KLASIFIKASI SPESIES LEBAH MADU TANPA SENGAT MENGGUNAKAN MODEL *DEEP LEARNING INCEPTION-RESNETV2* DI LEMBAH SUHITA

Skripsi

Oleh

ALFI JULIAN AZHARI
NPM 2115061021

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2026

KLASIFIKASI SPESIES LEBAH MADU TANPA SENGAT MENGGUNAKAN MODEL *DEEP LEARNING INCEPTION-RESNETV2* DI LEMBAH SUHITA

Oleh

ALFI JULIAN AZHARI

Skripsi

Diajukan untuk Memenuhi Kelulusan Gelar SARJANA TEKNIK
pada
Program Studi Teknik Informatika
Jurusan Teknik Elektro
Fakultas Teknik

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2026

ABSTRAK

KLASIFIKASI SPESIES LEBAH MADU TANPA SENGAT MENGGUNAKAN MODEL *DEEP LEARNING INCEPTION-RESNETV2* DI LEMBAH SUHITA

Oleh

ALFI JULIAN AZHARI

Lebah madu tanpa sengat (Meliponinae) berperan penting sebagai penyerbuk dalam menjaga keseimbangan ekosistem dan produktivitas pertanian, tetapi identifikasi spesies masih sulit dilakukan karena ukuran tubuh kecil dan kemiripan morfologi antarspesies. Penelitian ini bertujuan mengembangkan sistem klasifikasi spesies berbasis citra menggunakan arsitektur *deep learning Inception-ResNetV2*. Dataset terdiri atas 1.280 citra dari empat spesies lebah tanpa sengat, yaitu *Heterotrigona itama, Tetrigona apicalis, Tetrigona binghami,* dan *Tetrigona vidua*, yang diperoleh melalui pengambilan gambar langsung di Lembah Suhita. Model dikembangkan menggunakan pendekatan OSEMN dengan transfer learning melalui *feature extraction*, *fine tuning*, dan augmentasi citra. Model yang dihasilkan memperoleh akurasi sebesar 97,92% pada hasil pelatihan dan 92,5% pada pengujian. Model kemudian diimplementasikan dalam aplikasi berbasis web dengan mekanisme *energy based* untuk mengidentifikasi spesies sekaligus menolak citra yang tidak dikenali sistem. Pengujian implementasi model menunjukkan tingkat ketepatan hasil klasifikasi sebesar 70%.

Kata kunci: Lebah Madu Tanpa Sengat, Klasifikasi Citra, *Deep Learning*, *Inception-ResNetV2*, Transfer Learning

ABSTRACT

CLASSIFICATION OF STINGLESS BEE SPECIES USING THE *INCEPTION-RESNETV2 DEEP LEARNING* MODEL IN LEMBAH SUHITA

By

ALFI JULIAN AZHARI

Stingless bees (Meliponinae) play an important role as pollinators in maintaining ecosystem balance and agricultural productivity. However, species identification remains challenging due to their small body size and morphological similarities among species. This study aims to develop an image-based species classification system using the Inception-ResNetV2 deep learning architecture. The dataset consisted of 1,280 images from four stingless bee species, namely *Heterotrigona itama, Tetrigona apicalis, Tetrigona binghami,* and *Tetrigona vidua*, obtained through direct image acquisition at Lembah Suhita. The model was developed using the OSEMN approach with transfer learning through feature extraction, fine-tuning, and image augmentation. The resulting model achieved an accuracy of 97.92% during training and 92.5% during testing. The model was then implemented in a web-based application with an energy-based mechanism to identify species while rejecting images unrecognized by the system. The implementation testing showed a classification accuracy rate of 70%.

***Keywords**: Stingless Bee, Image Classification, Deep Learning, Inception-ResNetV2, Transfer Learning*

Judul Skripsi	:	**KLASIFIKASI SPESIES LEBAH MADU TANPA SENGAT MENGGUNAKAN MODEL *DEEP LEARNING INCEPTION-RESNETV2* DI LEMBAH SUHITA**
Nama Mahasiswa	:	Alfi Julian Azhari
Nomor Pokok Mahasiswa	:	**2115061021**
Jurusan	:	**Teknik Informatika**
Fakultas	:	**Teknik**

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Yessi Mulyani, S.T., M.T.	**Rio Ariestia Pradipta, S.Kom., M.T.I.**
NIP. 197312262000122001	**NIP. 198603232019031013**

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T., M.T.	**Yessi Mulyani, S.T., M.T.**
NIP. 197103141999032001	**NIP. 197312262000122001**

MENGESAHKAN

1. Tim Penguji

   Ketua : **Yessi Mulyani, S.T., M.T.**

   Sekretaris : **Rio Ariestia Pradipta, S.Kom., M.T.I.**

   Penguji : **Dr. Eng. Ir. Mardiana, S.T., M.T., IPM.**

2. Dekan Fakultas Teknik

Dr. Ahmad Herison, S.T., M.T.
NIP. 196910302000031001

Tanggal Lulus Ujian Skripsi : **06 Mei 2026**

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, menyatakan bahwa skripsi saya dengan judul "Klasifikasi Spesies Lebah Madu Tanpa Sengat Menggunakan Model *Deep Learning Inception-Resnetv2* Di Lembah Suhita" dibuat oleh saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 12 Juni 2026

Pembuat Pernyataan

Alfi Julian Azhari

NPM.2115061021

RIWAYAT HIDUP

Penulis lahir di Sukoharjo pada tanggal 3 Juli 2002. Penulis menyelesaikan pendidikan formal di SD Negeri 3 Sukoharjo pada tahun 2015, SMP Negeri 1 Sukoharjo pada tahun 2018, dan SMA Negeri 1 Pringsewu pada tahun 2021. Pada tahun 2021, penulis terdaftar sebagai mahasiswa Program Studi Teknik Informatika Universitas Lampung melalui jalur SBMPTN. Pada akhir tahun 2024, penulis melaksanakan Kuliah Kerja Nyata (KKN) Periode I di Pekon Sukaraja, Kecamatan Gunung Alip, Kabupaten Tanggamus selama kurang lebih 40 hari. Kegiatan tersebut memberikan pengalaman dalam pengabdian kepada masyarakat serta penerapan ilmu pengetahuan untuk mendukung pembangunan desa. Penulis juga melaksanakan kerja praktik (Magang) di Skypillar Studio, Bandar Lampung, yang memberikan pengalaman dan keterampilan baru, khususnya di bidang pengembangan game digital. Adapun organisasi, kegiatan, dan prestasi yang pernah diraih oleh penulis antara lain sebagai berikut:

1. Meraih juara lomba esai dan poster tingkat nasional sebanyak dua kali.
2. Mengikuti program Studi Independen MBKM Batch 2 tahun 2023 pada program *The President 3 Joined Project UI/UX Creating Digital Solutions & Transformation for Sustainable Agriculture Management Mobile Apps*.
3. Menjadi anggota aktif Himpunan Mahasiswa Jurusan Teknik Elektro Universitas Lampung (HIMATRO UNILA) Departemen Sosial dan Kewirausahaan Divisi Kewirausahaan periode 2022/2023 dan 2023/2024.

MOTTO

“When I was a boy, I was blinded by a chemical spill. It made me blind, but it also gave me a kind of sight.”

(Matt Murdock)

“Adab lebih tinggi daripada ilmu.”

(Imam Malik)

“Apa yang tidak Allah berikan hari ini, mungkin sedang digantikan dengan sesuatu yang lebih baik.”

PERSEMBAHAN

Bismillahirrohmanirrohim, segala puji bagi Allah SWT. Tuhan Yang Maha Esa, karena atas nikmat dan karunia-Nya, Saya dapat menyelesaikan skripsi ini.

Kupersembahkan Skripsi ini Kepada:

"Untuk diri sendiri yang telah berjuang dan bertahan sampai di titik ini. Rasa syukur dipanjatkan karena mampu melewati setiap proses, tantangan, dan hambatan selama penyusunan skripsi ini dengan penuh kesabaran dan kegigihan. Terima kasih karena tetap berusaha, tidak menyerah, serta terus memberikan waktu, tenaga, dan pikiran hingga tugas ini dapat diselesaikan. Semoga segala usaha yang telah dilakukan menjadi langkah baik untuk mencapai tujuan dan cita-cita di masa mendatang. Untuk ayah, ibu, adik, nenek, dan seluruh keluarga yang selalu memberikan doa, dukungan, serta semangat selama proses penyusunan skripsi ini. Terima kasih atas perhatian, pengorbanan, dan kepercayaan yang selalu diberikan dalam setiap langkah perjalanan ini. Kehadiran dan dukungan keluarga menjadi salah satu alasan untuk terus bertahan dan menyelesaikan tanggung jawab ini dengan baik. Semoga segala kebaikan yang diberikan selalu mendapat balasan dan keberkahan."

SANWACANA

Puji syukur kehadirat Allah SWT atas segala rahmat, karunia, dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Salawat serta salam semoga selalu tercurah kepada Nabi Muhammad SAW yang menjadi teladan bagi umat manusia.

Skripsi yang berjudul **"Klasifikasi Spesies Lebah Madu Tanpa Sengat Menggunakan Model *Deep Learning Inception-Resnetv2* Di Lembah Suhita"** disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Informatika, Universitas Lampung. Dalam proses penyusunan skripsi ini, penulis memperoleh banyak bantuan, dukungan, dan bimbingan dari berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Kedua orang tua, adik, nenek, dan seluruh keluarga yang selalu memberikan doa, dukungan, perhatian, serta semangat kepada penulis selama proses perkuliahan dan penyusunan skripsi ini.
2. Bapak Dr. Ahmad Herison, S.T., M.T., selaku Dekan Fakultas Teknik Universitas Lampung.
3. Ibu Herlinawati, S.T., M.T., selaku Ketua Jurusan Teknik Elektro Universitas Lampung.
4. Ibu Yessi Mulyani, S.T., M.T., selaku pembimbing utama yang telah memberikan bimbingan, arahan, masukan, serta dukungan kepada penulis selama proses penyusunan skripsi ini.
5. Bapak Rio Ariestia Pradipta, S.Kom., M.T.I., selaku pembimbing pendamping yang telah membantu dan memberikan banyak saran, masukan, serta motivasi dalam penyelesaian penelitian ini.

6. Ibu Dr. Eng. Ir. Mardiana, S.T., M.T., IPM., selaku penguji yang telah memberikan saran dan masukan yang membangun demi penyempurnaan skripsi ini.
7. Teman-teman seperjuangan livingbrg yang telah memberikan kebersamaan, dukungan, serta pengalaman selama masa perkuliahan, karena tanpa ada kalian perkuliahan akan kurang berwarna.
8. Semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu dan mendukung penulis dalam menyelesaikan skripsi ini.

Penulis menyadari bahwa skripsi ini masih memiliki kekurangan baik dari segi penulisan maupun pengembangan sistem. Oleh karena itu, penulis terbuka terhadap kritik dan saran yang membangun demi perbaikan di masa mendatang. Semoga skripsi ini dapat memberikan manfaat bagi pembaca dan menjadi salah satu kontribusi dalam pengembangan teknologi informasi, khususnya di bidang klasifikasi citra.

Bandar Lampung, 12 Juni 2026

Alfi Julian Azhari
NPM.2115061021

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

I. PENDAHULUAN

1.1 Latar Belakang

Lebah madu tanpa sengat *(Meliponinae)* merupakan jenis lebah yang sering kali luput dari perhatian masyarakat umum. Dibandingkan dengan lebah madu biasa (*Apis mellifera*), keberadaan lebah tanpa sengat ini tidak begitu mencolok karena ukurannya yang kecil dan perilakunya yang relatif tenang [1]. Di Indonesia sendiri, lebah madu tanpa sengat dikenal dengan berbagai nama lokal, seperti klanceng (Jawa), kelulut (Sumatera dan Kalimantan), galo-galo (Sulawesi), serta teuweul atau odeng (Sunda, Jawa Barat)[2]. Sekitar 75% tanaman pangan dunia bergantung pada aktivitas penyerbukan, dan sebagian besar proses tersebut dilakukan oleh serangga, termasuk lebah[3]. Di wilayah tropis seperti Indonesia, lebah tanpa sengat memiliki peran utama karena ketahanannya terhadap iklim lembap dan kemampuannya untuk menjangkau berbagai jenis tanaman lokal. Tanpa keberadaan mereka, proses penyerbukan alami akan terganggu, yang berpotensi menurunkan produktivitas pertanian dan mengancam keberlangsungan ekosistem.

