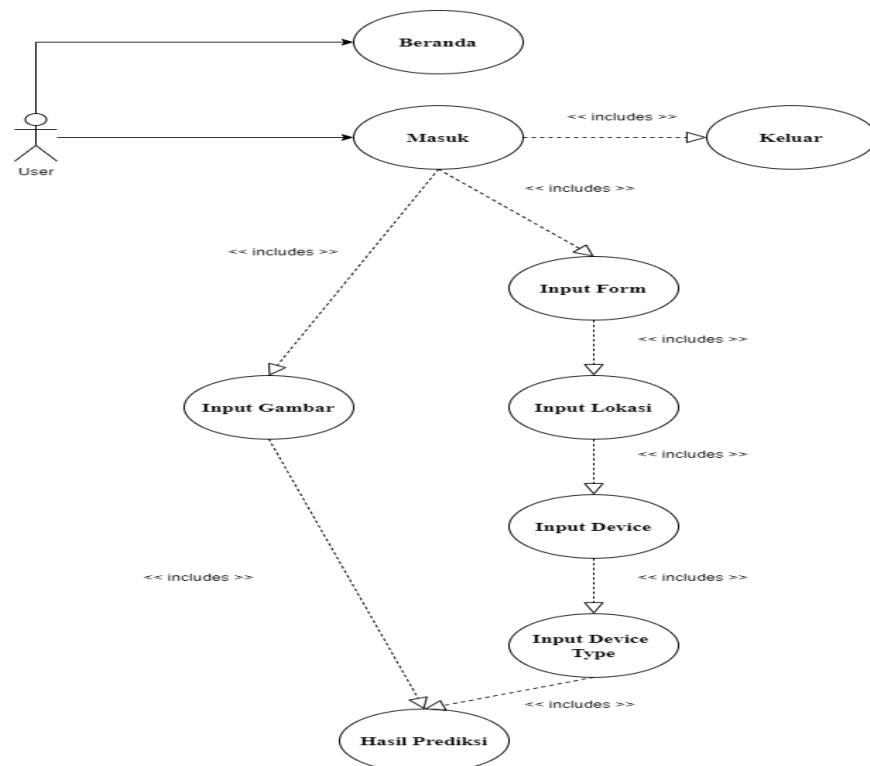
Gambar 3.1 dapat dilihat diagram umum sistem. Dimulai dari pengguna melakukan *input* gambar kedalam sistem dan kemudian gambar diidentifikasi oleh model dan diproses melalui 3 lapisan yaitu lapisan konvolusi dimana gambar akan dilakukan ekstraksi.

3.2. Pemodelan Sistem

Pemodelan sistem menjadi tahapan menguraikan langkah-langkah interaksi antara pengguna dengan aplikasi yang dibangun, sehingga sistem dapat beroperasi secara efisien. Biasanya, pemodelan sistem diwujudkan dalam format *Unified Modelling Language* (UML). UML merupakan bahasa paling umum dalam melakukan pemodelan. Digunakan untuk menggambarkan hubungan antara komponen-komponen dalam sistem, sehingga memungkinkan interaksi melalui pengguna. Dalam penelitian berikut, diagram yang menjadi dasar sebagai model adalah *Activity Diagram*, *Sequence Diagram*, serta *Use Case Diagram*.

3.2.1 Use Case Diagram

Diagram berikut dipakai demi melampirkan hubungan bagian perangkat atau aplikasi dengan aktor-aktor yang berinteraksi dengan sistem tersebut. Aktor merupakan faktor luar yang berperan dalam berinteraksi bersama sistem, seperti *User*, *Hardware*, atau juga sistem.

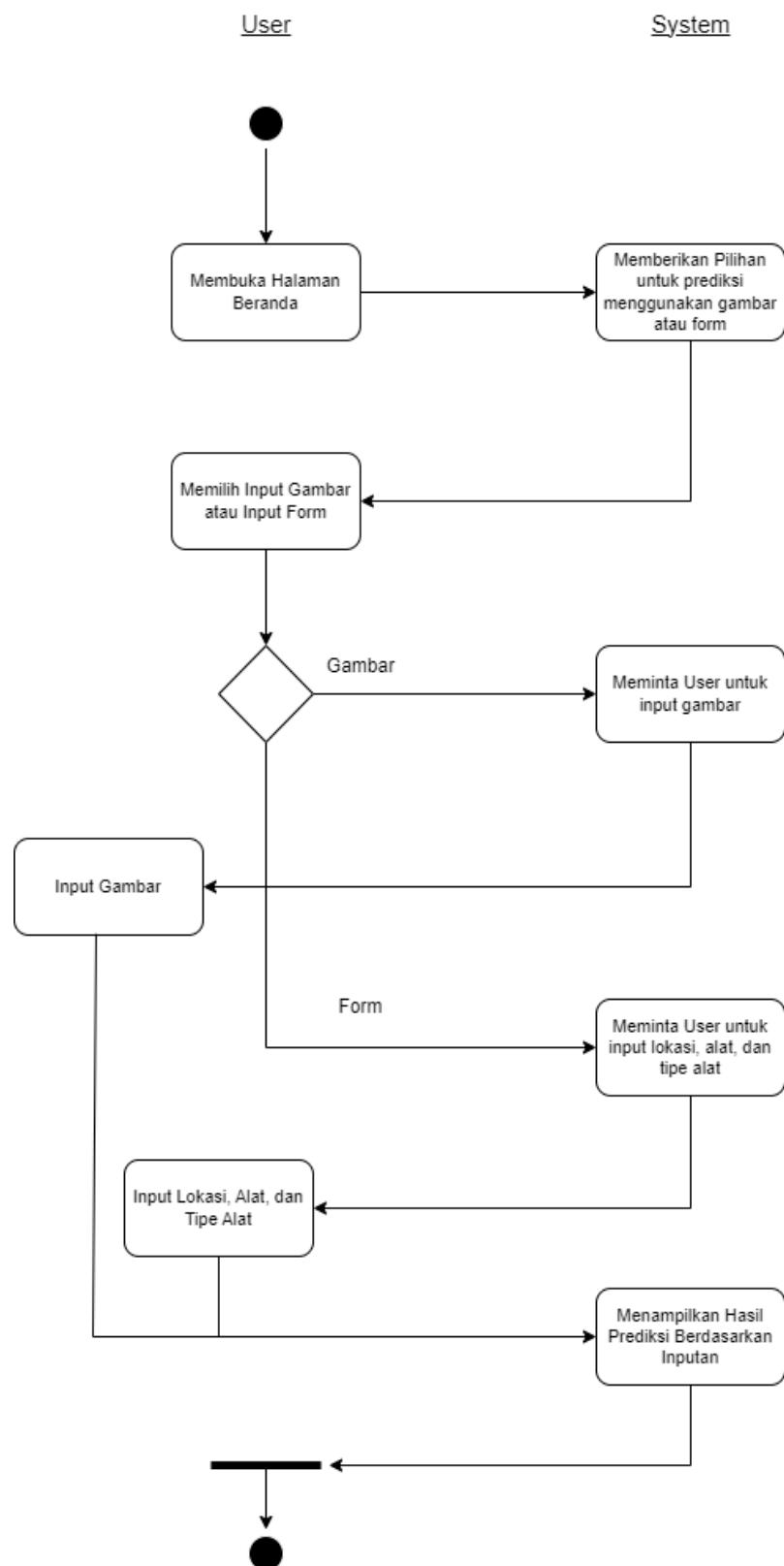


Gambar 3.2 Use Case Diagram

Gambar 3.2 menunjukkan hubungan aktor (*pengguna/user*) dengan sistem. *User* akan memulai interaksi dengan masuk ke bagian beranda aplikasi dan dapat langsung masuk ke inti program. *User* dapat memilih diantara 2 menu untuk berinteraksi. Yang pertama *User* dapat memilih Input Gambar dimana dengan memasukkan sebuah gambar kedalam aplikasi maka akan dilakukan prediksi energi terhadap alat tersebut dan dalam beberapa saat akan ditampilkan hasilnya. Yang kedua *User* dapat memilih Input Form dimana dengan memasukkan data-data seperti lokasi, alat, dan tipe alat, *User* dapat melihat prediksi penggunaan energi berdasarkan data yang diberikan.

3.2.2 *Activity Diagram*

Diagram tersebut merupakan tampilan visual atas urutan proses kegiatan pada sistem, dimulai dari langkah awal hingga mencapai langkah akhir. *Activity Diagram* juga berfungsi untuk menggambarkan bagian-bagian komponen yang ada dalam *Use Case Diagram*. Perihal konteks penelitian, berikut yang akan dijadikan konsep:

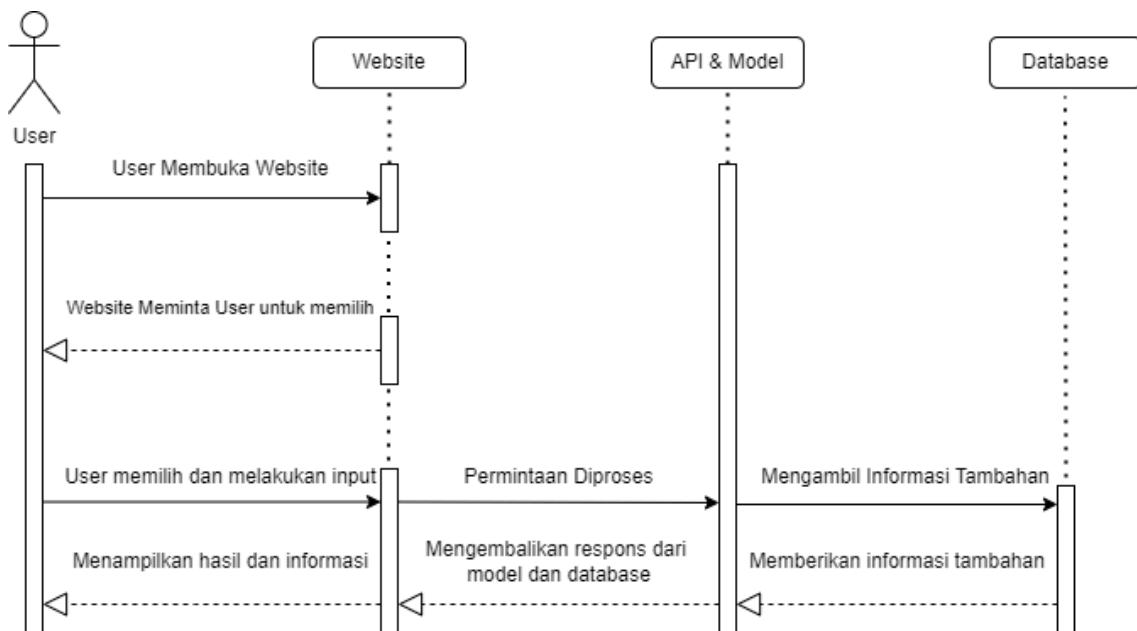


Gambar 3.3 Activity Diagram

Gambar 3.3 menampilkan diagram aktivitas dari sistem. Dimulai dari user yang membuka halaman beranda aplikasi dan pengguna akan diberi opsi oleh sistem apakah ingin melakukan prediksi dengan gambar atau form. Apabila *User* memilih gambar maka sistem akan meminta kembali kepada *User* untuk melakukan *input* gambar. *User* kemudian melakukan *input* gambar dan sistem akan menampilkan hasil prediksi tersebut kepada *User*. Apabila *User* memilih form, maka sistem akan meminta kepada *User* agar memasukkan *input* lokasi, alat, serta tipe alat. Setelah itu sistem kemudian mengeluarkan hasil prediksi kembali kepada *User*.