Indonesia memiliki keanekaragaman spesies lebah tanpa sengat yang sangat kaya, dengan lebih dari 46 spesies telah diidentifikasi sejauh ini dan diyakini masih banyak spesies lain yang belum terdokumentasikan[4]. Salah satu tempat yang menjadi habitat lebah tanpa sengat adalah Lembah Suhita, sebuah tempat peternakan lebah dengan ekosistem yang relatif alami dan mendukung kehidupan berbagai spesies serangga penyerbuk, termasuk Meliponini. Namun, tantangan besar muncul dalam upaya pelestarian dan pemanfaatan lebah tanpa sengat, yakni pada proses identifikasi spesiesnya. Spesies-spesies dari suku *Meliponini* memiliki kemiripan morfologi yang sangat tinggi, baik dari segi warna tubuh, ukuran, bentuk kepala, maupun struktur sayap, sehingga sulit dibedakan melalui

pengamatan visual biasa. Di lembah suhita kondisi ini menyebabkan proses identifikasi cukup sulit dengan keterbatasan pengetahuan dan berpotensi menimbulkan ketidaktepatan klasifikasi, yang pada akhirnya dapat memengaruhi efektivitas dokumentasi dan pengelolaan populasi di lapangan.

Kesalahan dalam mengidentifikasi spesies lebah dapat berdampak pada ketidaksesuaian dalam pemberian perawatan atau habitat, yang pada akhirnya menyebabkan kepunahan spesies lebah yang lemah. Padahal, pemahaman yang akurat tentang spesies sangat penting, mengingat setiap spesies memiliki kebutuhan lingkungan, perilaku sosial, dan karakteristik biologis yang berbeda. Oleh karena itu, dibutuhkan metode identifikasi yang cepat, akurat, dan mudah digunakan. Seiring perkembangan teknologi, kecerdasan buatan (*Artificial Intelligence*/AI) menjadi solusi inovatif yang dapat diterapkan untuk mengatasi kendala tersebut.

Salah satu model *deep learning* yang menjanjikan untuk klasifikasi citra adalah Inception-ResNetV2, sebuah arsitektur *Convolutional Neural Network* (CNN) yang menggabungkan keunggulan *Inception* dalam mengenali detail visual dengan *residual learning* dari *ResNet*. Kombinasi ini membuat model mampu mengekstraksi fitur-fitur kompleks dan halus dari citra dengan efisiensi tinggi serta meminimalkan risiko *vanishing gradient* saat pelatihan jaringan yang dalam[5]. Model ini juga mendukung penerapan *transfer learning*, yang memungkinkan model terlatih dari dataset besar seperti *ImageNet* dengan jumlah data terbatas dan kondisi yang sesuai untuk dataset citra lebah dari Lembah Suhita[6].

Model Inception-ResNetV2 dikenal memiliki performa tinggi dalam berbagai tugas klasifikasi citra diantaranya adalah penelitian klasifikasi spesies burung endemik di Taiwan mencatat akurasi 98,39% [7]. Model ini mencapai akurasi sebesar 96,63% pada klasifikasi citra tumor otak[8]. Hasil tersebut menunjukkan kemampuan arsitektur Inception-ResNetV2 dalam mengekstraksi fitur kompleks secara efisien, yang relevan dengan kebutuhan pengenalan morfologi halus pada citra lebah. Namun, model klasifikasi sering menghadapi tantangan ketika

menerima data yang berbeda dari distribusi pelatihan (*out of distribution*). Untuk menjaga keandalan hasil prediksi, penelitian ini menerapkan pendekatan energy-based scoring, yaitu metode yang mendeteksi apakah citra termasuk dalam distribusi pelatihan atau merupakan data asing yang sebaiknya tidak diklasifikasikan[9].

Proses pengembangan dilakukan menggunakan pendekatan OSEMN (*Obtain, Scrub, Explore, Model, and Interpret*) agar seluruh tahapan berjalan sistematis dan menghasilkan model optimal[10]. Sebagai langkah implementasi model yang diperoleh akan diimplementasikan dalam web sederhana agar dapat digunakan secara praktis oleh peternak maupun peneliti di lapangan. Dengan kombinasi arsitektur Inception-ResNetV2, pendekatan OSEMN, dan metode energy-based scoring, sistem yang dikembangkan diharapkan mampu menghasilkan klasifikasi lebah tanpa sengat yang akurat, andal, dan mudah diakses.

1.2 Rumusan Masalah

Adapun rumusan masalah dari penelitian ini adalah sebagai berikut:

1. Bagaimana membangun model klasifikasi citra spesies lebah madu tanpa sengat berbasis Inception-ResNetV2?
2. Bagaimana performa model Inception-ResNetV2 dalam mengklasifikasikan spesies lebah madu tanpa sengat?

1.3 Tujuan Penelitian

Adapun Tujuan dari penelitian ini adalah sebagai berikut:

1. Mengembangkan model klasifikasi citra berbasis Inception-ResNetV2 untuk mengidentifikasi spesies lebah madu tanpa sengat.
2. Menganalisis dan mengevaluasi performa kinerja model Inception-ResNetV2 dalam mengklasifikasikan spesies lebah madu tanpa sengat.

1.4 Manfaat Penelitian

Adapun Manfaat dari penelitian ini adalah sebagai berikut:

1. Manfaat Teoritis

   Penelitian ini diharapkan dapat memberikan kontribusi bagi pengembangan ilmu pengetahuan, khususnya dalam penerapan deep learning dengan arsitektur Inception-ResNetV2 pada bidang klasifikasi citra lebah madu. Hasil penelitian ini dapat dijadikan tahap awal bagi penelitian selanjutnya yang berkaitan dengan klasifikasi spesies lebah.
2. Manfaat Praktis

   Penelitian ini diharapkan dapat membantu dalam perancangan sistem identifikasi dan klasifikasi lebah madu tanpa sengat sebagai solusi yang memudahkan identifikasi spesies lebah madu tanpa sengat di Lembah Suhita dan wilayah tropis sejenis serta mendukung konservasi dan pengelolaan keanekaragaman hayati secara efektif.

1.5 Batasan Penelitian

Untuk menjaga fokus penelitian dan memastikan keterukuran hasil, penelitian ini memiliki beberapa batasan sebagai berikut:

1. Penelitian ini hanya difokuskan pada klasifikasi empat spesies lebah madu tanpa sengat yang berada di kawasan Lembah Suhita, yaitu *Tetrigona apicalis, Tetrigona binghami, Heterotrigona itama, dan Tetrigona vidua*.
2. Data citra lebah yang digunakan berasal dari dokumentasi lapangan di Lembah Suhita, dengan tambahan data dari platform publik seperti iNaturalist dan Flickr, namun tidak menggunakan dataset eksternal lain di luar konteks tersebut.
3. Model Penelitian ini hanya membahas proses klasifikasi citra lebah madu tanpa sengat menggunakan satu pendekatan model deep learning, tanpa melakukan studi komparatif antar arsitektur atau metode klasifikasi lainnya.

1.6 Sistematika Penulisan Skripsi

Dalam skripsi ini sistematika penulisan yang digunakan adalah sebagai berikut:

BAB I : PENDAHULUAN

Memuat latar belakang penyusunan skripsi ini, perumusan masalah, tujuan penelitian, manfaat penelitian dan batasan masalah serta sistematika penulisan dalam penelitian ini.

BAB II : TINJAUAN PUSTAKA

Memuat penjelasan terkait penelitian terkait dan dasar teori yang terkait dengan penelitian ini.

BAB III : METODOLOGI PENELITIAN

Memuat waktu dan tempat penelitian, tahapan penelitian secara umum, dan tahapan penelitian secara terperinci.

BAB IV : PEMBAHASAN

Memuat hasil penelitian dengan berdasarkan pada perumusan masalah serta pembahasan terkait hasil penelitian.

BAB V : KESIMPULAN DAN SARAN

Memuat kesimpulan dari analisa yang dilakukan pada penelitian ini.

II. TINJAUAN PUSTAKA

2.1 Lebah Madu Tanpa Sengat (*Stingless Bee*)

Lebah madu tanpa sengat atau yang dikenal dengan nama lokal, seperti klanceng (Jawa), kelulut (Sumatera dan Kalimantan), galo-galo (Sulawesi), serta teuweul atau odeng (Sunda, Jawa Barat) merupakan jenis lebah dari subfamili *Meliponinae* yang tersebar luas di wilayah tropis dan subtropis, termasuk di Indonesia[2]. Berbeda dari lebah madu biasa (*Apis mellifera*), lebah ini tidak memiliki sengat namun tetap mampu mempertahankan koloninya melalui cara-cara lain, seperti menggigit dan mengeluarkan cairan lengket dari tubuhnya[1]. Mereka dikenal sebagai penyerbuk alami yang sangat efisien serta memiliki kemampuan adaptasi tinggi terhadap lingkungan.

Berdasarkan taksonomi Claus Rasmussen terdapat 129 taksa lebah tanpa sengat (*Meliponini*) yang telah diusulkan di kawasan Indo-Malaya dan Australasia, dengan 89 di antaranya diakui sebagai spesies valid[11]. Temuan tersebut menegaskan tingginya keanekaragaman lebah tanpa sengat di wilayah tropis, termasuk Indonesia. Hasil observasi lapangan di Lembah Suhita di Juli tahun 2025 menunjukkan adanya 12 spesies lebah dari dua kelompok utama, yaitu lebah tanpa sengat (*Meliponini*) dan lebah bersengat (*Apis*). Informasi ini diperoleh dari pengelola kawasan saat kunjungan penelitian.

Spesies yang teridentifikasi:

- Meliponini: *Tetrigona apicalis*, *Heterotrigona itama*, *Tetragonula biroi*, *Geniotrigona thoracica*, *Tetragonula testaceitarsis*, *Tetragonula laeviceps*, *Tetrigona binghami*, *Tetrigona vidua*, *Tetragonula reepeni*
- Apis: *Apis cerana*, *Apis dorsata*, *Apis mellifera*

Dalam penelitian ini, digunakan empat spesies lebah tanpa sengat, yaitu *Heterotrigona itama*, *Tetrigona binghami*, *Tetrigona apicalis*, dan *Tetrigona vidua*.

1. *Heterotrigona itama*

Heterotrigona itama merupakan salah satu spesies lebah tanpa sengat yang paling umum dibudidayakan di Asia Tenggara, termasuk Indonesia dan Malaysia. Spesies ini dikenal memiliki koloni besar dan produktivitas tinggi dalam menghasilkan madu serta propolis. Selain mudah beradaptasi terhadap berbagai kondisi lingkungan tropis, *H. itama* juga memiliki perilaku sosial yang stabil, menjadikannya spesies unggulan dalam kegiatan meliponikultur (budidaya lebah tanpa sengat). Secara morfologis, *H. itama* dapat dikenali dari tubuh berwarna hitam mengilap dengan ukuran sedang dan pola terbang yang relatif tenang[12].

2. *Tetrigona binghami*

Tetrigona binghami merupakan spesies lebah tanpa sengat yang ditemukan di beberapa wilayah Asia Tenggara. Spesies ini memiliki ukuran tubuh relatif kecil dan dikenal membangun sarang di rongga pohon atau batang kayu berlubang. Meskipun tidak seproduktif *Heterotrigona itama*, *T. binghami* berperan penting dalam proses penyerbukan tanaman berbunga di ekosistem hutan tropis. Ciri khas morfologinya adalah warna tubuh kehitaman dengan rambut halus berwarna keabu-abuan pada toraks, serta bentuk sayap yang relatif sempit[12].

3. *Tetrigona apicalis*

Tetrigona apicalis termasuk salah satu spesies *Meliponini* yang banyak ditemukan di wilayah Asia tropis, termasuk Indonesia. Spesies ini dikenal memiliki kemampuan adaptasi yang baik terhadap lingkungan yang lembap dan vegetasi beragam. Koloni *T. apicalis* umumnya bersarang di batang pohon, dan menghasilkan madu dengan aroma kuat serta warna yang lebih gelap dibandingkan *H. itama*. Selain itu, propolis dari *T. apicalis* dilaporkan mengandung senyawa bioaktif dengan potensi antimikroba dan antioksidan tinggi, sehingga bernilai ekonomi dan farmakologis[13].

4. *Tetrigona vidua*

Tetrigona vidua merupakan salah satu spesies lebah tanpa sengat yang tersebar di kawasan Asia Tenggara. Spesies ini dikenal sebagai penyerbuk alami berbagai jenis tanaman hutan dan pertanian. *Tetrigona vidua* membentuk koloni berukuran sedang dan bersarang di rongga alami seperti batang pohon atau dinding kayu. Secara morfologis, tubuhnya berwarna coklat kehitaman dengan permukaan toraks berbulu halus. Walaupun produksinya relatif kecil dibandingkan *Heterotrigona itama*, spesies ini memiliki nilai ekologis penting dalam menjaga keberlanjutan ekosistem lokal[14].

Selain berperan penting dalam proses penyerbukan, lebah madu tanpa sengat juga menghasilkan berbagai produk bernilai ekonomi seperti madu, propolis, dan bee bread, yang memiliki manfaat kesehatan dan potensi pasar yang besar[15]. Keunikan mereka membuat banyak pihak tertarik untuk melakukan budidaya, terutama karena karakteristiknya yang jinak dan dapat dipelihara di sekitar permukiman manusia. Namun demikian, identifikasi dan klasifikasi jenis-jenis *stingless bee* masih menjadi tantangan tersendiri. Morfologi antarspesies sering kali mirip, sehingga diperlukan pendekatan ilmiah dan teknologi yang lebih modern untuk membedakan spesies satu dengan lainnya. Tantangan ini menjadi menarik ketika dikaitkan dengan pengembangan sistem otomatis berbasis teknologi citra digital dan kecerdasan buatan yang mulai banyak diteliti.

2.2 Citra

Citra merupakan representasi visual dari objek atau fenomena yang disajikan dalam bentuk gambar atau data digital[16]. Dalam konteks pengolahan citra, citra digunakan untuk menganalisis objek, pola, dan informasi lainnya. Citra digital pada dasarnya berupa matriks dua dimensi yang terdiri dari piksel dengan nilai intensitas tertentu. Citra ini diperoleh dari berbagai sumber seperti kamera, sensor optik, atau alat penginderaan jauh lainnya[17]. Dalam pengolahan citra untuk deteksi objek, seperti deteksi lebah madu tanpa sengat, citra menjadi sumber

utama informasi yang akan diproses oleh sistem untuk mengidentifikasi dan mengklasifikasikan objek dalam citra tersebut.

Citra digital dapat direpresentasikan dalam tiga bentuk berdasarkan informasi piksel yang dikandungnya, yaitu citra biner, citra grayscale, dan citra berwarna[18]. Citra biner hanya memiliki dua tingkat nilai piksel yang merepresentasikan objek dan latar belakang, sedangkan citra grayscale menyimpan informasi intensitas cahaya dalam satu kanal[18]. Adapun citra berwarna atau RGB mengandung informasi warna pada beberapa kanal, seperti kanal merah (*red*), hijau (*green*), dan biru (*blue*) pada model RGB, sehingga mampu merepresentasikan informasi visual yang lebih kaya dibandingkan citra grayscale maupun citra biner[18]. Pada deteksi objek, citra berwarna lebih sering digunakan karena dapat menangkap berbagai variasi warna yang lebih kompleks dari objek yang ada, seperti perbedaan warna tubuh dan sayap lebah madu tanpa sengat. Namun, citra grayscale juga memiliki keunggulan dalam hal kesederhanaan dan efisiensi dalam pemrosesan, terutama ketika objek yang dikenali memiliki perbedaan bentuk yang jelas[19].

2.2.1 Klasifikasi Citra

Klasifikasi citra adalah proses pengelompokan citra ke dalam kategori atau kelas yang telah ditentukan sebelumnya. Dalam deteksi objek, ini berarti mengidentifikasi objek yang ada dalam citra dan mengklasifikasikannya ke dalam kelas tertentu berdasarkan fitur visualnya. Proses ini melibatkan pengambilan informasi dari citra, yang dikenal dengan istilah fitur citra, dan menggunakan fitur tersebut untuk memutuskan kategori objek[20]. Metode yang digunakan untuk klasifikasi citra dapat bervariasi. Salah satu metode yang sering digunakan adalah pembelajaran mesin, di mana algoritma seperti jaringan saraf konvolusional (CNN), *Support Vector Machine* (SVM), dan *k-Nearest* Neighbor (k-NN) dapat digunakan untuk mengenali objek berdasarkan fitur yang diekstraksi dari citra. Pembelajaran mendalam, dengan menggunakan jaringan saraf yang lebih

kompleks, juga sangat populer dalam klasifikasi citra modern, karena kemampuannya untuk menangkap pola yang lebih rumit dalam data gambar[21].

2.2.2 Pra-pemrosesan Citra (*Image Preprocessing*)

Pra-pemrosesan citra merupakan tahap awal yang sangat penting sebelum melakukan analisis lebih lanjut, terutama dalam deteksi dan klasifikasi objek[22]. Citra yang diperoleh dari kamera atau perangkat pengambil gambar lainnya seringkali memiliki gangguan seperti noise (derau), pencahayaan yang tidak merata, dan distorsi lainnya yang dapat mengurangi kualitas data yang digunakan oleh model.

Langkah dalam pra-pemrosesan citra adalah sebagai berikut:

1. Konversi ke *Grayscale*

Untuk menyederhanakan gambar, citra sering kali diubah ke format *grayscale* (hitam putih), di mana informasi warna diabaikan[23]. Hal ini memperkecil jumlah informasi yang harus diproses, sehingga mempercepat analisis dan mengurangi kemungkinan gangguan dari variasi warna yang tidak relevan.

2. Normalisasi Ukuran

Citra yang diambil mungkin memiliki ukuran yang berbeda-beda. Untuk mempermudah pemrosesan oleh jaringan saraf, gambar-gambar tersebut harus dinormalisasi ke ukuran yang seragam[24]. Misalnya, mengubah semua gambar menjadi ukuran 224 x 224 piksel.

3. Pengurangan Noise

Noise adalah gangguan visual pada citra, seperti bintik-bintik yang tidak diinginkan atau perubahan warna yang tiba-tiba. Filter seperti *Gaussian blur* dapat digunakan untuk mengurangi noise dan membuat fitur yang relevan lebih terlihat.

4. Penyesuaian Pencahayaan

Variasi dalam pencahayaan bisa membuat objek dalam gambar sulit dikenali. Oleh karena itu, teknik untuk menyesuaikan kontras atau mengubah tingkat kecerahan pada citra dapat membantu meningkatkan kualitas visual dan

memudahkan analisis. Pra-pemrosesan sangat penting untuk mendapatkan hasil yang akurat dan optimal dalam deteksi atau klasifikasi citra [25].

2.2.3 Ekstraksi Fitur (*Feature Extraction*)

Ekstraksi fitur adalah tahap di mana informasi penting yang terkandung dalam citra diekstraksi untuk digunakan dalam analisis lebih lanjut. Fitur yang diekstraksi dari citra bisa berupa bentuk objek, tekstur, warna, atau bahkan pola geometris[17]. Ekstraksi fitur ini penting dalam deteksi objek, karena fitur-fitur ini digunakan oleh algoritma untuk membedakan satu objek dengan objek lainnya. Teknik ekstraksi fitur meliputi :

1. *Edge Detection*

Proses mendeteksi perubahan tajam dalam intensitas gambar yang biasanya menandakan batas objek. Metode seperti Canny edge detector sangat umum digunakan untuk mendeteksi kontur objek dalam citra.

2. *Histogram of Oriented Gradients* (HOG)

Teknik ini menangkap informasi orientasi gradien di citra dan berguna untuk mendeteksi bentuk objek.

3. SIFT (*Scale-Invariant Feature Transform*)

Mengidentifikasi titik-titik fitur yang stabil terhadap perubahan skala dan rotasi, sangat berguna dalam aplikasi di mana objek dapat terlihat dengan berbagai ukuran atau sudut pandang.

Dengan ekstraksi fitur keberhasilan deteksi objek sangat bergantung pada kualitas fitur yang diekstraksi, karena fitur yang buruk akan menyebabkan deteksi yang kurang akurat[26].

2.2.4 Anotasi Citra (*Image Annotation*)

Anotasi citra adalah proses pemberian label atau informasi tambahan pada citra untuk menunjukkan objek yang terdapat dalam gambar[22]. Pada deteksi objek, anotasi citra umumnya melibatkan penandaan posisi objek dengan menggunakan *bounding boxes* atau teknik segmentasi untuk memisahkan objek dari latar belakang. Teknik anotasi yang umum digunakan dalam pengolahan citra adalah:

1. *Bounding Boxes*

Menandai objek dengan menggambar kotak persegi panjang di sekeliling objek yang ingin dikenali. Misalnya, untuk menandai lebah madu tanpa sengat dalam gambar, kita dapat menggambar kotak di sekitar tubuh lebah.

2. *Segmentation*

Memisahkan objek dengan latar belakang menggunakan teknik segmentasi gambar, yang memberi informasi lebih detail tentang bentuk objek.

3. *Keypoint Annotation*

Menandai titik-titik kunci pada objek, seperti titik-titik pada sayap lebah untuk meningkatkan akurasi deteksi.

Anotasi yang tepat sangat penting dalam melatih model deteksi objek, karena model akan belajar untuk mengenali objek berdasarkan data yang telah diberi label. Tanpa anotasi yang akurat, model mungkin akan kesulitan dalam mempelajari pola yang benar[24]. Anotasi yang dilakukan secara tepat dan konsisten berkontribusi besar terhadap keberhasilan model dalam mengenali objek secara akurat.

2.2.5 Augmentasi Citra (*Image Augmentation*)

Augmentasi citra adalah teknik untuk meningkatkan jumlah data pelatihan dengan cara menghasilkan variasi dari gambar yang sudah ada[27]. Hal ini sangat penting untuk mencegah *overfitting*, di mana model hanya belajar dari gambar yang

terbatas dan gagal mengenali objek baru dengan baik. Beberapa teknik augmentasi citra yang umum digunakan antara lain:

1. Rotasi, Memutar gambar dengan sudut tertentu untuk mengenalkan variasi posisi objek.
2. *Flipping*, Membalik gambar secara horizontal atau vertikal agar model dapat mengenali objek dalam orientasi yang berbeda.
3. *Zooming*, Memperbesar atau memperkecil gambar untuk melihat objek dalam berbagai skala.
4. Penambahan Noise, Menambahkan gangguan visual pada gambar untuk mensimulasikan kondisi dunia nyata, seperti gambar yang kabur atau buram.

Augmentasi citra memungkinkan model untuk belajar lebih banyak tentang variasi objek, dan dengan demikian meningkatkan kemampuan deteksi objek pada kondisi yang tidak terduga. Augmentasi citra membantu dalam melatih model yang lebih robust dan mampu menangani berbagai kondisi citra yang mungkin ditemukan dalam dunia nyata[27].

Salah satu pendekatan yang banyak digunakan dalam penelitian terkini adalah augmentasi *on-the-fly* (online), yaitu teknik augmentasi yang dilakukan secara dinamis selama proses pelatihan model berlangsung. Pendekatan ini banyak digunakan pada sistem pembelajaran online dan *data stream* yang memiliki keterbatasan data berlabel dan kapasitas penyimpanan. augmentasi *on-the-fly* mampu meningkatkan keragaman data latih secara efektif karena setiap sampel dapat menghasilkan variasi yang berbeda pada setiap proses pelatihan[28]. Dengan mengintegrasikan augmentasi *on-the-fly* ke dalam mekanisme *active learning*, model memperoleh representasi data yang lebih kaya meskipun jumlah data asli terbatas, sehingga mampu meningkatkan kualitas pembelajaran, khususnya pada kondisi ketidakseimbangan kelas.