3.2.2 Sequence Diagram

Digunakan untuk memvisualisasikan hubungan antar objek atau komponen dalam sistem dalam urutan waktu tertentu. Diagram ini sangat berguna untuk memvisualisasikan bagaimana objek-objek berkomunikasi satu sama lain dalam konteks sebuah skenario atau proses tertentu.



Gambar 3.4 Sequence Diagram

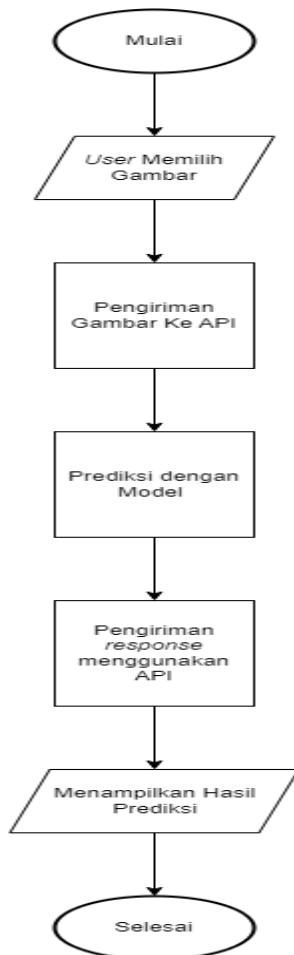
Gambar 3.4 menjelaskan bahwa dapat dilihat 4 komponen yang saling berinteraksi. Diagram diatas menjelaskan proses dari awal *User* membuka halaman *Website* hingga permintaan *User* selesai dilaksanakan.

3.3. Flowchart

Flowchart (Diagram Alir) adalah representasi visual dari serangkaian langkah-langkah yang dijalankan oleh sebuah algoritma secara teratur. Setiap tahapan prosesnya direpresentasikan oleh simbol-simbol yang berbeda dan dilengkapi dengan penjelasan untuk setiap tahapnya.

3.3.1 Flowchart Sistem

Berikut dilampirkan diagram alir sederhana untuk penelitian tersebut:

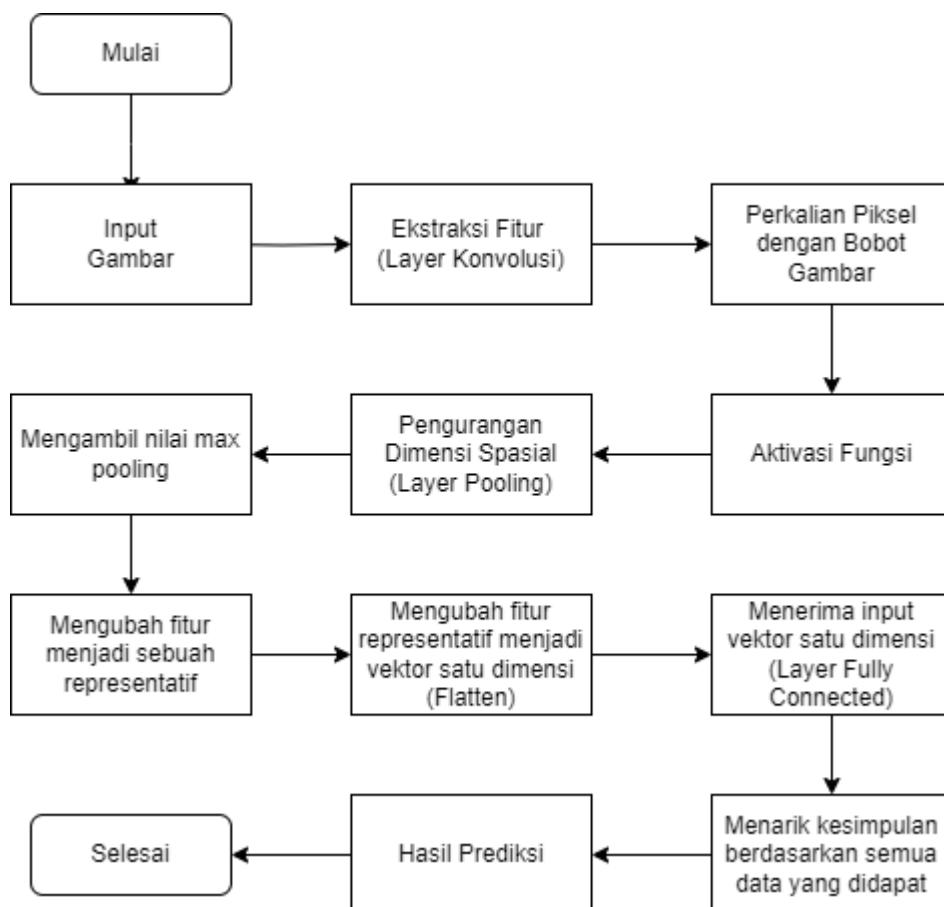


Gambar 3.5 Flowchart Sistem

Gambar 3.5 menjelaskan diagram alir dari sistem yang pertama yakni memilih gambar yang akan digunakan. Kemudian, gambar akan dikirim ke API yang akan mengirimkan gambar tersebut kepada model. Model akan melakukan prediksi dan akan mengembalikan *response* berdasarkan hasil prediksi kepada *user* melalui API. API kemudian menampilkan hasil dan informasi relevan terhadap prediksi yang dilakukan oleh model.

3.3.2 Diagram Arsitektur Algoritma *Convolutional Neural Network* (CNN)

Berikut adalah arsitektur sederhana dari algoritma CNN:



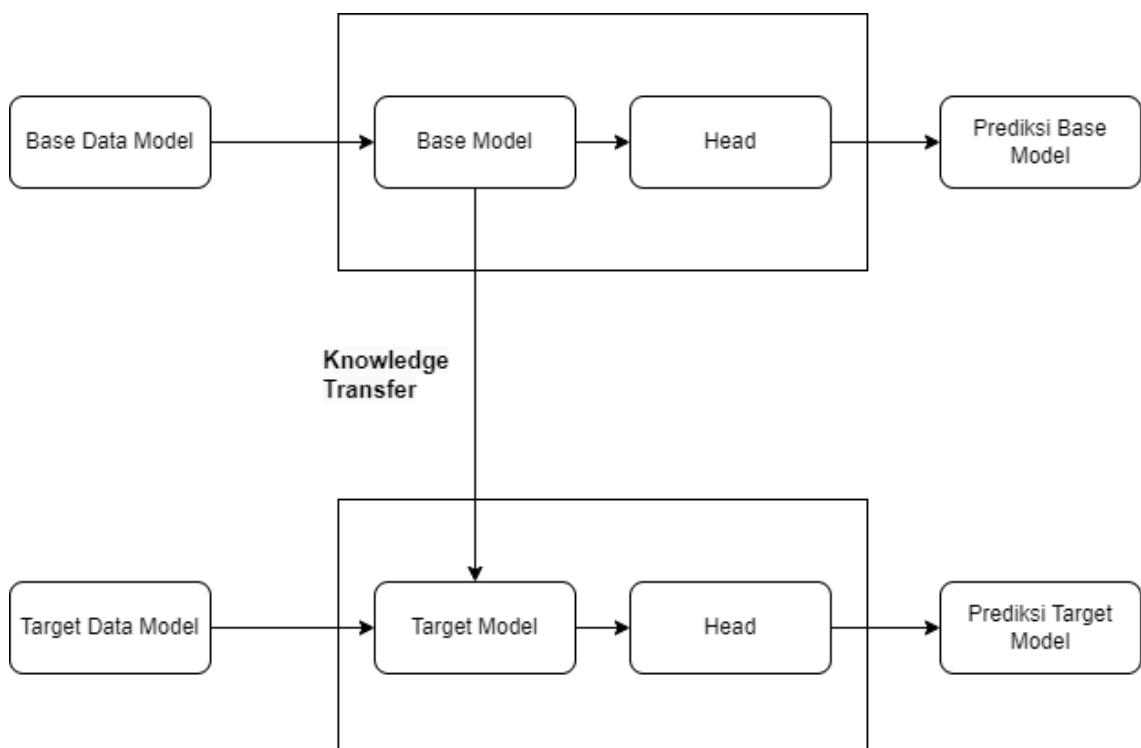
Gambar 3.6 Diagram Arsitektur Algoritma CNN

Gambar 3.6 menjelaskan cara kerja dari algoritma CNN. Dimulai dengan *input* gambar yang diterima kemudian dilakukan ekstraksi fitur pada lapisan konvolusi. Pada lapisan ini dilakukan ekstraksi fitur dari gambar dan perkalian

piksel terhadap bobot gambar. Kemudian diterapkan sebuah fungsi aktivasi untuk mengenalkan non linearitas kepada gambar. Setelah itu masuk ke lapisan *pooling* dimana dilakukan pengurangan dimensi spasial agar model dapat lebih cepat memahami gambar yang diberikan serta menghindari *overfitting*. Kemudian diambil nilai *max pooling* yang akan digunakan untuk memecah fitur menjadi sebuah representasi fitur agar model dapat mempelajari pola lebih cepat tanpa mengurangi informasi yang ada pada gambar atau piksel. Kemudian dilakukan teknik *flatten*, yaitu perubahan fitur menjadi vektor satu dimensi. Hal ini dilakukan karena lapisan *fully connected* hanya mampu menerima *input* dalam bentuk vektor satu dimensi. Setelah itu lapisan *fully connected* menarik kesimpulan berdasarkan semua data yang diterima dan dapatlah hasil prediksi dari gambar tersebut.

3.3.3 Diagram Arsitektur *Transfer Learning*

Berikut adalah diagram sederhana dari arsitektur *Transfer Learning*:



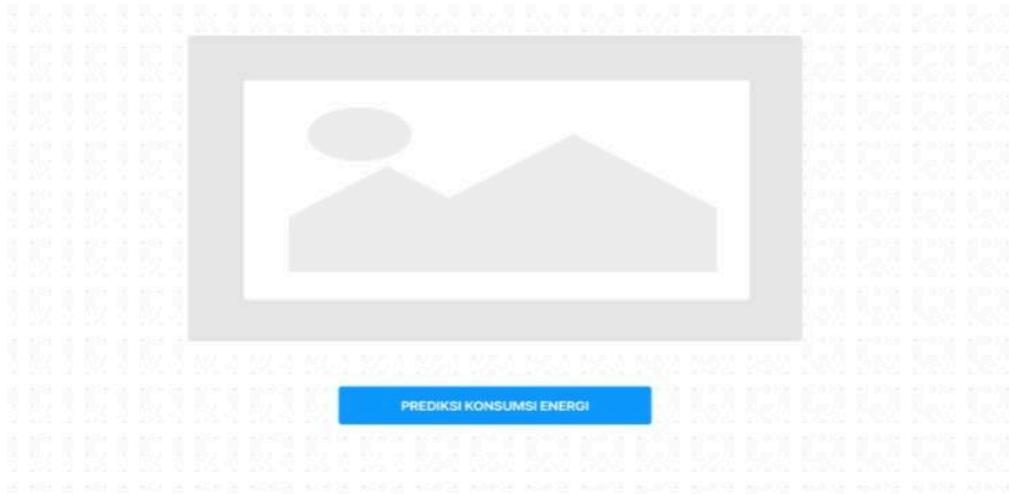
Gambar 3.7 Diagram Arsitektur Transfer Learning

Pada gambar 3.7 dapat dilihat bagaimana cara kerja dari arsitektur *Transfer Learning*. Secara sederhana model mengambil *knowledge* dari model yang telah dilatih sebelumnya. Kemudian data tersebut dilatih kembali dan dilakukan *mapping* untuk kelas tertentu pada bagian *head* hingga akhirnya model selesai dilatih dan dapat melakukan prediksi.