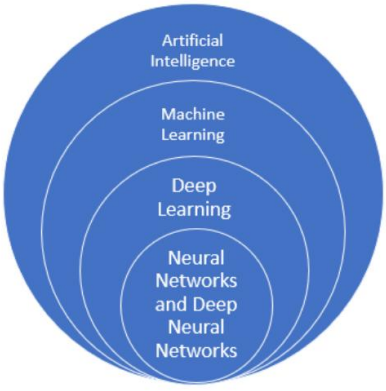
Penerapan augmentasi *on-the-fly* pada klasifikasi citra berbasis *Convolutional Neural Network* (CNN) efektif dalam mengurangi *overfitting* dan meningkatkan kemampuan generalisasi model[29]. Augmentasi dilakukan secara real-time

melalui transformasi seperti rotasi, *flipping*, dan perubahan skala, sehingga model tidak hanya menghafal pola tertentu, tetapi belajar fitur yang lebih umum. Meskipun augmentasi *on-the-fly* meningkatkan beban komputasi karena dilakukan bersamaan dengan pelatihan, penggunaan framework deep learning modern memungkinkan proses ini berjalan secara efisien.

2.3 *Artificial Intelligence*

Kecerdasan Buatan atau *Artificial Intelligence (AI)* adalah bidang ilmu komputer yang berfokus pada penciptaan sistem yang dapat meniru atau mensimulasikan kecerdasan manusia dalam menyelesaikan tugas-tugas tertentu[30]. Sistem AI dirancang agar mampu melakukan proses seperti belajar dari data (*learning*), mengenali pola (*pattern recognition*), memahami bahasa alami (*natural language processing*), dan membuat keputusan berdasarkan informasi yang tersedia (*decision making*).

AI bekerja dengan memanfaatkan algoritma yang dapat memproses dan menganalisis data dalam jumlah besar, sehingga memungkinkan sistem untuk belajar dan meningkatkan performa secara otomatis dari waktu ke waktu[31]. Pendekatan ini menjadikan AI sangat berguna dalam berbagai bidang seperti kesehatan, pertanian, manufaktur, dan juga pengolahan citra digital.

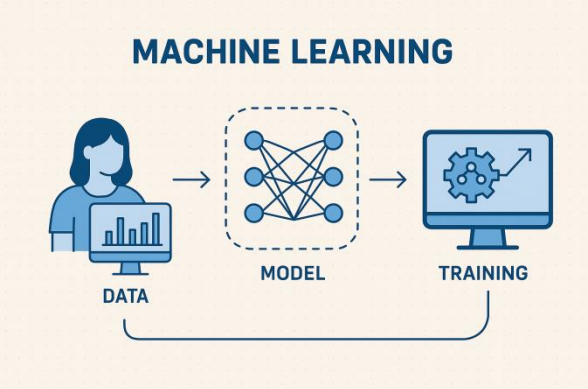


Gambar 1. Hirarki Kecerdasan Buatan

Gambar 1 merupakan hirarki konsep dalam bidang Kecerdasan Buatan (*Artificial Intelligence*/AI)[32]. Berikut adalah penjelasan dari masing-masing kategori :

1. *Artificial Intelligence* (AI), mencakup semua teknologi yang dirancang untuk meniru kecerdasan manusia, seperti mengenali suara, memahami bahasa, dan membuat keputusan.
2. *Machine Learning* (ML), ML adalah metode di mana sistem belajar dari data dan membuat prediksi atau keputusan tanpa diprogram secara eksplisit. Sistem belajar dari pola dalam data[33].
3. *Deep Learning*, sub-bidang dari ML yang menggunakan jaringan saraf buatan (*neural networks*) dengan banyak lapisan (*deep*) untuk meniru cara kerja otak manusia dalam memproses data yang kompleks, seperti gambar atau suara.
4. *Neural Networks* dan *Deep Neural Networks*, Merupakan dasar dari *Deep Learning*. Neural Networks adalah struktur komputasi yang meniru neuron pada otak manusia dan *Deep Neural Networks* adalah *neural network* dengan banyak lapisan tersembunyi (*hiden layer*), memungkinkan pemrosesan data yang jauh lebih kompleks dan presisi tinggi[34].

2.4 *Machine Learning*



Gambar 2. Konsep *Machine Learning*

Machine learning adalah cabang dari kecerdasan buatan (AI) yang berfokus pada pengembangan algoritma yang memungkinkan komputer untuk belajar dari data

tanpa diprogram secara eksplisit[33]. Proses ini dimulai dengan pengumpulan data**,** di mana data dikumpulkan dan disiapkan untuk pelatihan model. Data yang berkualitas sangat penting untuk hasil yang akurat. Selanjutnya adalah tahap pra-pemrosesan dan pemodelan, di mana data yang telah dikumpulkan diproses agar bersih dan siap untuk digunakan. Model *machine learning* seperti regresi, decision tree, SVM, atau *deep learning* kemudian dibangun untuk mengenali pola dari data. Pada tahap pelatihan model, algoritma mempelajari data dengan menyesuaikan parameter model untuk meminimalkan kesalahan. Model kemudian dievaluasi pada tahap evaluasi dan prediksi, menggunakan metrik seperti akurasi, presisi, dan recall untuk menilai performa model. Setelah dievaluasi, model siap digunakan untuk memprediksi data baru di dunia nyata.

Tiga pendekatan utama dalam machine learning adalah supervised learning (menggunakan data berlabel), unsupervised learning (mengelompokkan data tanpa label), dan reinforcement learning (belajar dari interaksi dengan lingkungan untuk memaksimalkan reward)[35]. Deep learning**,** sebagai bagian dari machine learning, menggunakan jaringan saraf tiruan (*neural networks*) berlapis-lapis yang sangat efektif dalam pengolahan data besar seperti gambar, suara, dan teks. Dengan langkah-langkah ini, *machine learning* mampu memberikan prediksi dan klasifikasi yang akurat dalam berbagai bidang.

2.5 *Deep Learning*

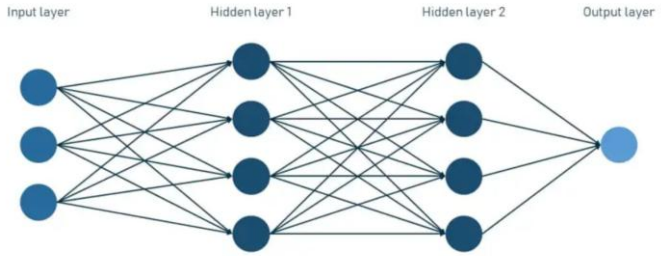
Deep Learning merupakan cabang dari jaringan syaraf tiruan yang memanfaatkan struktur multilapis (*deep neural networks*) untuk memproses data yang kompleks secara lebih efektif. Berbeda dengan jaringan syaraf tiruan sederhana, deep learning menggunakan banyak lapisan tersembunyi yang memungkinkan pembelajaran representasi data pada berbagai tingkat abstraksi, mulai dari pola dasar hingga konsep yang lebih kompleks[36]. Kekuatan utama dari *deep learning* adalah kemampuannya untuk melakukan ekstraksi fitur secara otomatis dari data mentah, tanpa memerlukan rekayasa fitur manual. Misalnya, pada pengolahan citra, lapisan awal dapat mengenali pola sederhana seperti tepi atau warna,

sedangkan lapisan lebih dalam mampu mendeteksi bentuk objek yang lebih spesifik. Proses ini dilakukan melalui mekanisme *forward propagation* untuk menghasilkan prediksi dan *backpropagation* untuk memperbarui bobot jaringan agar kesalahan prediksi semakin kecil[37].

Dalam praktiknya, berbagai arsitektur *deep learning* telah dikembangkan sesuai dengan jenis data yang diolah. *Convolutional Neural Networks* (CNN) banyak digunakan dalam pengolahan citra, sedangkan *Recurrent Neural Networks* (RNN) digunakan untuk data sekuensial seperti teks atau suara. Kemampuan ini menjadikan *deep learning* sangat efektif untuk tugas-tugas seperti klasifikasi, deteksi objek, segmentasi citra, pengenalan suara, hingga pemrosesan bahasa alami[38].

2.6 Jaringan Syaraf Tiruan

Jaringan Saraf Tiruan (JST) adalah model komputasi yang terinspirasi dari struktur dan fungsi jaringan saraf biologis[34]. JST terdiri dari unit pemrosesan sederhana yang disebut neuron buatan, saling terhubung melalui bobot (*weights*), dan bekerja bersama dalam lapisan-lapisan untuk menyelesaikan tugas seperti klasifikasi, regresi, dan pengenalan pola. JST belajar melalui penyesuaian bobot secara iteratif berdasar umpan balik dari kesalahan (*error*) yang dihasilkan selama pelatihan menggunakan metode seperti backpropagation dan algoritma optimisasi berbasis gradien. Metode-metode ini terus mengalami penyempurnaan dengan integrasi teknik modern seperti *dropout, batch normalization*, dan *optimizer adaptif* (misalnya Adam) untuk meningkatkan stabilitas dan performa jaringan. Model multi-layer dengan lapisan tersembunyi memungkinkan pembelajaran representasi data secara hirarkis (dalam), yang menjadi dasar *deep Learning[34]*. Keunggulan utama JST adalah kemampuannya dalam ekstraksi fitur otomatis, sehingga tidak lagi bergantung sepenuhnya pada rekayasa fitur manual[21].



Gambar 3. Lapisan Jaringan Syaraf Tiruan

Berdasarkan gambar 3, secara umum, JST terdiri dari tiga tipe lapisan utama: lapisan input, lapisan tersembunyi, dan lapisan output. Setiap neuron menerima input, melakukan transformasi (penjumlahan berbobot + bias), menerapkan fungsi aktivasi non-linear, dan meneruskan output ke neuron berikutnya. Hal ini memungkinkan jaringan untuk memodelkan hubungan non-linear kompleks dalam data. JST modern umumnya memiliki lebih dari dua lapisan tersembunyi, disebut *Deep Neural Network* (DNN). Lapisan-lapisan ini saling mengirim sinyal (*feedforward*), dan pelatihan menggunakan *backpropagation* untuk memperbarui bobot melalui minimisasi fungsi loss. Proses Pembelajaran dalam JST adalah sebagai berikut:

1. *Forward propagation*

Input melewati lapisan demi lapisan, dihitung dan diproses melalui fungsi aktivasi hingga menghasilkan output.

2. *Backpropagation*

Menghitung error dari output, menyebarkannya kembali untuk memperbarui bobot menggunakan algoritma optimisasi. Metode optimisasi modern seperti *Adam* dipakai untuk mempercepat dan memperhalus proses training, sebagaimana dibahas dalam berbagai tinjauan *deep Learning* terbaru[21].

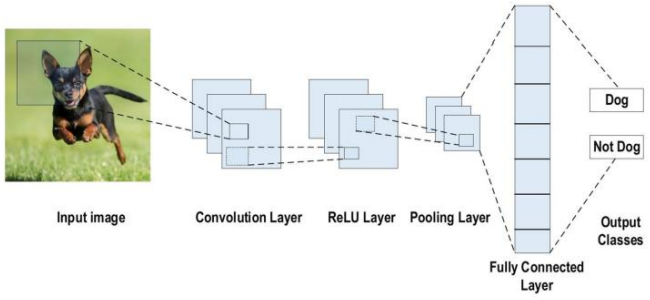
Dalam JST terdapat fungsi aktivasi. Fungsi aktivasi memberikan non-linearitas pada JST. ReLU (*Rectified Linear Unit*) menjadi sangat populer sejak diperkenalkan dan mulai digunakan secara luas dalam jaringan dalam (*deep networks*) karena efisiensi dan stabilitasnya dalam pelatihan.

2.7 CNN (*Convolutional Neural Network*)

Convolutional Neural Network (CNN) Dalam dunia *Deep Learning*, Convolutional Neural Network (CNN) adalah algoritma yang paling populer dan sering digunakan. Kelebihan utama CNN adalah kemampuannya mengenali fitur-fitur penting dari data secara otomatis, tanpa perlu bantuan manusia[39]. CNN banyak digunakan di berbagai bidang, seperti pengolahan gambar (computer vision), pengolahan suara, dan pengenalan wajah. Cara kerja CNN terinspirasi dari cara kerja otak manusia dan hewan, khususnya dari cara otak kucing memproses penglihatan[21]. Di otak, ada bagian yang disebut korteks visual yang memproses gambar secara bertahap inilah yang ditiru oleh CNN. CNN memiliki tiga kelebihan utama yaitu :

1. Representasi yang efisien
2. Koneksi yang lebih sedikit dan fokus hanya pada bagian penting
3. Penggunaan bobot yang sama berulang-ulang (parameter sharing)

Berbeda dari jaringan saraf biasa yang semua neuron saling terhubung (*fully connected*), CNN hanya melihat bagian kecil dari gambar pada satu waktu. Ini membuat CNN bisa bekerja lebih cepat dan butuh lebih sedikit data [21].