3.4. Perancangan *Interface*

Perancangan *Interface* adalah proses menciptakan kerangka desain sistem yang akan dikonstruksi. Perancangan *Interface* adalah langkah yang diperlukan untuk memastikan bahwa proses pembuatan sistem dapat berjalan dengan lebih efisien berdasarkan panduan yang telah didefinisikan dalam desain antarmuka tersebut.

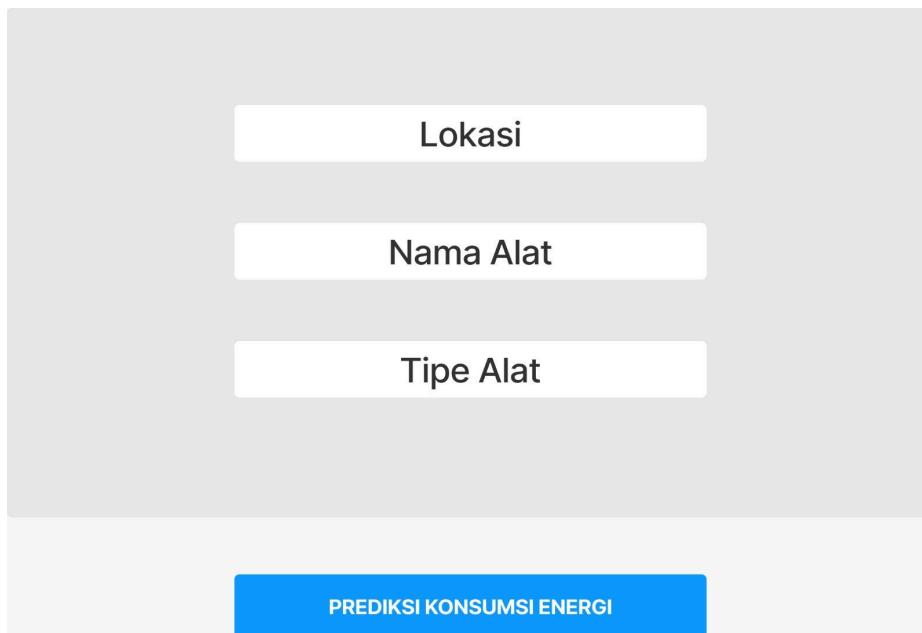
3.4.1 Halaman Prediksi dengan Gambar



Gambar 3.8 Halaman Prediksi Dengan Gambar

Gambar 3.8 menunjukkan secara singkat bagaimana antarmuka halaman prediksi dengan gambar akan ditampilkan. Pada halaman ini pengguna akan melakukan *input* gambar yang kemudian gambar tersebut akan ditampilkan dan pengguna dapat menekan tombol *predict* untuk mengetahui penggunaan energi alat tersebut.

3.4.2 Halaman Prediksi dengan *Form*



Gambar 3.9 Halaman Prediksi Dengan Form

Gambar 3.9 menunjukkan secara singkat bagaimana antarmuka halaman prediksi dengan *form* akan ditampilkan. Pada halaman ini pengguna akan melakukan *input* terhadap tiga hal yakni lokasi, perangkat, dan tipe perangkat. Selanjutnya pengguna dapat klik tombol *predict* untuk mengetahui keluaran penggunaan energi dari alat..

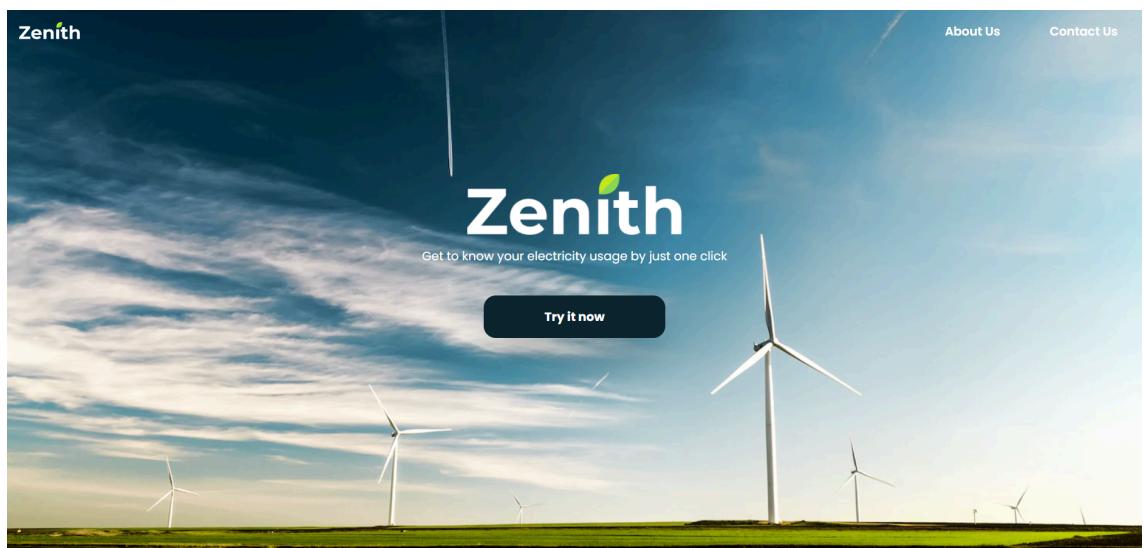
BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1. Implementasi

Dalam penelitian berikut, sistem dibangun menggunakan algoritma CNN dan arsitektur *transfer learning*. Sistem akan dibangun agar memiliki bagian *front-end* serta *back-end*. Bagian *front-end* akan diciptakan dengan bantuan *library* JavaScript (JS) yaitu react.js dan *back-end* akan dibangun menggunakan bantuan bahasa pemrograman *python* dan *Structured Query Language* (SQL). Sistem juga akan dibantu menggunakan aplikasi seperti *postman* dan *visual studio code*.

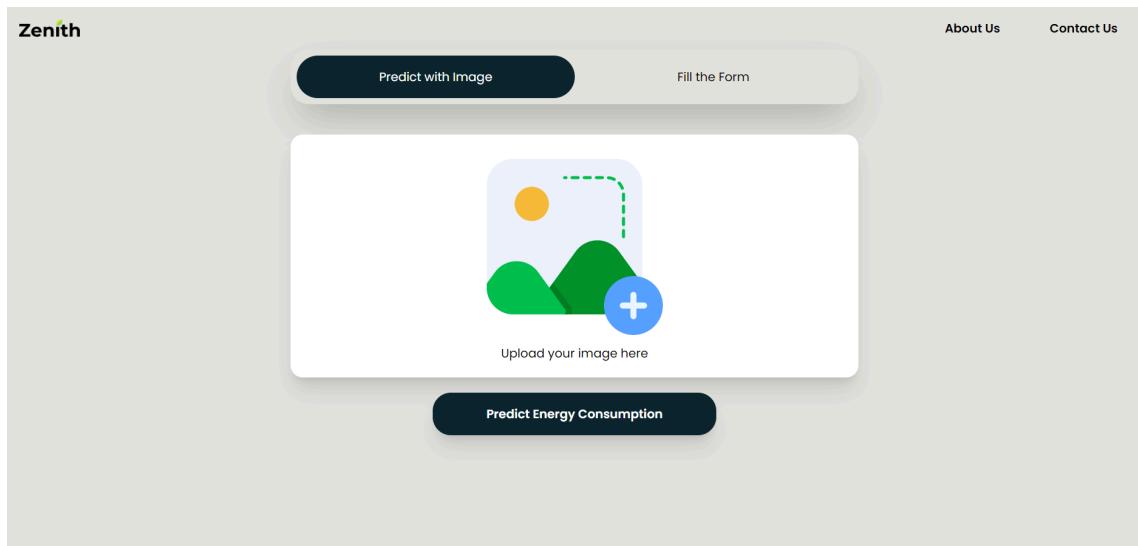
4.1.1 *Landing Page*



Gambar 4.1 Landing Page Sistem

Pada gambar 4.1, dapat dilihat *landing page* yang digunakan oleh sistem.

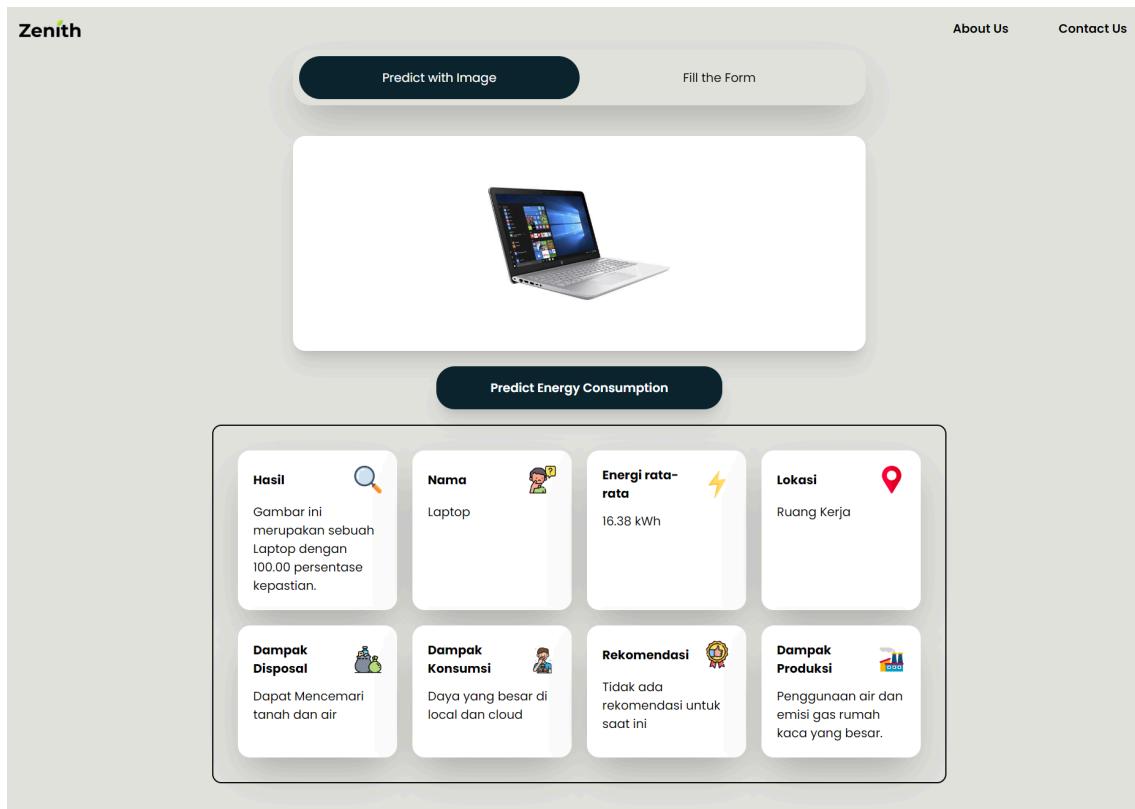
4.1.2 Halaman Prediksi dengan Gambar



Gambar 4.2 Halaman Prediksi dengan Gambar

Gambar 4.2 merupakan halaman prediksi dengan gambar. Pada halaman ini, pengguna dapat melakukan *input* terhadap gambar yang dapat diambil dari perangkat mereka dan melakukan prediksi konsumsi energi dengan menekan tombol *predict energy consumption*.