Gambar 4. Arsitektur CNN

Berdasarkan gambar 4, CNN biasanya terdiri dari beberapa lapisan utama, yaitu:

1. Lapisan konvolusi (mendeteksi fitur penting seperti garis, warna, dan bentuk).
2. Lapisan pooling (mengecilkan ukuran data agar proses lebih cepat dan mencegah *overfitting*).

3. Lapisan *fully connected* (mengambil semua informasi dari lapisan sebelumnya dan mengubahnya menjadi hasil akhir).

Data yang masuk ke CNN biasanya berupa gambar berukuran tinggi × lebar × kedalaman (misalnya 224×224 piksel dengan 3 saluran warna untuk gambar RGB)[21]. Di setiap lapisan konvolusi, digunakan filter atau kernel kecil untuk menyapu bagian-bagian gambar dan menangkap pola tertentu. Filter ini memiliki ukuran yang lebih kecil dari gambar dan digunakan berulang-ulang ke seluruh bagian gambar. Setelah hasil konvolusi dihitung, data akan diproses oleh fungsi aktivasi untuk menambahkan sifat non-linear. Kemudian dilakukan proses pooling, misalnya max pooling, untuk mengambil nilai tertinggi di setiap area kecil gambar dan mengecilkan ukurannya.

Di akhir, lapisan *fully connected* akan menggabungkan semua fitur yang sudah dideteksi dan menghasilkan skor klasifikasi. Misalnya, dalam klasifikasi gambar, skor ini menunjukkan seberapa besar kemungkinan gambar tersebut adalah kucing, anjing, mobil, atau lainnya. Dengan kata lain, CNN bekerja seperti mata dan otak kita dalam mengenali sesuatu melihat bagian demi bagian, mengenali pola, lalu menyimpulkan hasil akhirnya. Berikut Persamaan lapisan konvolusi dalam CNN :

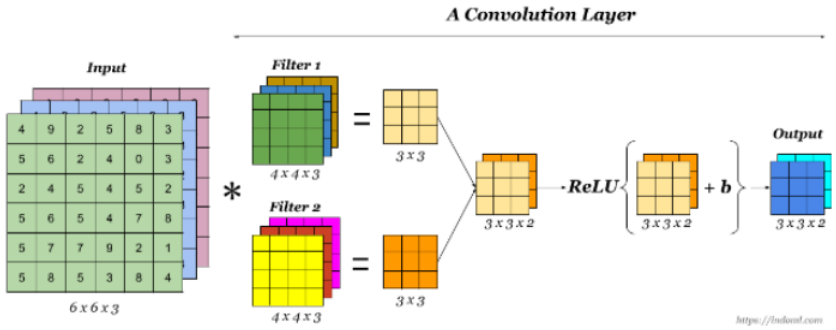
$$Hk = f(W^k x + b^k)$$

Hk = hasil dari filter ke-k (disebut feature map atau peta fitur),
Wk = bobot (weights) dari filter ke-k (kernel),
x = input (misalnya gambar atau feature map dari lapisan sebelumnya),
bk = bias untuk filter ke-k,
f = fungsi aktivasi (misalnya ReLU) yang membuat hasilnya non-linear

Persamaan tersebut menunjukkan cara CNN menghasilkan fitur dari gambar input. Ini adalah dasar dari cara CNN belajar mengenali pola dalam gambar[21].

2.7.1 *Convolutional Layer*

Lapisan konvolusi merupakan komponen paling penting dalam arsitektur CNN. Lapisan ini terdiri dari kumpulan filter konvolusi (disebut juga kernel). Citra masukan (*input image*) yang direpresentasikan dalam bentuk matriks berdimensi-N akan dikonvolusikan dengan filter-filter ini untuk menghasilkan peta fitur (*feature map*)[21]. Berikut merupakan gambar dari konsep layer konvolusi.

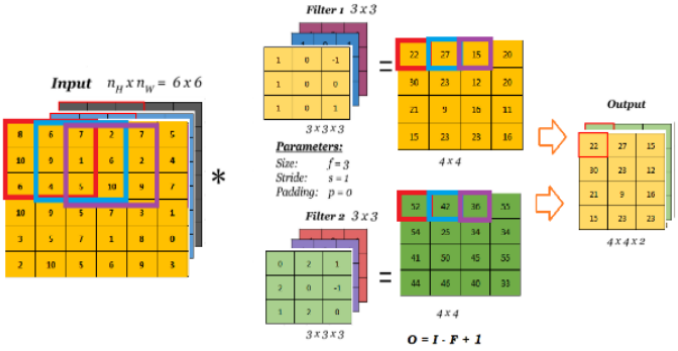


Gambar 5. Layer Konvolusi

1. Kernel

Kernel adalah grid atau kisi dari sejumlah nilai diskrit. Setiap nilai disebut sebagai bobot kernel. Bobot-bobot ini awalnya diisi dengan angka acak saat proses pelatihan CNN dimulai. Seiring pelatihan berlangsung, bobot-bobot ini akan diperbarui agar kernel dapat "belajar" mengekstrak fitur-fitur penting dari citra.

2. Operasi Konvolusi



Gambar 6. Operasi Konvolusi

Dalam *Convolutional Neural Network* (CNN), input yang digunakan berupa gambar dengan kanal tunggal maupun jamak. Gambar grayscale hanya memiliki satu kanal, sedangkan gambar berwarna seperti RGB memiliki tiga kanal, yaitu merah (*Red*), hijau (*Green*), dan biru (*Blue*). Kanal ini berfungsi sebagai representasi dari komponen warna yang membentuk gambar tersebut[21].

Operasi dasar pada CNN adalah konvolusi, yaitu proses di mana sebuah kernel (atau filter) dengan ukuran tertentu digeser secara horizontal dan vertikal pada seluruh bagian gambar. Misalnya, jika digunakan gambar grayscale berukuran 4 × 4 piksel dan kernel berukuran 2 × 2 yang diinisialisasi dengan nilai acak, maka kernel tersebut akan melakukan perhitungan pada setiap posisi yang dilaluinya. Pada setiap posisi, dilakukan operasi perkalian elemen per elemen (*dot product*) antara nilai piksel pada area gambar dan nilai bobot pada kernel, kemudian hasil perkalian tersebut dijumlahkan untuk menghasilkan satu nilai. Nilai ini akan menjadi bagian dari feature map output, yaitu representasi baru yang berisi fitur-fitur penting dari gambar awal[21]. Keuntungan Utama Lapisan Konvolusi adalah sebagai berikut.

1. Konektivitas Jarang (*Sparse Connectivity*)

Tidak semua neuron terhubung ke semua neuron pada lapisan berikutnya seperti pada jaringan saraf biasa. Ini mengurangi jumlah bobot yang dibutuhkan serta menghemat memori.

2. Berbagi Bobot (*Weight Sharing*)

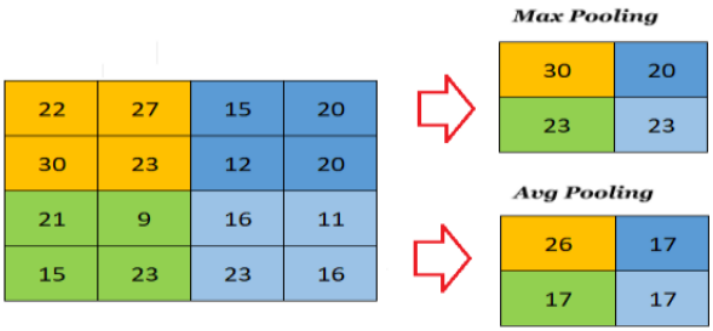
Semua piksel pada input menggunakan bobot yang sama, sehingga proses pelatihan lebih efisien dan cepat.

2.7.2 *Pooling Layer*

Lapisan pooling merupakan salah satu komponen penting dalam arsitektur *Convolutional Neural Network* (CNN) yang berfungsi untuk mengurangi dimensi spasial (ukuran) dari feature map hasil konvolusi[32]. Pengurangan dimensi ini

dilakukan dengan tetap mempertahankan informasi penting dari fitur-fitur utama yang telah diekstraksi sebelumnya. Pooling membantu menyederhanakan representasi data, mengurangi jumlah parameter yang harus dipelajari, serta meningkatkan efisiensi komputasi dan ketahanan model terhadap variasi kecil dalam input. Tujuan Pemrosesan pada pooling layer adalah sebagai berikut :

1. Menurunkan ukuran data (*downsampling*), sehingga mengurangi beban komputasi.
2. Memperkuat fitur utama dan mengurangi noise.
3. Meningkatkan ketahanan model terhadap translasi kecil, rotasi, atau distorsi gambar.
4. Mengurangi risiko *overfitting* dengan menyederhanakan representasi fitur.



Gambar 7. Proses pooling

Pooling dilakukan dengan menerapkan jendela (*kernel*) berukuran tertentu pada feature map dan menghitung nilai tertentu (misalnya nilai maksimum atau rata-rata) dari elemen-elemen dalam jendela tersebut[32] (gambar 7). Proses ini dilakukan secara bertahap sesuai dengan ukuran langkah (*stride*) yang telah ditentukan, hingga seluruh feature map terproses. Misalkan *Feature map* berukuran 4×4, kernel pooling berukuran 2×2, stride adalah 2. Maka hasil pooling akan menghasilkan output berukuran 2×2. Beberapa jenis metode pooling yang umum digunakan adalah:

1. *Max Pooling*: Mengambil nilai maksimum
2. *Min Pooling*: Mengambil nilai minimum.
3. *Average Pooling*: Mengambil nilai rata-rata.

4. *Global Average Pooling* (GAP) dan *Global Max Pooling* juga digunakan dalam beberapa arsitektur modern.

2.7.3 *Activation Function*

Fungsi aktivasi merupakan komponen penting dalam setiap jenis jaringan saraf, termasuk *Convolutional Neural Network* (CNN). Tugas utamanya adalah melakukan pemetaan dari input ke output secara non-linear, sehingga jaringan saraf mampu mempelajari pola kompleks dalam data[21]. Tanpa fungsi aktivasi, jaringan saraf hanya akan menjadi model linear sederhana, yang tidak mampu menyelesaikan masalah klasifikasi atau prediksi yang bersifat non-linear.
Sebelum fungsi aktivasi dijalankan, nilai input ke sebuah neuron dihitung terlebih dahulu melalui proses berikut :

$$z = \sum w1.\,\mathrm{x1} + \mathrm{b}$$

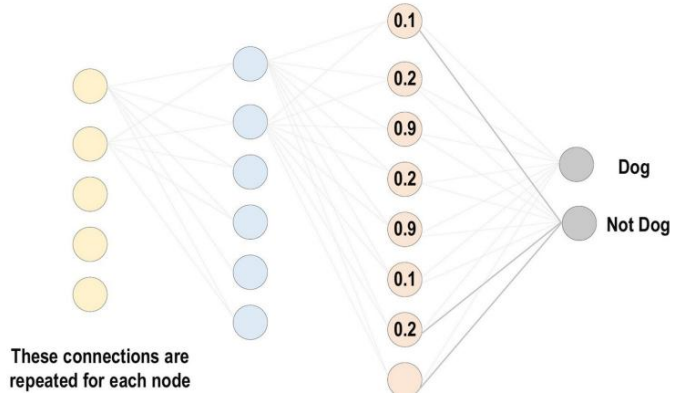
di mana,

x1 = input ke neuron
$w1$ = bobot (weight) yang terhubung ke input tersebut
b = bias
z = hasil penjumlahan tertimbang yang akan menjadi input fungsi aktivasi

Kemudian, nilai z ini akan dimasukkan ke dalam fungsi aktivasi, yang akan menentukan apakah neuron tersebut "menembak" (*fire*) atau tidak, dengan menghasilkan output tertentu (biasanya antara rentang nilai tertentu, seperti 0 hingga 1 atau -1 hingga 1). Fungsi aktivasi membantu neuron memutuskan apakah harus meneruskan informasi atau tidak. Ia bekerja pada hasil perhitungan input × bobot. Dengan demikian, fungsi aktivasi tidak mengatur bobot, tetapi mengatur seberapa besar sinyal yang dikeluarkan neuron[40].