4.1.3 Halaman Hasil Prediksi dengan Gambar



Gambar 4.3 Halaman Hasil Prediksi dengan Gambar

Gambar 4.3 merupakan halaman hasil prediksi dengan gambar. Ketika pengguna menekan tombol *predict energy consumption* setelah melakukan *input* gambar, hasil prediksi konsumsi energi alat tersebut akan ditampilkan. Ada beberapa informasi yang didapat oleh user ketika mendapatkan hasil prediksi. Dimulai dari nama alat yang di prediksi, rata-rata penggunaan energi alat tersebut, lokasi biasanya ditempatkan alat tersebut, dampak disposal, dampak konsumsi, dan dampak produksi alat tersebut serta rekomendasi yang dapat digunakan pengguna agar lebih menghemat penggunaan energi mereka.

4.1.4 Halaman Prediksi dengan *Form*

Gambar 4.4 Halaman Prediksi dengan Form

Gambar 4.4 menampilkan halaman prediksi menggunakan *form*. Pada halaman ini, pengguna dapat melakukan *input* terhadap beberapa data yang akan digunakan untuk melakukan prediksi alat berdasarkan ciri-ciri yang mereka berikan pada *form*. Setelah memasukkan semua data diatas, pengguna dapat menekan tombol *predict energy consumption* untuk melihat hasil prediksi dari model untuk alat yang datanya diberikan.

4.1.5 Halaman Hasil Prediksi dengan *Form*

Gambar 4.5 Halaman Hasil Prediksi dengan Form

Gambar 4.5 menampilkan halaman hasil prediksi menggunakan *form*. Disini, pengguna dapat melihat hasil prediksi konsumsi energi terhadap alat yang datanya dimasukkan tadi. Pada halaman ini hanya akan menampilkan konsumsi energi berdasarkan prediksi model terhadap ciri-ciri alat yang dimasukkan tadi.

4.2. Pengujian Sistem

Berikut merupakan tahapan yang dimana apabila tahapan implementasi telah berhasil dilakukan. Pengujian sistem memiliki fungsi untuk menampilkan bahwa sistem yang diciptakan mampu bekerja secara benar dalam memprediksi konsumsi energi sebuah alat. Pada tahap pengujian, terdapat 11 alat yang akan digunakan untuk prediksi energi. Alat yang digunakan adalah, *Air Conditioner* (AC), Oven, Mesin Cuci, Laptop, Pengering Rambut, Setrika, Lampu, Kulkas, Penanak Nasi, Televisi, dan Penyedot Debu. Berikut adalah tabel yang akan menampilkan alat dan jenis alat yang digunakan.

Tabel 4.1 Daftar Nama dan Tipe Alat

No.	Nama Alat	Tipe Alat
1.	Air Conditioner	Copper AC
		Alloy AC
		R-22 AC
		R-32 AC
		R-34 AC
		R-38 AC
		R410a AC
2.	Oven	Less Than or Equal to 1 Cubic Ft Oven
		More Than 1 Cubic Ft Oven

		Plastic-Based Oven
		Stainless Steel + Glass + Plastic + Steel
		Glass
		Stainless Steel
		Stainless Steel + Glass
		Metal
		Metal + Glass + Plastic
		Metal + Stainless Steel
		Steel
		Stainless Steel + Plastics
		Glass + Plastic + Metal
		Aluminium
3.	Lampu	G9
		E12 (Candelabra)
		E17
		GU10
		GX5.3
		GU5.3
		E26 (Medium)
		GU24
		E26d
4.	Televisi	13 - 20 inch
		21 - 28 inch
		31 - 39.98 inch

		40 - 49.54 inch
		50 - 59.5 inch
		60 - 69.51 inch
		70 - 78.5 inch
		80.49-85.6 inch
5.	Laptop	Low Power & Moderate Laptop
		High-End Laptops
6.	Pengering Rambut	General Purpose
7.	Setrika	General Purpose
8.	Kulkas	General Purpose
9.	Penanak Nasi	General Purpose
10.	Penyedot Debu	General Purpose
11.	Mesin Cuci	General Purpose

Berikut adalah *step-by-step* penggunaan sistem, yang dimulai dengan pembuatan model.

```
[2] import tensorflow as tf
from tensorflow.keras.applications import DenseNet121
from tensorflow.keras.layers import Dense
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import glob
```

Gambar 4.6 Import Library

1. Gambar 4.6 menjelaskan hal yang pertama sekali perlu dilakukan adalah melakukan *import* terhadap *library* yang akan digunakan. Pada model, diperlukan beberapa *library* guna mendukung penyelesaian latihan. *Library* yang diperlukan antara lain yaitu, *tensorflow*, *keras*, *glob*, dan *matplotlib*.

```

train_data_directory = '/content/drive/MyDrive/Dataset/train'
validation_data_directory = '/content/drive/MyDrive/Dataset/validation'
batch_size = 32
# Create data generators for training and validation with augmentation
train_datagen = ImageDataGenerator(
    rescale=1.0/255.,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

val_datagen = ImageDataGenerator(rescale=1.0/255.)

```

Gambar 4.7 Persiapan Training Model

2. Pada gambar diatas, dapat dilihat *dataset* yang siap digunakan untuk pelatihan model. Terdapat dua jenis direktori untuk pelatihan yakni *training dataset* dan *validation dataset*. Hal ini digunakan agar model dapat belajar bukan hanya dari *training data* namun juga data yang belum pernah dilihat sebelumnya. Hal ini juga mampu mengatasi *overfitting* pada model. Untuk perbandingan data latihan dan validasi digunakan perbandingan 70:30. Dimana data latih memiliki 70% gambar dan validasi memiliki 30%.

```

# Prepare the training and validation data with labels
train_generator = train_datagen.flow_from_directory(
    train_data_directory,
    target_size=(224, 224),
    batch_size=batch_size,
    class_mode='categorical'
)

validation_generator = val_datagen.flow_from_directory(
    validation_data_directory,
    target_size=(224, 224),
    batch_size=batch_size,
    class_mode='categorical'
)

# Load the pre-trained DenseNet model without top layers
base_model = DenseNet121(include_top=False, weights='imagenet', input_shape=(224, 224, 3))

# Freeze the base model layers
base_model.trainable = False

```

Gambar 4.8 Pemberian Label dan Load Pre-trained Model

3. Pada gambar diatas, dapat dilihat bahwa dataset akan diberikan label dan program akan mengambil data gambar yang ada dari direktori yang disediakan. Gambar akan diubah menjadi ukuran 224 piksel x 224 piksel. Kemudian dilakukan *load* terhadap model yang telah dilatih sebelumnya, yakni model *DenseNet 121* yang juga telah dilatih menggunakan *dataset ImageNet*. Selanjutnya akan dilakukan pembekuan atau *freeze* terhadap lapisan yang ada pada model *DenseNet 121*. Hal ini dilakukan agar pada model *DenseNet 121*, bobot tidak diperbarui dan model juga tidak akan diperbarui karena kita akan menggunakan arsitektur *transfer learning*.

```
# Add your own classification head on top of the base model
model = tf.keras.Sequential([
    base_model,
    tf.keras.layers.GlobalAveragePooling2D(),
    Dense(11, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model using the generators for initial training
history = model.fit(
    train_generator,
    epochs=50,
    validation_data=validation_generator
)
```

Gambar 4.9 Proses Compile Model dan Melatih Model

4. Pada gambar diatas, proses pelatihan model akan segera dimulai. Dengan tahapan awal inisiasi pembangunan model dengan metode *Sequential* yang akan memiliki 11 neuron dan fungsi aktivasi *softmax*. *Softmax* digunakan sebagai fungsi aktivasi disini karena *softmax* cocok untuk *multiclass classification*, mendukung probabilitas normalisasi *backpropagation* efektif, dan keluaran yang komprehensif. Selanjutnya model akan dikompilasi menggunakan *adam optimizer*. Selanjutnya model dilatih dengan 50 kali iterasi (*epochs*) menggunakan *data generator* yang telah disiapkan sebelumnya. Digunakan 50 iterasi karena setelah dilakukan percobaan, akurasi model telah mencapai 97.61%.

```
Found 7714 images belonging to 11 classes.
Found 3310 images belonging to 11 classes.
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet121\_weights\_tf\_dim\_ordering\_tf\_kernels\_notop.h5
29084464 [=====] - 2s 0us/step
Epoch 1/50
30/242 [=>.....] - ETA: 1:01:34 - loss: 2.1396 - accuracy: 0.2698/usr/local/lib/python3.10/dist-packages/PIL/Image.py:996: UserWarning:
242/242 [=====] - 6516s 27s/step - loss: 0.8199 - accuracy: 0.7660 - val_loss: 0.2743 - val_accuracy: 0.9257
Epoch 2/50
242/242 [=====] - 147s 606ms/step - loss: 0.3096 - accuracy: 0.9125 - val_loss: 0.1937 - val_accuracy: 0.9450
Epoch 3/50
242/242 [=====] - 146s 602ms/step - loss: 0.2358 - accuracy: 0.9317 - val_loss: 0.1600 - val_accuracy: 0.9544
Epoch 4/50
242/242 [=====] - 144s 594ms/step - loss: 0.2019 - accuracy: 0.9398 - val_loss: 0.1605 - val_accuracy: 0.9495
Epoch 5/50
242/242 [=====] - 142s 586ms/step - loss: 0.1835 - accuracy: 0.9415 - val_loss: 0.1383 - val_accuracy: 0.9562
Epoch 6/50
242/242 [=====] - 150s 618ms/step - loss: 0.1669 - accuracy: 0.9476 - val_loss: 0.1365 - val_accuracy: 0.9553
```

Gambar 4.10 Proses Pelatihan Model dengan 50 Iterasi

5. Pada gambar diatas, proses pelatihan model sedang berjalan dan dapat dilihat perkembangan akurasi, *loss function*, *val_loss*, *val_accuracy* pada model yang perlahan akan meningkat sesuai dengan iterasi.

```
import matplotlib.pyplot as plt

# Get the training and validation accuracy values
train_acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

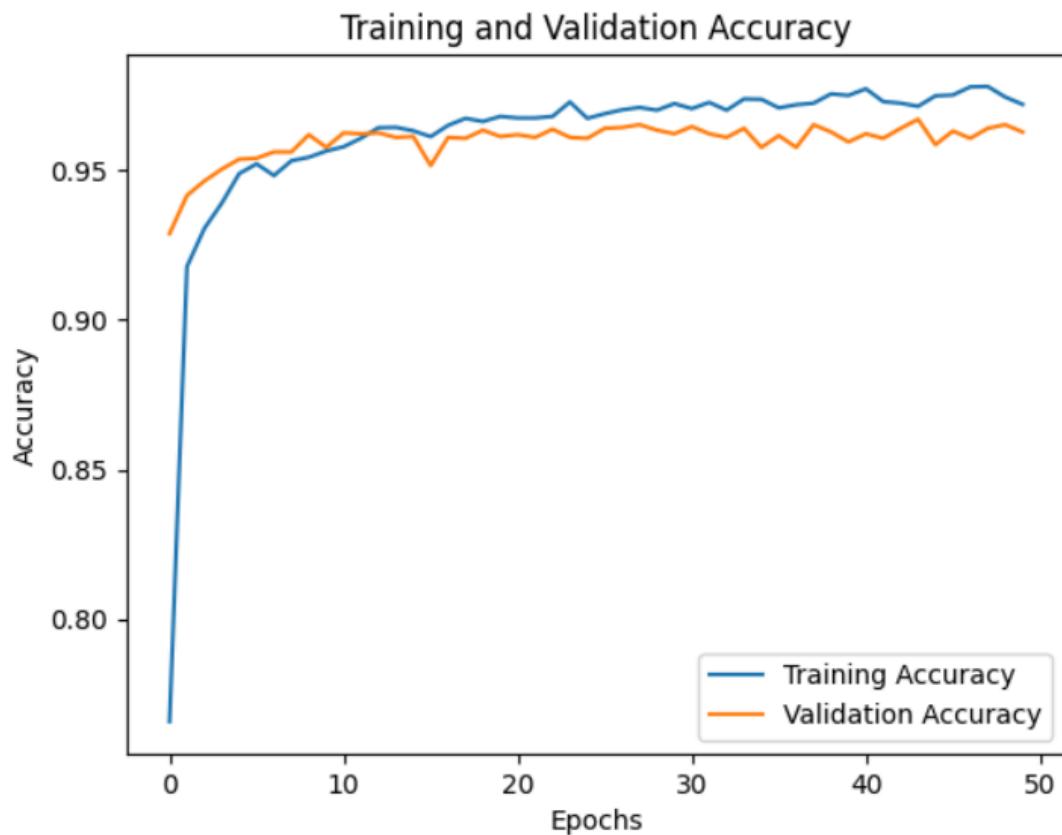
# Get the training and validation loss values
train_loss = history.history['loss']
val_loss = history.history['val_loss']

# Plot the accuracy values
plt.plot(train_acc, label='Training Accuracy')
plt.plot(val_acc, label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

# Plot the loss values
plt.plot(train_loss, label='Training Loss')
plt.plot(val_loss, label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

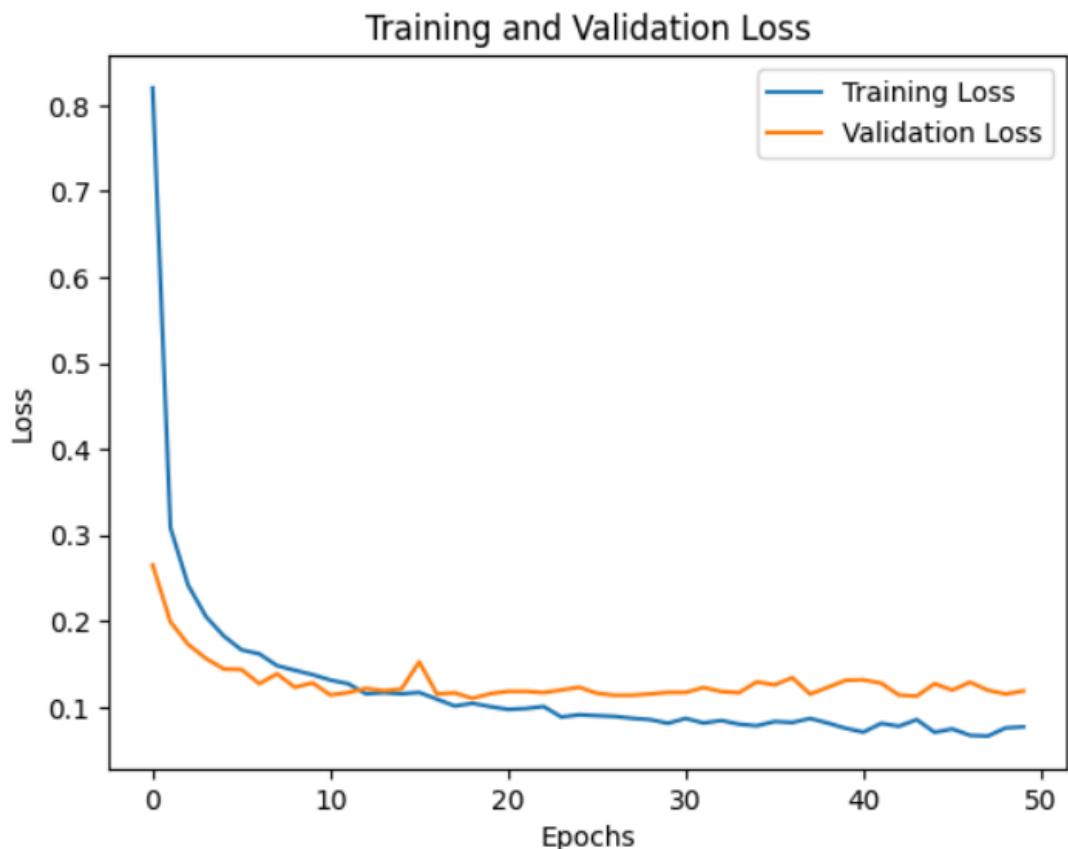
Gambar 4.11 Menampilkan data hasil pelatihan dengan library matplotlib

6. Pada gambar diatas, terdapat kode yang digunakan untuk menampilkan grafik yang akan mengambil data berdasarkan hasil pelatihan model. Grafik akan dibangun menggunakan bantuan *library matplotlib*. Grafik akan menampilkan data seperti *history*, *accuracy*, *loss* dari data latihan dan validasi.



Gambar 4.12 Grafik Training dan Validation Accuracy

7. Pada gambar diatas, terdapat grafik yang dihasilkan berdasarkan akurasi latihan dan validasi data. Kedua garis terlihat saling berdampingan pada bagian atas grafik. Hal ini menunjukkan bahwa model dapat melakukan prediksi yang baik dengan data latihan serta data yang baru pertama kali diperlihatkan.



Gambar 4.13 Grafik Training dan Validation Loss

8. Pada gambar diatas, terdapat grafik yang dihasilkan berdasarkan *loss* latihan dan validasi. Hal ini mengindikasikan bahwa model mempelajari pola data dengan baik pada saat latihan dan dapat mempelajari data baru dengan baik pula pada validasi.

```
[ ] # Save the model in the Keras HDF5 format  
model.save("model/accDenseNet v1.3/model v1.3.h5")
```

Gambar 4.14 Penyimpanan model dengan ekstensi h5

9. Pada gambar diatas, terdapat baris kode yang menunjukkan bahwa model akan diekspor menggunakan ekstensi h5. h5 merupakan sebuah ekstensi populer apabila kita ingin menyimpan model. h5 memberikan fleksibilitas karena pengembang tidak perlu lagi melatih model yang telah disimpan dalam ekstensi h5. Dengan ekstensi h5, data yang terdapat pada model tersimpan dan tidak perlu lagi dilatih ulang.

```
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import numpy as np

# Define the class labels
class_labels = ['Air Conditioner', 'Hair Dryer', 'Iron', 'Lamp','Laptop', 'Oven', 'Refrigerator', 'Rice Cooker',
                 'Television', 'Vacuum Cleaner', 'Washing Machine']

# List of image paths
image_paths = [
    '/content/drive/MyDrive/Datatesting/ac.png',
    '/content/drive/MyDrive/Datatesting/kulkas.jpg',
    '/content/drive/MyDrive/Datatesting/lampu.png',
    '/content/drive/MyDrive/Datatesting/laptop.png',
    '/content/drive/MyDrive/Datatesting/wash.png',
    '/content/drive/MyDrive/Datatesting/oven.jpg',
    '/content/drive/MyDrive/Datatesting/penanak_nasi.png',
    '/content/drive/MyDrive/Datatesting/pengering_rambut.png',
    '/content/drive/MyDrive/Datatesting/setrika.png',
    '/content/drive/MyDrive/Datatesting/tv.jpg',
    '/content/drive/MyDrive/Datatesting/vac.jpg'
]
```

Gambar 4.15 Pemberian label kelas dan pengujian model dengan data testing

10. Pada gambar diatas, terdapat baris kode yang menunjukkan pemberian label pada kelas dan akan dilakukan pengujian model dengan memanfaatkan *data testing*. Tindakan ini dimaksudkan demi mencari tahu bahwa model dapat memprediksi data baru dan belum pernah dilihat sama sekali pada data latihan maupun validasi.

```

# Load your trained model
model = tf.keras.models.load_model('model/accDenseNet v1.3/model v1.3.h5') # Replace with the path to your trained model

# Loop through the image paths
for image_path in image_paths:
    # Load the image
    img = image.load_img(image_path, target_size=(224, 224))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)

    # Preprocess the image
    img_array = img_array / 255.0

    # Make a prediction
    predictions = model.predict(img_array)
    predicted_class_index = np.argmax(predictions[0])
    predicted_class = class_labels[predicted_class_index]
    confidence = predictions[0][predicted_class_index] * 100 # Confidence in percentage

    # Print the predicted class and confidence
    print('Image:', image_path)
    print('Predicted class:', predicted_class)
    print('Confidence:', confidence, '%')
    print('---')

```

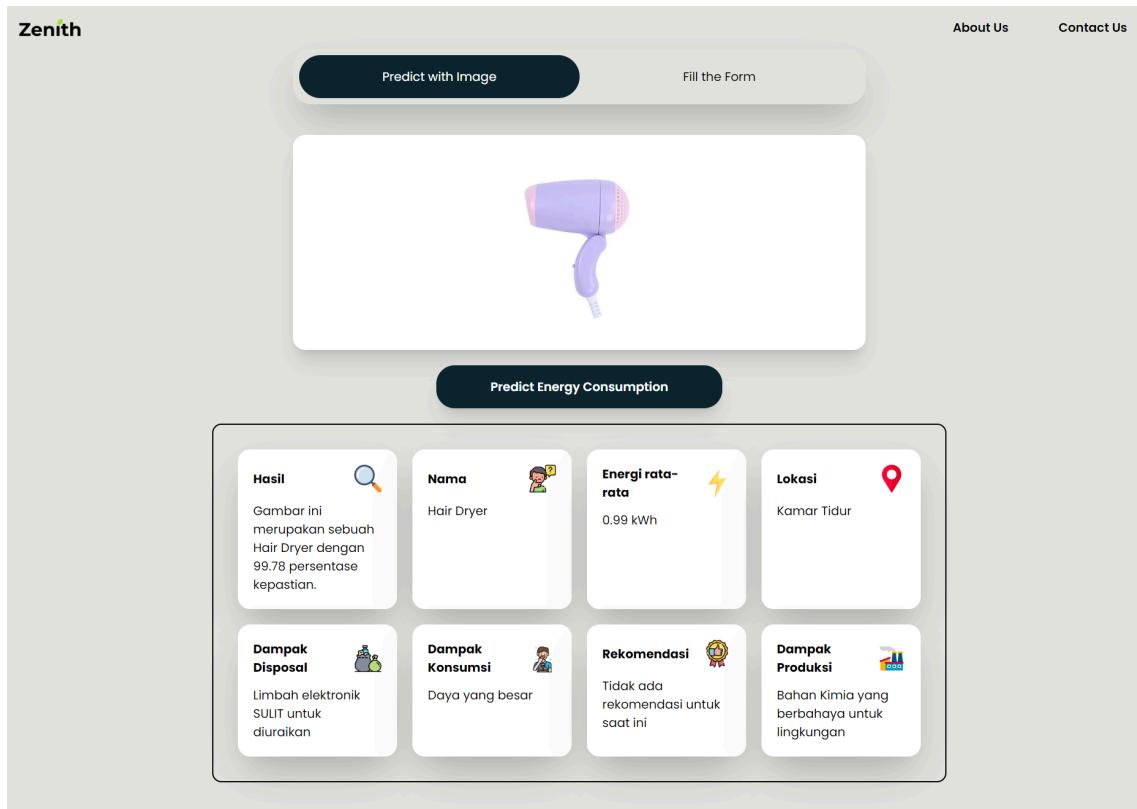
Gambar 4.16 Melakukan testing terhadap model

11. Pada gambar diatas, terdapat baris kode yang menunjukkan model *testing*. Dimulai dengan memuat model yang disimpan dalam bentuk h5 sebelumnya serta membuat perulangan untuk memuat gambar. Kemudian dilakukan *preprocessing* pada gambar dan akhirnya membuat prediksi sekaligus menampilkan keluaran dari hasil prediksi setiap gambar yang dimuat.