2.7.4 *Fully Connected Layer*

Lapisan *Fully Connected* (FC) adalah bagian dari arsitektur CNN yang umumnya terletak di akhir jaringan. Pada lapisan ini, setiap neuron terhubung dengan semua neuron di lapisan sebelumnya, mirip dengan struktur pada jaringan saraf tiruan tradisional (*Multilayer Perceptron*). Lapisan ini berfungsi sebagai klasifikator utama dalam CNN, yaitu menentukan kelas akhir dari suatu input berdasarkan fitur-fitur yang telah diekstraksi dan diproses oleh lapisan-lapisan sebelumnya (konvolusi dan pooling)[21]. Gambaran cara kerja lapisan ini adalah sebagai berikut :



Gambar 8. *Fully Connected Layer*

1. Input ke FC Layer

Data yang masuk ke FC layer berasal dari lapisan konvolusi atau pooling terakhir. Namun, karena FC hanya dapat menerima data dalam bentuk vektor 1 dimensi, maka data berupa feature map (berdimensi 2D atau 3D) harus terlebih dahulu diubah melalui proses flattening.

2. Proses Klasifikasi

Setelah di-*flatten*, data akan diproses melalui satu atau lebih lapisan FC, di mana setiap neuron menghitung kombinasi linear dari input dan bobot, ditambah bias, dan hasilnya diteruskan ke fungsi aktivasi.

3. Output Akhir

Neuron pada lapisan output (biasanya FC terakhir) akan menghasilkan nilai representasi dari kelas (misalnya probabilitas kelas jika menggunakan (*Softmax*), yang akan digunakan untuk proses prediksi atau klasifikasi.

2.7.5 *Loss Function*

Loss function adalah fungsi yang digunakan untuk mengukur seberapa baik atau buruk performa prediksi model CNN terhadap data pelatihan[21]. Fungsi ini menghitung selisih (*error*) antara hasil prediksi model dan label sebenarnya. Nilai loss yang dihasilkan akan digunakan sebagai dasar untuk proses optimisasi, yaitu memperbarui bobot-bobot jaringan menggunakan algoritma seperti backpropagation dan gradient descent, sehingga model dapat belajar dengan lebih baik di setiap epoch. Loss function bekerja berdasarkan dua parameter utama yakni :

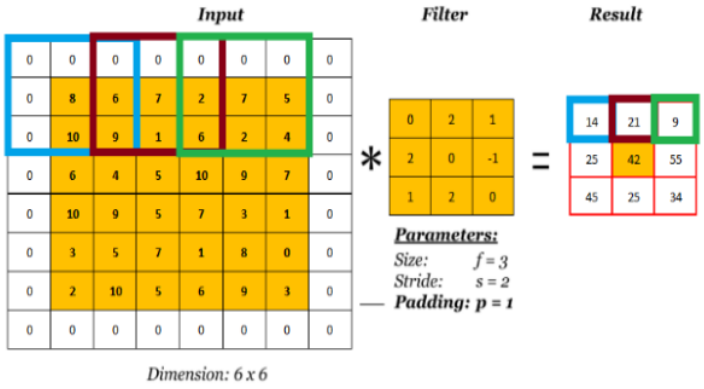
1. Prediksi model (*predicted output*) – nilai yang dihasilkan oleh CNN pada output layer.
2. Label aktual (*true label*) – nilai target yang benar dari dataset pelatihan.

2.7.6 *Stride*

Stride merupakan bagian penting dalam proses konvolusi pada *Convolutional Neural Network* (CNN). Secara umum, konvolusi dilakukan dengan cara menggeser filter (atau matriks kernel) di atas matriks input gambar. Pada setiap posisi pergeseran, dilakukan perhitungan konvolusi antara nilai-nilai pada filter dan nilai pada area input yang sesuai. Hasil perhitungan ini kemudian menjadi elemen dalam *feature map output.* Nilai *stride* menentukan seberapa jauh filter berpindah setiap kali konvolusi dilakukan[41]. Misalnya, jika *stride* ditetapkan sebesar 1, maka filter akan bergeser satu langkah ke kanan atau ke bawah setelah setiap proses konvolusi. Proses ini dimulai dengan melakukan konvolusi pertama antara filter dan area awal dari input. Setelah selesai, filter akan bergeser sejauh nilai stride ke posisi berikutnya, dan proses konvolusi kedua dilakukan dengan

area input yang baru. Proses ini terus diulang hingga seluruh area input telah dikonvolusi dengan filter.

Dengan menggunakan nilai stride tertentu, ukuran dari feature map hasil konvolusi menjadi lebih kecil dibandingkan dengan ukuran input awal. Hal ini dikarenakan filter tidak melintasi seluruh posisi piksel secara satu per satu (kecuali jika *stride* = 1), melainkan dengan langkah yang lebih besar, sehingga jumlah total posisi yang dihitung menjadi lebih sedikit. Oleh karena itu, pengaturan nilai stride dalam setiap lapisan konvolusi sangat memengaruhi resolusi spasial dari output dan jumlah informasi yang dipertahankan dari input asli.



Gambar 9. *Stride* dan *padding*

2.7.7 *Padding*

Padding merupakan proses penambahan nilai tertentu, biasanya nol (0) di setiap sisi gambar input sebelum dilakukan proses konvolusi[32]. Meskipun ukuran matriks input menjadi lebih besar setelah *padding*, hasil dari konvolusinya tetap akan lebih kecil dari ukuran gambar aslinya. Tujuan dari padding adalah memberikan batasan yang lebih jelas dari suatu image, sehingga proses dapat menjadi lebiih fokus. Tanpa padding, fitur-fitur di bagian pinggir gambar bisa cepat terabaikan karena area konvolusi tidak cukup menangkap bagian tepi tersebut[42].

Selain itu, padding juga membantu agar ukuran output (*feature map*) tidak menyusut terlalu cepat, sehingga mempermudah pengolahan pada lapisan-lapisan selanjutnya dalam jaringan. Dalam beberapa kasus, padding juga digunakan untuk menyesuaikan ukuran output agar sama dengan ukuran input, terutama jika output tersebut akan digunakan lagi di proses konvolusi berikutnya. Dengan demikian, padding berperan penting dalam mempertahankan struktur informasi gambar dan menjaga stabilitas dimensi dalam arsitektur CNN.

2.8 *Inception*

Arsitektur *Inception* pertama kali diperkenalkan oleh tim Google pada tahun 2015 dalam model yang disebut *GoogLeNet*. Inception dirancang untuk memproses gambar (citra) dengan ukuran dan bentuk objek yang bervariasi. Misalnya, dalam satu gambar bisa saja ada objek kecil dan besar yang perlu dikenali dengan baik. Untuk mengatasi masalah tersebut, *Inception* menggunakan *multi-scale convolution*. Dimana pada satu lapisan layer jaringan, digunakan beberapa filter konvolusi dengan ukuran berbeda-beda secara paralel[43], seperti :

1. Filter ukuran 1×1: untuk menangkap detail yang sangat kecil dan mengurangi dimensi data.
2. Filter ukuran 3×3: untuk menangkap pola sedang.
3. Filter ukuran 5×5: untuk menangkap pola yang lebih besar.

Setelah masing-masing filter ini memproses citra, hasilnya digabungkan (*concatenate*) menjadi satu. Proses ini membantu model untuk mengenali fitur dari berbagai skala dalam satu langkah. Untuk membuat proses komputasi tetap ringan, *Inception* menggunakan *convolution* 1×1 sebagai pengurang dimensi sebelum menggunakan filter besar (misalnya 5×5), gambar terlebih dahulu dilewatkan melalui filter 1×1 untuk mengurangi ukuran datanya. Ini membuat model lebih cepat dan memori yang digunakan lebih sedikit [44].

Keunggulan utama Inception adalah kemampuannya untuk menangani kompleksitas gambar, terutama ketika ada banyak objek dengan ukuran berbeda

dalam satu gambar[45]. Misalnya, dalam penelitian ini, saat mendeteksi dan mengklasifikasikan spesies lebah tanpa sengat, ukuran tubuh lebah, warna, dan bentuknya bisa bervariasi. *Inception* mampu mengatasi ini karena bisa mengenali pola besar dan kecil dalam satu waktu. Selanjutnya, pengembangan dari *GoogLeNet* menghasilkan model yang lebih kuat, seperti:

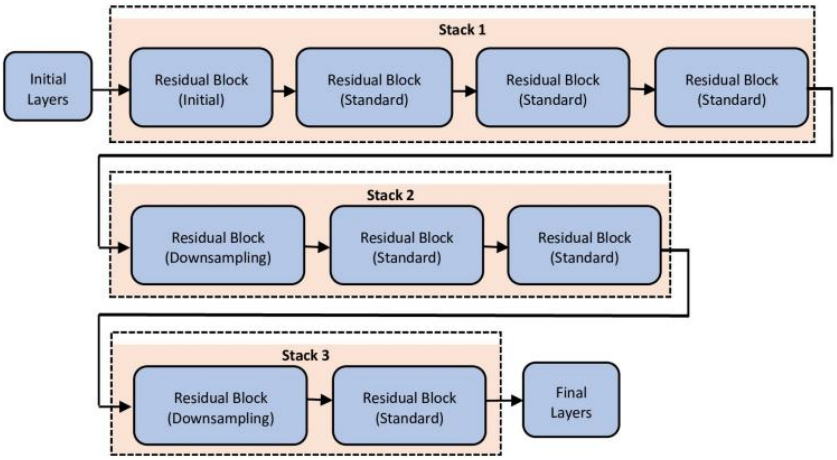
1. *Inception-v3*: ditambahkan faktor batch normalization dan faktor regularisasi agar pelatihan lebih stabil.
2. *Inception-ResNet*: menggabungkan arsitektur *Inception* dengan konsep residual (loncatan koneksi) agar pelatihan lebih cepat dan akurat.

2.9. *Residual Network (ResNet)*

Residual Network (*ResNet*) adalah arsitektur *Artificial Neural Network* (ANN) yang dikembangkan tahun 2015. ResNet dirancang untuk mengatasi masalah *vanishing gradient* yang sering muncul saat melatih jaringan saraf yang sangat dalam. Dalam jaringan yang dalam, gradien dari fungsi *loss* bisa menjadi sangat kecil sehingga bobot pada lapisan awal sulit diperbarui, membuat proses pelatihan menjadi tidak efektif[46]. *ResNet* memecahkan masalah ini dengan memperkenalkan *residual connection* atau *shortcut connection.* Konsep utamanya adalah membiarkan input awal x diteruskan langsung ke output dari blok konvolusional, sehingga model hanya perlu mempelajari fungsi residual F(x), yang mewakili selisih antara output yang diinginkan dan input awal[45]. Persamaan sederhananya adalah sebagai berikut :

$$H(x) = F(x) + x$$

di mana $H(x)$ adalah output blok, $F(x)$ Fadalah fungsi residual yang dipelajari oleh lapisan konvolusional, dan (x) adalah input awal. *Shortcut connection* ini memungkinkan arus informasi dan gradien mengalir lebih mudah ke lapisan-lapisan sebelumnya, sehingga memungkinkan pelatihan jaringan yang sangat dalam menjadi lebih stabil dan efisien.



Gambar 10. Arsitektur ResNet

Dari gambar 10, arsitektur *ResNet* secara umum terdiri dari tiga bagian utama:

1. Lapisan awal (*initial layers*): berfungsi untuk mengekstraksi fitur dasar dari input.
2. Stack of residual blocks: bagian inti jaringan yang berisi blok-blok residual.
3. Lapisan akhir (*final layers*): berfungsi sebagai pengklasifikasi berdasarkan fitur yang telah dipelajari.