```
1/1 [=====] - 2s 2s/step
Image: /content/drive/MyDrive/Datatesting/ac.png
Predicted class: Air Conditioner
Confidence: 99.9990463256836 %
---
1/1 [=====] - 0s 26ms/step
Image: /content/drive/MyDrive/Datatesting/kulkas.jpg
Predicted class: Refrigerator
Confidence: 99.99996423721313 %
---
1/1 [=====] - 0s 26ms/step
Image: /content/drive/MyDrive/Datatesting/lampu.png
Predicted class: Lamp
Confidence: 99.99998807907104 %
---
1/1 [=====] - 0s 28ms/step
Image: /content/drive/MyDrive/Datatesting/laptop.png
Predicted class: Laptop
Confidence: 99.99998807907104 %
---
1/1 [=====] - 0s 35ms/step
Image: /content/drive/MyDrive/Datatesting/wash.png
Predicted class: Washing Machine
Confidence: 99.98724460601807 %
---
1/1 [=====] - 0s 27ms/step
Image: /content/drive/MyDrive/Datatesting/oven.jpg
Predicted class: Oven
Confidence: 99.99829530715942 %
```

Gambar 4.17 Tampilan keluaran model

12. Pada gambar diatas, terdapat hasil keluaran prediksi gambar yang dilakukan. Dapat dilihat, model berhasil memprediksi data baru yang dimasukkan dengan akurasi tinggi. Ini menunjukkan bahwa model telah dilatih dengan baik dan siap melakukan prediksi untuk data baru yang belum pernah dilihat lainnya.



Gambar 4.18 Percobaan pada Website

13. Pada gambar diatas, dapat dilihat percobaan yang dilakukan terhadap model menggunakan *website* sistem. Pada gambar dapat terlihat data-data yang berhubungan dengan alat tersebut seperti nama, penggunaan energi, dampak konsumsi dan lain-lain. Prediksi yang dilakukan model sudah benar dan menunjukkan model telah berhasil dilatih dan dapat melakukan *testing* dengan baik.

4.3. Hasil Pengujian Sistem

Berikut merupakan hasil yang didapat dengan mencoba semua alat yang telah dipaparkan diatas:

Tabel 4.2 Daftar Hasil Percobaan

No.	Gambar	Nama Alat	Konsumsi Energi
1.		Pengering Rambut	0.99 KWh
2.		Air Conditioner	1.77 KWh
3.		Televisi	0.06 KWh
4.		Mesin Cuci	4.94 Wh
5.		Laptop	16.38 Wh

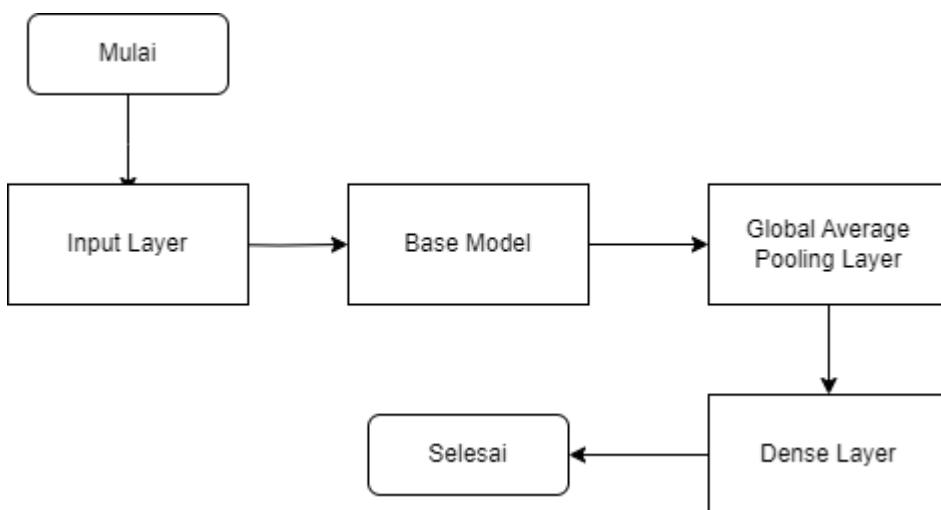
6.		Penyedot Debu	1.32 KWh
7.		Oven	1.03 KWh
8.		Setrika	0.37 KWh
9.		Penanak Nasi	13.93 Wh
10.		Kulkas	8.93 KWh
11.		Lampu	0.01 KWh

4.4. Perbandingan Model Arsitektur CNN dengan *Transfer Learning* dan tanpa *Transfer Learning*

Pada bagian ini akan dibuktikan bagaimana pengaruh *Transfer Learning* dapat meningkatkan akurasi serta kinerja model. Akan dilampirkan dua buah model dimana satu model akan ditemani oleh *Transfer Learning* dan satu lagi hanya dengan bantuan CNN saja. Berikut perbandingannya:

4.4.1 Model dengan *Transfer Learning*

Model dengan *transfer learning* berarti ketika model dibangun akan digunakan sebuah model yang telah dilatih sebelumnya dan digunakan sebagai *base model* untuk meningkatkan akurasi dan kinerja dari model target yang akan kita ciptakan. Untuk lebih memahami, berikut merupakan arsitektur sederhana dari model CNN yang dilengkapi dengan *transfer learning*:



Gambar 4.19 Diagram Arsitektur CNN dengan Transfer Learning

Diagram diatas menjelaskan bagaimana sebuah model CNN yang dilengkapi *transfer learning* bekerja. Dimulai dengan adanya lapisan *input* kemudian digunakan *base model* yang akan membantu kinerja model dan dilanjutkan dengan lapisan *pooling* dan lapisan *dense* hingga model selesai dilatih. Metode ini digunakan untuk model yang diciptakan dengan *base model*

adalah densenet121. Hasil yang didapat adalah akurasi model mencapai 97% dan model dapat memprediksi gambar baru dengan sangat baik.

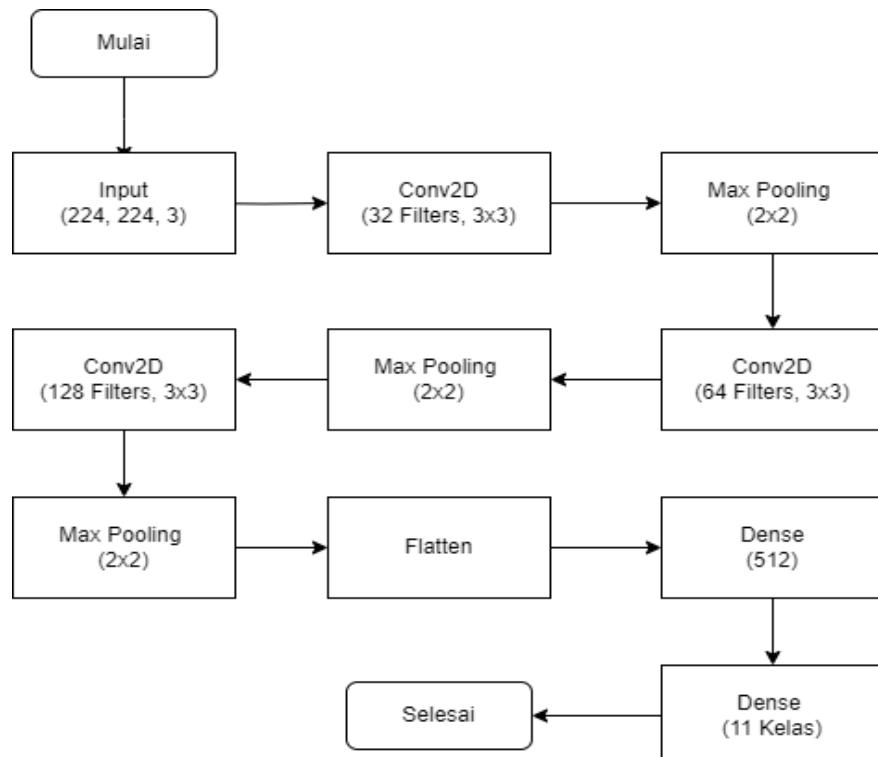
```
Epoch 49/50
242/242 [=====] - 138s 570ms/step - loss: 0.0697 - accuracy: 0.9777 - val_loss: 0.1106 - val_accuracy: 0.9653
Epoch 50/50
242/242 [=====] - 139s 573ms/step - loss: 0.0715 - accuracy: 0.9761 - val_loss: 0.1157 - val_accuracy: 0.9647
```

Gambar 4.20 Hasil Model CNN dilengkapi Transfer Learning

Gambar diatas melampirkan hasil yang didapat dari sebuah model CNN yang dilengkapi *transfer learning*. Dapat dilihat akurasi model mencapai 97% dan val_accuracy 96% serta *loss* yang cukup rendah.

4.4.2 Model tanpa *Transfer Learning*

Model selanjutnya yang akan dibahas adalah model CNN tanpa *transfer learning*. Model ini hanya akan dibantu dengan arsitektur CNN untuk mengklasifikasikan gambar yang ada. Untuk lebih memahami, berikut diagram sederhana bagaimana model CNN ini akan bekerja:



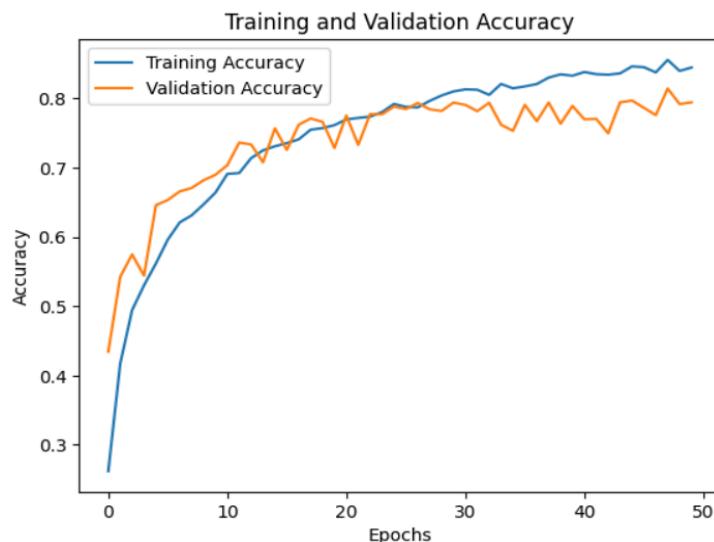
Gambar 4.21 Diagram Arsitektur CNN tanpa Transfer Learning

Diagram diatas menjelaskan bagaimana proses pelatihan model dengan arsitektur CNN saja. Dimulai dengan lapisan *input* sesuai dengan ukuran, kemudian masuk ke lapisan konvolusi dengan kernel, kemudian masuk ke lapisan *pooling* dengan ukuran diatas. Kemudian diulangi hingga sampai ke lapisan terhubung penuh dimana akan dilakukan *flatten* untuk membuat *input* menjadi vektor satu dimensi. Pada akhirnya, akan diberikan kepada lapisan *dense* dengan 512 neuron hingga sampai pada bagian keluaran.