ResNet sendiri dikenal memiliki struktur yang sederhana dan intuitif dibandingkan GoogLeNet, karena menggunakan *residual connections* yang memungkinkan aliran gradien tetap stabil pada jaringan yang sangat dalam. Meskipun demikian, ResNet konvensional memiliki keterbatasan dalam mempertahankan fitur penting pada lapisan awal akibat proses *downsampling*[40]. Residual blocks dalam *ResNet* terdiri dari tiga jenis :

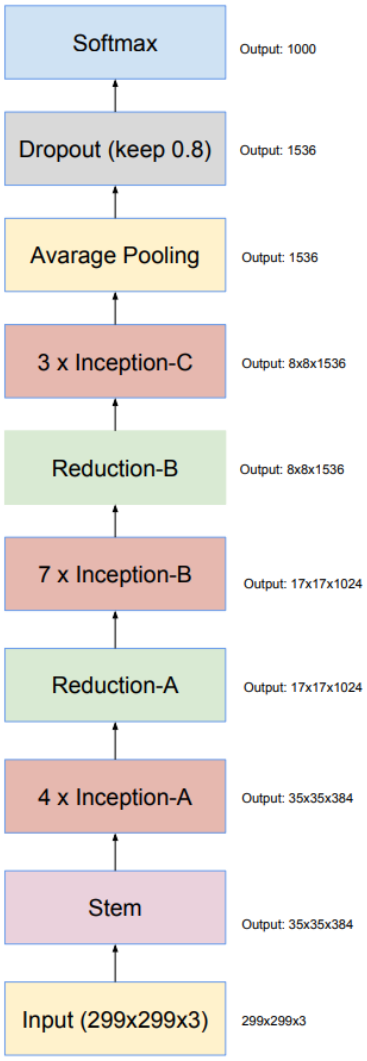
1. *Initial Residual Block*, terletak di awal stack pertama dan menggunakan struktur *bottleneck* (konvolusi 1×1 → 3×3 → 1×1). Pada blok ini, proses konvolusi dilakukan dengan stride [1,1] agar ukuran spasial tetap sama.
2. *Standard Residual Block*, muncul beberapa kali dalam satu stack dan berfungsi menjaga ukuran aktivasi (*activation size*) tetap sama. Blok ini umumnya menggunakan konvolusi 3×3.

3. *Down-Sampling Residual Block*, muncul satu kali di awal setiap stack (kecuali stack pertama) dan bertugas mengecilkan dimensi spasial (*downsampling*) sebesar faktor 2, dilakukan melalui stride pada konvolusi pertama di blok tersebut.

Struktur residual seperti ini memungkinkan pelatihan jaringan dengan kedalaman besar bahkan hingga ratusan lapisan tanpa penurunan performa yang signifikan [45].

2.10 *Inception-ResNet*

Inception-ResNet merupakan arsitektur jaringan saraf konvolusional yang menggabungkan dua konsep utama, yaitu arsitektur *Inception* dan *Residual Connection* [7]. Inception awalnya diperkenalkan untuk mengatasi masalah kompleksitas model dan meningkatkan efisiensi ekstraksi fitur melalui penggunaan beberapa ukuran kernel konvolusi secara paralel dalam satu layer. Sedangkan *Residual Connection* digunakan untuk mengatasi degradasi performa pada jaringan yang sangat dalam dengan cara menambahkan *shortcut connection* yang melewati beberapa lapisan, sehingga memudahkan propagasi gradien selama proses pelatihan. Penggabungan kedua konsep ini dalam *Inception-ResNet* bertujuan untuk memaksimalkan kekuatan fitur ekstraksi *Inception* dengan stabilitas dan kemudahan pelatihan yang dibawa oleh *residual connection*. Dengan demikian, *Inception-ResNet* dapat membangun jaringan yang sangat dalam tanpa mengalami penurunan performa akibat *vanishing gradient*. Berikut adalah arsitektur dasar dari Inception-Resnet V2.



Gambar 11. Arsitektur Inception-ResNet V2

Arsitektur InceptionResNetV2 terdiri atas 164 lapisan, yang mencakup 4 lapisan *max pooling* serta 160 lapisan konvolusi, dengan total parameter sekitar 55 juta[47]. Berdasarkan gambar 11 usunan arsitekturnya meliputi beberapa blok utama, yaitu *input block* (berukuran 229 × 229 piksel), *stem*, 5 blok Inception-Resnet-A, Reduction-A, 10 blok Inception-Resnet-B, Reduction-B, 5 blok Inception-Resnet-C, kemudian diikuti *average pooling*, *dropout*, dan *softmax* sebagai lapisan klasifikasi[5]. Rincian lengkap arsitektur ditunjukkan pada gambar 11. Pada bagian *stem*, jaringan terdiri dari lapisan konvolusi, proses *filter concatenation*, serta *max pooling*. Lapisan konvolusi bekerja menggunakan kernel

atau *convolutional filters* untuk mengekstraksi fitur di mana bobot filter akan disesuaikan selama pelatihan agar mampu mempelajari pola penting pada citra Sementara itu, *max pooling* digunakan untuk memperkecil ukuran *feature map* namun tetap mempertahankan fitur dominan[48].

Setelah bagian *stem*, jaringan dilanjutkan ke blok Inception-Resnet-A. Setiap rangkaian blok Inception-Resnet umumnya diikuti blok Reduction, kecuali pada bagian Inception-Resnet-C. Perbedaan utama di antara Inception-Resnet-A, B, dan C terletak pada jumlah filter serta ukuran *feature map* yang dihasilkan. Pada Inception-ResNet-A, proses konvolusi terakhir menggunakan filter 1 × 1 Conv (384 *linear*), sehingga menghasilkan 384 *feature map*[5]. Jumlah filter dan keluaran *feature map* ini meningkat seiring bertambahnya blok, di mana Inception-Resnet-B dan Inception-Resnet-C masing-masing menghasilkan 1154 dan 2048 filter linear[5]. Peningkatan bertahap ini menunjukkan bahwa jaringan mampu menangkap pola visual yang semakin kompleks sepanjang proses pelatihan.

Setiap blok Inception-Resnet memanfaatkan koneksi residual, yaitu dengan menjumlahkan input awal dan keluaran dari jalur konvolusi paralel, sehingga aliran informasi residual dapat berlangsung langsung dan membantu mengatasi masalah *vanishing gradient*. Hasil penjumlahan tersebut kemudian diproses oleh fungsi aktivasi ReLU, yang mempertahankan nilai positif dan mengubah nilai negatif menjadi nol. ReLU memperkenalkan sifat non-linear, memungkinkan jaringan mempelajari representasi fitur yang lebih kompleks[49]. Setelah lima kali pengulangan Inception-Resnet-A, keluaran disalurkan ke blok Reduction-A. Lalu, keluaran Reduction-A diproses pada sepuluh pengulangan Inception-Resnet-B, dan selanjutnya menuju blok Reduction-B. Kedua blok Reduction berfungsi memperkecil dimensi *feature map* sambil tetap menjaga informasi visual penting melalui penggunaan lapisan *MaxPool*. Hal ini terlihat dari berkurangnya dimensi keluaran, misalnya dari 35 × 35 × 256 pada Inception-Resnet-A menjadi 17 × 17 × 896 pada Reduction-A, serta dari 17 × 17 × 896 pada Inception-Resnet-B menjadi 8 × 8 × 1792 pada Reduction-B. Perubahan ini menunjukkan bahwa

InceptionResNet mampu mengekstraksi fitur secara efisien dengan tingkat kompleksitas yang lebih rendah [5].

Setelah melewati blok Inception-Resnet-C, *feature map* kemudian diproses melalui *average pooling*. Selanjutnya, *dropout* diterapkan sebagai metode regularisasi dengan menetapkan beberapa unit tersembunyi secara acak menjadi nol, sehingga mencegah terjadinya *overfitting*. Pada arsitektur ini, *dropout* menggunakan pengaturan "keep 0.8", yang berarti 80% unit tetap dipertahankan selama proses pelatihan. Di bagian akhir, fungsi *softmax* digunakan pada lapisan keluaran untuk menghasilkan probabilitas kelas dalam rentang 0 hingga 1 [48].

Arsitektur *Inception-ResNet-v1* dan *Inception-ResNet-v2* dikembangkan dengan menggabungkan modul multi-skala *Inception* dan *residual connection* dari ResNet. Versi 1 dirancang agar memiliki biaya komputasi setara *Inception-v3* (efisien), sedangkan versi 2 memiliki kompleksitas dan jumlah parameter lebih besar mirip Inception-v4 namun menghasilkan akurasi lebih tinggi [5]. Berikut adalah versi *Inception-ResNet.*

Tabel 1. Perbedaan versi *Inception-ResNet* 1 dan 2

Model	**Deskripsi**
Inception-ResNet-v1	Inception-ResNet-v1 merupakan versi awal dari arsitektur gabungan antara *Inception dan ResNet*. Model ini dirilis pada tahun 2016. Inception-ResNet-v1 memiliki arsitektur yang lebih ringan dengan *stem* yang menghasilkan keluaran sebesar 256 channel. Jumlah filter pada modul-modul seperti Inception-ResNet-A, B, C, serta Reduction-A lebih kecil dibandingkan v2, sehingga total parameter jaringan lebih sedikit. Struktur yang lebih sederhana ini membuat v1 lebih cepat dilatih dan membutuhkan sumber daya komputasi yang lebih rendah[50].
Inception-ResNet-v2	Inception-ResNet-v2 adalah versi lanjutan dari model sebelumnya yang juga dirilis pada tahun 2016. Inception-

	ResNet-v2 memiliki kompleksitas yang lebih tinggi, dimulai dari *stem* yang menghasilkan keluaran 384 channel, lebih besar dari v1. Selain itu, setiap modul Inception dan blok reduksi memiliki jumlah filter dan parameter yang lebih banyak. Peningkatan kapasitas ini memungkinkan v2 menghasilkan representasi fitur yang lebih kuat, namun dengan konsekuensi memerlukan komputasi yang lebih besar dan proses pelatihan yang lebih berat[50]

Salah satu keunggulan utama *Inception-ResNet* adalah kemampuannya dalam mendeteksi detail objek kecil dengan akurasi tinggi. Hal ini dikarenakan arsitektur *Inception* yang menggunakan berbagai ukuran filter secara paralel mampu menangkap fitur dengan skala berbeda, mulai dari fitur kasar hingga fitur halus[51]. *Residual connection* membantu mempertahankan informasi penting selama proses pelatihan sehingga fitur-fitur detail tidak hilang. Berbagai studi menunjukkan bahwa model ini unggul dalam tugas-tugas pengenalan gambar yang menuntut sensitivitas tinggi terhadap detail, seperti deteksi objek kecil dan segmentasi citra.

Penelitian terkait juga membuktikan bahwa *Inception-ResNet* memberikan performa yang lebih baik dibandingkan dengan arsitektur konvolusional tradisional maupun Inception biasa, terutama dalam konteks dataset dengan kompleksitas tinggi dan variasi objek yang besar. Hanya saja Inception-ResNet memeprlukan sumberdaya komputasi yang cukup besar karena tergolong model yang sangat besar dengan jumplah parameter yang banyak[52]. Dengan demikian, penggunaan *Inception-ResNet* sangat direkomendasikan dalam aplikasi *computer vision* yang membutuhkan akurasi tinggi pada objek-objek berukuran kecil. Kekurangan dan kelebihan *Inception RestNet* dapat dilihat pada tabel Berikut.