```
Epoch 49/50
242/242 [=====] - 141s 585ms/step - loss: 0.4397 - accuracy: 0.8522 - val_loss: 0.8212 - val_accuracy: 0.7816
Epoch 50/50
242/242 [=====] - 154s 639ms/step - loss: 0.4567 - accuracy: 0.8523 - val_loss: 0.6948 - val_accuracy: 0.8091
```

Gambar 4.22 Hasil Model CNN tanpa Transfer Learning

Gambar diatas melampirkan hasil yang didapat dari sebuah model CNN yang tidak *transfer learning*. Dapat dilihat akurasi model mencapai 85% dan val_accuracy 80% serta *loss* yang lebih tinggi dari model dengan *transfer learning*. Hal ini menunjukkan perbedaan yang signifikan apabila model tidak dilengkapi dengan arsitektur *transfer learning*.



Gambar 4.23 Grafik Hasil Model CNN Tanpa Transfer Learning

Gambar diatas melampirkan graph hasil dari model yang hanya menggunakan CNN. dapat disimpulkan berdasarkan grafik diatas, model mengalami *overfitting*.

```
1/1 [=====] - 0s 421ms/step
Image: /content/drive/MyDrive/Datatesting/ac.png
Predicted class: Air Conditioner
Confidence: 65.74578285217285 %

---
1/1 [=====] - 0s 37ms/step
Image: /content/drive/MyDrive/Datatesting/kulkas.jpg
Predicted class: Refrigerator
Confidence: 99.98965263366699 %

---
1/1 [=====] - 0s 26ms/step
Image: /content/drive/MyDrive/Datatesting/lampu.png
Predicted class: Lamp
Confidence: 99.99197721481323 %

---
1/1 [=====] - 0s 18ms/step
Image: /content/drive/MyDrive/Datatesting/laptop.png
Predicted class: Laptop
Confidence: 99.99997615814209 %

---
1/1 [=====] - 0s 17ms/step
Image: /content/drive/MyDrive/Datatesting/wash.png
Predicted class: Washing Machine
Confidence: 47.29777276515961 %

---
1/1 [=====] - 0s 17ms/step
Image: /content/drive/MyDrive/Datatesting/oven.jpg
Predicted class: Oven
Confidence: 99.9748170375824 %
```

Gambar 4.24 Hasil Pengujian Model Tanpa Transfer Learning

Gambar diatas melampirkan hasil pengujian model tanpa *transfer learning*. Dapat dilihat bahwa model memiliki beberapa hasil pengujian yang tidak baik seperti hasil 65% dan 47%. Hal ini tentu saja dipengaruhi oleh keadaan *overfitting* tersebut.

4.5. Faktor Pengaruh Algoritma CNN

Faktor pengaruh algoritma CNN menjelaskan seberapa rumit atau kompleksnya perhitungan yang akan dilakukan CNN dalam mempelajari pola pada gambar. Berikut adalah beberapa faktor yang mempengaruhi hal ini:

1. **Ukuran Model:** Apabila sebuah model CNN memiliki banyak parameter dan lapisan, tentunya akan menambah ukuran dari model. Dikarenakan model yang besar memerlukan perhitungan yang lebih banyak.
2. **Input Data:** Apabila *input data* yang digunakan dalam jumlah besar seperti gambar yang memiliki resolusi tinggi, ini akan menambah waktu perhitungan pada model daripada gambar dengan resolusi sedang atau rendah.
3. **Jumlah Lapisan:** Jika sebuah model CNN memiliki banyak lapisan, maka semakin kompleks model yang akan dihasilkan.
4. **Runtime:** *Runtime* seperti GPU dapat mengurangi waktu perhitungan dan meningkatkan efisiensi algoritma.
5. **Filter Konvolusi:** Filter yang digunakan dapat mempengaruhi algoritma CNN. Jika filter berukuran besar maka akan diperlukan perhitungan yang lebih kompleks pula.
6. **Fungsi Aktivasi:** Menggunakan fungsi aktivasi yang tepat dapat mempengaruhi efisiensi model. Beberapa fungsi aktivasi memerlukan perhitungan yang lebih rumit.
7. **Operasi Pooling:** Dengan operasi *pooling*, dapat dilakukan pengurangan dimensi data yang dapat mengurangi waktu perhitungan pada lapisan.

4.6. Faktor Pengaruh Arsitektur *Transfer Learning*

Faktor pengaruh arsitektur *transfer learning* menjelaskan seberapa rumit atau kompleksnya arsitektur jaringan saraf yang digunakan dalam *transfer learning*. Berikut adalah faktor-faktor tersebut:

1. **Jumlah Lapisan:** Semakin banyak lapisan, maka semakin kompleks pula arsitektur. Hal ini mengapa jumlah lapisan mempengaruhi *transfer learning*. *Pre-trained* model seperti *VGG*, *ResNet*, atau *Inception*, umumnya memiliki lapisan lebih banyak.
2. **Ukuran Model:** Semakin banyak parameter yang digunakan sebuah model maka akan semakin besar ukuran model tersebut.
3. **Resolusi Masukan:** Masukan seperti gambar dengan resolusi tinggi dapat mempengaruhi arsitektur *transfer learning* karena memerlukan waktu lebih lama dalam perhitungan daripada gambar dengan resolusi rendah atau sedang.
4. **Fungsi Aktivasi:** Menggunakan fungsi aktivasi yang tepat dapat mempengaruhi efisiensi model. Beberapa fungsi aktivasi memerlukan perhitungan yang lebih rumit.
5. **Regularisasi:** menggunakan teknik regularisasi, seperti *L2 regularization*, dapat mempengaruhi kompleksitas model. Regularisasi dapat membantu menghindari *overfitting*, tetapi juga menambah kompleksitas dari model tersebut.
6. **Batch Normalisation:** penggunaan *batch normalisation* dapat membantu dalam mempercepat pelatihan model namun dapat meningkatkan kompleksitas pula.
7. **Tugas Target:** Tugas target dapat mempengaruhi kompleksitas model. Seperti tugas mengenali gambar memerlukan perhitungan yang lebih kompleks.

BAB V

PENUTUP

5.1. Kesimpulan

Berikut merupakan kesimpulan menggunakan analisis, perancangan, implementasi dan pengujian sebagai dasar Algoritma CNN dan Arsitektur *Transfer Learning* dalam prediksi konsumsi energi pada alat rumah tangga.

1. Algoritma *Convolutional Neural Network* dan Arsitektur *Transfer Learning* dapat digunakan untuk memprediksi konsumsi energi alat rumah tangga.
2. Menggunakan *transfer learning* dapat mempercepat proses pelatihan dan memperlebar *knowledge* dari model target.
3. Penggunaan Algoritma CNN pada pengenalan gambar dan model sangat dipengaruhi oleh jumlah data yang digunakan pada pelatihan dan validasi untuk mendapatkan model yang stabil.
4. Jumlah lapisan, fungsi aktivasi, ukuran gambar, filter, dan *runtime* dapat mempengaruhi kompleksitas model.
5. Akurasi yang dimiliki model mencapai 97.61%.
6. Akurasi model dalam memprediksi gambar baru mencapai 99.98%.
7. Prediksi konsumsi energi dapat menjadi acuan bagi pengguna dalam menggunakan alat rumah tangga dan menjaga lingkungan sekitar.

5.2. Saran

Berikut merupakan rekomendasi untuk dipertimbangkan demi penelitian lanjutan:

1. Membuat fitur *filter* jenis alat rumah tangga berdasarkan tipe alat rumah tangga dan kegunaannya.
2. Menambahkan fitur dimana pengguna dapat memasukkan foto yang didalamnya terdapat lebih dari 1 alat dan dijumlahkan penggunaan energi dari alat-alat tersebut.
3. Menambahkan fitur *history* dimana pengguna dapat melihat kembali alat-alat yang mungkin sudah dilakukan prediksi untuk menghemat waktu.
4. Diharapkan kedepannya sistem dapat dijalankan pada perangkat seluler agar lebih mudah diakses oleh pengguna.
5. Diharapkan kedepannya dapat bekerja sama dengan instansi atau perusahaan yang berkaitan untuk memperkuat sistem dan dapat berguna bagi masyarakat

DAFTAR PUSTAKA

- A. González-Briones, G. Hernández, T. Pinto, Z. Vale and J. M. Corchado, "A Review of the Main Machine Learning Methods for Predicting Residential Energy Consumption," 2019 16th International Conference on the European Energy Market (EEM), Ljubljana, Slovenia, 2019, pp. 1-6, doi: 10.1109/EEM.2019.8916406.
- Banik, R., Das, P., Ray, S. et al. Prediction of electrical energy consumption based on machine learning technique. *Electr Eng* 103, 909–920 (2021). <https://doi.org/10.1007/s00202-020-01126-z>
- Bi, Q., Goodman, K. E., Kaminsky, J., & Lessler, J. (2019). What is Machine Learning? A Primer for the Epidemiologist. *American Journal of Epidemiology*, 188(12), 2222–2239. <https://doi.org/10.1093/aje/kwz189>
- Bourhname, S., Abid, M.R., Lghoul, R. et al. Machine learning for energy consumption prediction and scheduling in smart buildings. *SN Appl. Sci.* 2, 297 (2020). <https://doi.org/10.1007/s42452-020-2024-9>
- F. Ciancetta, G. Bucci, E. Fiorucci, S. Mari and A. Fioravanti, "A New Convolutional Neural Network-Based System for NILM Applications," in IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1-12, 2021, Art no. 1501112, doi: 10.1109/TIM.2020.3035193.
- F. Zhuang et al., "A Comprehensive Survey on Transfer Learning," in Proceedings of the IEEE, vol. 109, no. 1, pp. 43-76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.
- K. Yan, W. Li, Z. Ji, M. Qi and Y. Du, "A Hybrid LSTM Neural Network for Energy Consumption Forecasting of Individual Households," in IEEE Access, vol. 7, pp. 157633-157642, 2019, doi: 10.1109/ACCESS.2019.2949065.
- Kim, T.-Y., & Cho, S.-B. (2019). Predicting residential energy consumption using CNN-LSTM neural networks. *Energy*, 182, 72-81. <https://doi.org/10.1016/j.energy.2019.05.230>.

- Le, T.; Vo, M.T.; Vo, B.; Hwang, E.; Rho, S.; Baik, S.W. Improving Electric Energy Consumption Prediction Using CNN and Bi-LSTM. *Appl. Sci.* 2019, 9, 4237. <https://doi.org/10.3390/app9204237>
- Lima, F., Nonogaki, L. K., Chang, J., & Massote, A. A. (2022, August). Estimation of Energy Consumption in Manufacturing Lines Using Machine Learning into Industry 4.0 Context. In 2022 Portland International Conference on Management of Engineering and Technology (PICMET) (pp. 1-7). IEEE.
- Morariu, C., Morariu, O., Răileanu, S., & Borangiu, T. (2020). Machine learning for predictive scheduling and resource allocation in large scale manufacturing systems. *Computers in Industry*, 120, 103244. <https://doi.org/10.1016/j.compind.2020.103244>.
- Mosavi, A., & Bahmani, A. (2019). Energy Consumption Prediction Using Machine Learning; A Review. Preprints. <https://doi.org/10.20944/preprints201903.0131.v1>
- Naranjo-Torres, J., Mora, M., Hernández-García, R., Barrientos, R. J., Fredes, C., & Valenzuela, A. (2020). A Review of Convolutional Neural Network Applied to Fruit Image Processing. *Applied Sciences*, 10(10), 3443. MDPI AG. Retrieved from <http://dx.doi.org/10.3390/app10103443>
- Noman Khan, Ijaz Ul Haq, Samee Ullah Khan, Seungmin Rho, Mi Young Lee, Sung Wook Baik, DB-Net: A novel dilated CNN based multi-step forecasting model for power consumption in integrated local energy systems, International Journal of Electrical Power & Energy Systems,,<https://doi.org/10.1016/j.ijepes.2021.107023>.
- Olu-Ajayi, R., Alaka, H., Sulaimon, I., Sunmola, F., & Ajayi, S. (2022). Building energy consumption prediction for residential buildings using deep learning and other machine learning techniques. *Journal of Building Engineering*, 45, 103406. <https://doi.org/10.1016/j.jobe.2021.103406>.
- Sedlmeir, J., Buhl, H. U., Fridgen, G., & Keller, R. (2021). Recent Developments in Blockchain Technology and their Impact on Energy

Consumption.

Informatik

Spektrum

<https://doi.org/10.48550/arXiv.2102.07886>

Sugimoto, G. (2019). Introduction to Populating a Website with API Data.

DOI:10.46430/phen0086

LAMPIRAN
LISTING PROGRAM

App.py

```
from flask import Flask, request, jsonify
from flask_mysqldb import MySQL
from flask_cors import CORS
from dotenv import load_dotenv
import os
load_dotenv()

app = Flask(__name__)
cors = CORS(app, origins="*")

UPLOAD_FOLDER = 'uploads'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

app.config['MYSQL_HOST'] = str(os.getenv("DB_HOST"))
app.config['MYSQL_USER'] = str(os.getenv("DB_USER"))
app.config['MYSQL_PASSWORD'] = str(os.getenv("DB_PASS"))
app.config['MYSQL_DB'] = str(os.getenv("DB_NAME"))
app.config['MYSQL_PORT'] = 3306

mysql = MySQL(app)

# router blueprints import statement
from router.predict import predict_bp
from router.check_db import check_db_bp
from router.devices import devices_bp
```

```
from router.form import form_bp
from router.test_route import test_route_bp

blueprints = [
    predict_bp,
    check_db_bp,
    devices_bp,
    form_bp,
    test_route_bp,
]

# blueprint register
for blueprint in blueprints:
    app.register_blueprint(blueprint)

if __name__ == "__main__":
    app.run(port=9000, debug=True)
```

Predict.py

```
predict_bp = Blueprint("predict",__name__)

@predict_bp.route('/predict',methods = ['POST'])
def predict():
    try:
```

```
from app import app, mysql

model_path = './model/model_v1.3.h5'

model = tf.keras.models.load_model(model_path, compile=False)

class_labels = ['Air Conditioner', 'Hair Dryer', 'Iron', 'Lamp', 'Laptop',
'Oven', 'Refrigerator', 'Rice Cooker', 'Television', 'Vacuum Cleaner', 'Washing
Machine']

if 'file' not in request.files:
    return jsonify({
        "error": "No file part in the request."
    })

file = request.files['file']

if file.filename == "":
    return jsonify({
        "error": "No file selected."
    })

if not allowed_file(file.filename):
    return jsonify({
        "error": "Invalid file type. Only PNG and JPEG images are allowed."
    })

filename = secure_filename(file.filename)
```

```

file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))

# Load the uploaded image and preprocess it
img = Image.open(os.path.join(app.config['UPLOAD_FOLDER'],
filename))

img = img.resize((224, 224))
img_array = np.array(img)
img_array = img_array / 255.0
img_array = np.expand_dims(img_array, axis=0)

# Make predictions
predictions = model.predict(img_array)
predicted_class_index = np.argmax(predictions[0])
predicted_class = class_labels[predicted_class_index]
confidence = predictions[0][predicted_class_index] * 100
result = "This image most likely belongs to {} with a {:.2f} percent
confidence.".format(predicted_class, confidence)

# Perform database operations
cursor = mysql.connection.cursor()
# Query the database based on the prediction result
sql_query = "SELECT * FROM Main WHERE Object_Name = %s"
cursor.execute(sql_query, (predicted_class,)) # Pass the predicted class as a
parameter
item = cursor.fetchone() # Retrieve the first matching item

if item:
    item_info = {

```

```

        "name": item[0],
        "Image": item[1],
        "Dampak Produksi": item[7],
        "Dampak Konsumsi": item[8],
        "Dampak Disposal": item[9],
        "Lokasi": item[10],
        "Average Energy": item[6],
        "Sumber": item[11],
        "recommendations": get_recommendations(predicted_class) # Add
    recommendations based on the predicted class
    # Add more fields as needed
}

return jsonify({
    "result": result,
    "item_info": item_info,
    "time": current_time
})
else:
    return jsonify({
        "result": result,
        "item_info": None,
        "time": current_time
})

except Exception as e:
    return jsonify({
        "error": str(e)
})

```

```

def allowed_file(filename):
    # Define allowed extensions for image files
    ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg'}
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in
ALLOWED_EXTENSIONS

def get_recommendations(predicted_class):
    if predicted_class == 'Air Conditioner':
        return "R-38 AC"
    elif predicted_class == 'Oven':
        return "Less Than or Equal to 1 Cubic Ft Oven"
    elif predicted_class == 'Lamp':
        return "E17"
    else:
        return "No recommendations at this time"

```

Form.py

```

from flask import Flask,Blueprint,jsonify,request

form_bp = Blueprint('form',__name__)

@form_bp.route('/form',methods = ['POST'])
def form():
    try:
        data = request.get_json() # Assuming the user sends a JSON payload with
        the selected location and device
        location = data.get("location")
        device = data.get("device")
        device_type = data.get("device_type")
    
```

```
if location and device and device_type:
    # Perform database operations to retrieve energy usage based on the
    selected location, device, and device type
    energy_usage = calculate_energy_usage(location, device, device_type)

    return jsonify({
        "energy_usage": energy_usage
    })

else:
    return jsonify({
        "error": "Incomplete data. Please provide the location, device, and
device_type."
    })

except Exception as e:
    return jsonify({
        "error": str(e)
    })

def calculate_energy_usage(location, device, device_type):
    try:
        # Query to retrieve the object name based on the selected location
        from app import mysql
        cursor = mysql.connection.cursor()
        sql_query = "SELECT Object_Name FROM Main WHERE lokasi = %s"
        cursor.execute(sql_query, (location,))

        object_names = cursor.fetchall()
```

```

if object_names:
    object_name = object_names[0][0]

    # Query to retrieve the detailed type based on the selected device
    sql_query = "SELECT detailed_Type.Type_Desc FROM detailed_Type
INNER JOIN Type ON detailed_Type.Type_ID = Type.Type_ID INNER JOIN
Main ON Type.Object_Name = Main.Object_Name WHERE
Main.Object_Name = %s"
    cursor.execute(sql_query, (object_name,))
    detailed_types = cursor.fetchall()

    if detailed_types:
        detailed_type = detailed_types[0][0]

        # Query to retrieve the energy consumption based on the selected
        # detailed type
        sql_query = "SELECT Energy.Energy_Consumed FROM Energy
INNER JOIN detailed_Type ON Energy.Type_ID = detailed_Type.Type_ID
WHERE detailed_Type.Type_Desc = %s"
        cursor.execute(sql_query, (detailed_type,))
        energy_consumed = cursor.fetchall()

        if energy_consumed:
            energy_usage = energy_consumed[0][0]
            return {"location": location, "device": device, "device_type":
device_type, "energy_usage": energy_usage}
        else:
            return {"error": "Energy consumption not found for the selected
detailed type."}

    else:

```

```

        return {"error": "Detailed type not found for the selected device."}

    else:
        return {"error": "Object name not found for the selected location."}

except Exception as e:
    return {"error": str(e)}

```

Model.ipynb

```

train_data_directory = '/content/drive/MyDrive/Dataset/train'
validation_data_directory = '/content/drive/MyDrive/Dataset/validation'
batch_size = 32

```

```

train_datagen = ImageDataGenerator(
    rescale=1.0/255.,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

```

```
val_datagen = ImageDataGenerator(rescale=1.0/255.)
```

```

# Prepare the training and validation data with labels
train_generator = train_datagen.flow_from_directory(
    train_data_directory,
    target_size=(224, 224),

```

```
batch_size=batch_size,
class_mode='categorical'
)

validation_generator = val_datagen.flow_from_directory(
    validation_data_directory,
    target_size=(224, 224),
    batch_size=batch_size,
    class_mode='categorical'
)

# DenseNet load
base_model      =      DenseNet121(include_top=False,      weights='imagenet',
input_shape=(224, 224, 3))

base_model.trainable = False

model = tf.keras.Sequential([
    base_model,
    tf.keras.layers.GlobalAveragePooling2D(),
    Dense(11, activation='softmax')
])

# Compiling
model.compile(optimizer='adam',                  loss='categorical_crossentropy',
metrics=['accuracy'])

# Training by using the generators for initial training
history = model.fit(
```

```
train_generator,  
epochs=50,  
validation_data=validation_generator  
)  
  
# Save the model in the Keras HDF5 format  
model.save("model/accDenseNet v1.3/model v1.3.h5")  
  
# Define the labels  
class_labels = ['Air Conditioner', 'Hair Dryer', 'Iron', 'Lamp','Laptop', 'Oven',  
'Refrigerator', 'Rice Cooker',  
'Television', 'Vacuum Cleaner', 'Washing Machine']  
  
# List of image paths  
image_paths = [  
    '/content/drive/MyDrive/Datatesting/ac.png',  
    '/content/drive/MyDrive/Datatesting/kulkas.jpg',  
    '/content/drive/MyDrive/Datatesting/lampu.png',  
    '/content/drive/MyDrive/Datatesting/laptop.png',  
    '/content/drive/MyDrive/Datatesting/wash.png',  
    '/content/drive/MyDrive/Datatesting/oven.jpg',  
    '/content/drive/MyDrive/Datatesting/penanak_nasi.png',  
    '/content/drive/MyDrive/Datatesting/pengering_rambut.png',  
    '/content/drive/MyDrive/Datatesting/setrika.png',  
    '/content/drive/MyDrive/Datatesting/tv.jpg',
```

```
'/content/drive/MyDrive/Datatesting/vac.jpg'  
]  
  
model = tf.keras.models.load_model('model/accDenseNet v1.3/model v1.3.h5')  
# Replace with the path to your trained model  
  
  
for image_path in image_paths:  
    # Load the image  
    img = image.load_img(image_path, target_size=(224, 224))  
    img_array = image.img_to_array(img)  
    img_array = np.expand_dims(img_array, axis=0)  
  
    # Preprocess the image  
    img_array = img_array / 255.0  
  
    # predict  
    predictions = model.predict(img_array)  
    predicted_class_index = np.argmax(predictions[0])  
    predicted_class = class_labels[predicted_class_index]  
    confidence = predictions[0][predicted_class_index] * 100
```

```
# Confidence in percentage  
# Print the predicted class and confidence  
print('Image:', image_path)  
print('Predicted class:', predicted_class)  
print('Confidence:', confidence, '%')  
print('---')
```