Tabel 2. Kekurangan dan Kelebihan *Inception ResNet*

Aspek	Kelebihan	Kekurangan
Arsitektur	Menggabungkan keunggulan Inception (*multi-scale feature extraction*) dan ResNet (*residual learning)*	Arsitektur kompleks, sulit dimodifikasi dan dipahami bagi pemula
Kecepatan Konvergensi	Lebih cepat konvergen dibanding Inception biasa karena adanya *residual connection*	Membutuhkan penyesuaian skala aktivasi agar training tetap stabil
Akurasi & Performa	Dapat mencapaki akurasi tinggi untuk data yang sangat detail	Sangat bergantung pada dataset karena merupakan model yang besar
Komputasi	Efisien dalam penggunaan feature map karena gabungan pendekatan *convolution* besar & kecil	Butuh sumber daya besar: parameter dan FLOPs tinggi
Training	Mengatasi masalah *vanishing gradient* pada jaringan sangat dalam	Lebih sensitif terhadap hyperparameter tuning dibanding *ResNet* biasa
Fleksibilitas	Cocok untuk berbagai jenis klasifikasi gambar skala besar	Kurang modular, tidak sefleksibel *ResNet* untuk transfer learning ringan
Ekstraksi Fitur	Fitur yang dihasilkan lebih robust dan berkualitas tinggi	Fitur tidak jauh berbeda secara signifikan dibanding *ResNet.*

2.11 *Transfer Learning*

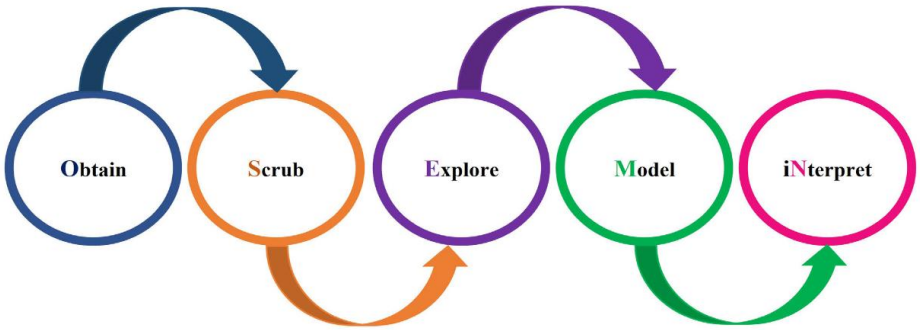
Transfer learning adalah pendekatan dalam pembelajaran mesin di mana pengetahuan yang diperoleh dari pelatihan model pada tugas tertentu digunakan kembali untuk tugas lain yang berbeda namun terkait[53]. Dalam konteks

computer vision, *transfer learning* biasanya memanfaatkan model deep learning yang telah dilatih sebelumnya (*pre-trained model*) pada dataset besar seperti *ImageNet*, kemudian diadaptasi untuk tugas baru dengan dataset yang lebih kecil. Teknik ini memberikan solusi pada masalah keterbatasan data, dengan memanfaatkan model yang telah dilatih menggunakan dataset yang besar, lalu digunakan kembali pada tugas dengan dataset yang baru. Ketika dataset kecil digunakan untuk pelatihan model dari awal, risiko *overfitting* sangat tinggi. Dengan memanfaatkan pre-trained models, model dapat menjaga generalisasi yang baik meskipun jumlah data terbatas.

Salah satu model populer yang digunakan dalam *transfer learning* adalah *Inception-ResNet*, yaitu kombinasi dari arsitektur *Inception* dan *Residual Network* (*ResNet*). *Inception-ResNet* menggabungkan keunggulan dari modul Inception yang memproses informasi pada berbagai skala dengan *shortcut connections* dari *ResNet* yang membantu dalam mengurangi masalah vanishing gradient dan mempercepat konvergensi pelatihan[8]. Model ini telah dilatih pada dataset besar seperti ImageNet, yang berisi lebih dari 1 juta gambar dari 1000 kelas berbeda, sehingga memiliki representasi fitur visual yang sangat kuat dan dapat diadaptasi untuk berbagai tugas klasifikasi gambar lainnya. Implementasi *InceptionResNet-V2* telah digunakan dalam berbagai penelitian di Indonesia, seperti pada klasifikasi malware, yang menunjukkan akurasi tinggi dalam skenario dengan dataset terbatas.

2.12 OSEMN

OSEMN merupakan salah satu metodologi yang sering digunakan dalam bidang *data science* untuk mengelola, menganalisis, dan menginterpretasikan data secara sistematis. OSEMN merupakan akronim dari lima tahapan utama, yaitu *Obtain, Scrub, Explore, Model,* dan *Interpret.*



Gambar 12. Alur OSEMN

1. Obtain

Tahap awal berupa pengumpulan data yang relevan dan sesuai dengan tujuan penelitian. Data dapat berasal dari berbagai sumber baik internal maupun eksternal, dengan format terstruktur maupun tidak terstruktur. Kualitas dan kelengkapan data pada tahap ini sangat mempengaruhi keberhasilan analisis selanjutnya. Tahap *Obtain* dalam konteks ini berfungsi untuk memastikan bahwa data yang dikumpulkan relevan, representatif, dan sesuai dengan tujuan klasifikasi[54].

2. Scrub

Data yang telah diperoleh kemudian melalui proses pembersihan (*data cleaning*) untuk mengatasi masalah duplikasi, *missing values*, atau inkonsistensi. Tujuannya adalah memastikan bahwa dataset berada dalam kondisi berkualitas tinggi dan bebas dari gangguan yang dapat menurunkan performa model klasifikasi. Tahap ini sangat penting karena kualitas pengolahan awal sangat memengaruhi akurasi, reliabilitas, dan stabilitas hasil model di tahap akhir[54].

3. Explore

Pada tahap ini dilakukan eksplorasi data dengan teknik statistik deskriptif maupun visualisasi. Tahap ini membantu peneliti mengidentifikasi pola mendasar, ketidakseimbangan kelas, atau potensi masalah yang mungkin memengaruhi pembuatan model. Dengan demikian, *Explore* berfungsi sebagai jembatan antara pembersihan data dan pemodelan, memastikan bahwa keputusan teknis yang diambil pada tahap berikutnya berbasis bukti[54].

4. Model

Data yang telah dieksplorasi kemudian digunakan untuk membangun model *machine learning* atau statistik sesuai kebutuhan penelitian. Model berfungsi menemukan pola lebih kompleks, membuat prediksi, atau melakukan klasifikasi. Tahap *Model* menggabungkan pemilihan model, pelatihan, evaluasi, dan optimasi, dengan fokus pada pengurangan ukuran model tanpa mengorbankan performa secara signifikan. Pemilihan algoritma harus disesuaikan dengan karakteristik data serta tujuan penelitian[54].

5. Interpret

Tahap akhir pada kerangka OSEMN adalah interpret yang berfokus pada analisis dan pemaknaan hasil yang diperoleh dari model yang telah dibangun[54]. Hasil analisis biasanya divisualisasikan melalui dashboard, laporan, atau metrik evaluasi (seperti akurasi, presisi, *recall, F1-score*). Interpretasi yang tepat akan menjembatani hasil teknis dengan kebutuhan praktis pengguna.

Secara keseluruhan, OSEMN memberikan kerangka kerja yang komprehensif dan terstruktur dalam penelitian berbasis data[10].

2.13 *Energy Based Scoring*

Model klasifikasi umumnya dilatih pada data dengan distribusi tertentu. Ketika menerima data yang berbeda dari distribusi pelatihan, model dapat menghasilkan prediksi yang tidak akurat. Kondisi ini disebut *out-of-distribution* (OOD). Untuk mengatasinya, digunakan metode *energy-based scoring* yang berfungsi mendeteksi apakah data termasuk dalam distribusi pelatihan (*in-distribution*) atau tidak. Metode energy-based scoring dikembangkan berdasarkan konsep *Energy-Based Model (EBM)*, yang berusaha memetakan setiap data masukan x ke suatu nilai energi $E_\theta(x)$. Nilai energi ini menggambarkan seberapa besar kemungkinan data tersebut termasuk dalam distribusi pelatihan. Data yang berasal dari distribusi pelatihan (*in-distribution*) akan memiliki nilai energi yang rendah, sedangkan data

yang asing atau tidak dikenal (*out-of-distribution*) akan memiliki nilai energi yang tinggi [9]. Secara matematis, nilai energi tersebut dapat didefinisikan sebagai:

$$E(x; f) = -T \cdot \log \sum_{i} exp\ (f_i(x)/T)$$

di mana $f_i(x)$adalah keluaran *logit* dari model untuk kelas ke-i, dan Tadalah parameter suhu (*temperature scaling*). Nilai ini dihitung langsung dari hasil keluaran model klasifikasi yang telah dilatih, sehingga metode ini tidak memerlukan model tambahan atau pelatihan ulang yang kompleks.

Berbeda dengan pendekatan tradisional yang menggunakan nilai *softmax confidence* sebagai indikator keyakinan model, energy-based scoring menggunakan nilai energi untuk menentukan apakah suatu data tergolong OOD atau tidak. Semakin besar nilai energi, semakin besar kemungkinan bahwa data tersebut tidak termasuk dalam distribusi pelatihan. Pendekatan ini terbukti lebih stabil karena tidak terlalu bergantung pada nilai probabilitas hasil *softmax* yang sering kali terlalu optimistik terhadap data asing[9]. Penelitian terkait *energy-based scoring* ini masih sangat minim, khususnya di Indonesia, sehingga kajian dan penerapan metode ini masih terbatas pada tingkat teoretis dan penelitian luar negeri. Sebagian besar penelitian dalam negeri yang berhubungan dengan deteksi data tidak normal (*anomaly detection*) masih menggunakan pendekatan konvensional seperti *Isolation Forest*, *Autoencoder*, atau *Support Vector Machine* (SVM), bukan model berbasis energi.

2.14 Analisis Performa Model

Analisis performa model menjadi tahap penting untuk mengevaluasi sejauh mana model mampu bekerja sesuai tujuan yang diharapkan. Pada bagian ini, dilakukan pengukuran kinerja model menggunakan berbagai metrik evaluasi guna melihat tingkat akurasi, kesalahan prediksi, serta kemampuan generalisasi model.

2.14.1 Analisis Inferensial Menggunakan Uji Dua Proporsi

Uji dua proporsi (*two-proportion Z-test*) merupakan metode statistik inferensial yang digunakan untuk menguji apakah terdapat perbedaan yang signifikan antara dua proporsi dari dua kelompok independen. Metode ini banyak digunakan dalam analisis data biner, di mana setiap observasi diklasifikasikan ke dalam dua kategori, seperti keberhasilan atau kegagalan. Dalam berbagai penelitian modern, termasuk analisis komparatif dan eksperimen, pengujian perbedaan dua proporsi menjadi pendekatan yang umum digunakan untuk mengevaluasi perbedaan performa antar kelompok.

Pendekatan ini didasarkan pada asumsi bahwa data berasal dari distribusi binomial dan, untuk ukuran sampel yang cukup besar, distribusi proporsi sampel dapat didekati dengan distribusi normal melalui Teorema Limit Tengah Central Limit Theorem. Dalam penelitian statistik modern, pendekatan ini sering dikaitkan dengan metode berbasis *normal approximation*, yang menjadi dasar dalam pengujian perbedaan dua proporsi[55]. Selain itu, pengujian perbedaan proporsi juga dapat dilakukan melalui pendekatan lain seperti *score test* dan *likelihood ratio test*, yang secara konseptual memiliki hubungan dengan formulasi uji Z klasik[56].

Hipotesis yang digunakan dalam uji dua proporsi adalah sebagai berikut:

Hipotesis nol (H_0): $p_1 = p_2$

Hipotesis alternatif (H_1): $p_1 \neq p_2$(dua sisi)

Statistik uji Z untuk membandingkan dua proporsi dirumuskan sebagai berikut:

$$z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})\left(\frac{1}{n_1}+\frac{1}{n_2}\right)}}$$

dengan:

$\hat{p}_1$= proporsi sampel pertama

$\hat{p}_2$= proporsi sampel kedua

n_1= jumlah sampel pertama

n_2= jumlah sampel kedua

$\hat{p}$= proporsi gabungan (*pooled proportion*)

Proporsi gabungan dihitung menggunakan:

$$\hat{p} = \frac{x_1 + x_2}{n_1 + n_2}$$

di mana $x_1$dan $x_2$merupakan jumlah keberhasilan pada masing-masing sampel. Proporsi gabungan digunakan sebagai estimasi peluang keberhasilan yang sama pada kedua kelompok di bawah hipotesis nol.

Nilai statistik Z yang diperoleh kemudian dibandingkan dengan distribusi normal standar untuk menentukan signifikansi perbedaan. Pada tingkat signifikansi 5% (α = 0,05), hipotesis nol ditolak apabila nilai absolut Z lebih besar dari 1,96, yang menunjukkan bahwa perbedaan proporsi yang diamati signifikan secara statistik. Selain pengujian hipotesis, analisis dua proporsi juga dapat diperluas ke dalam pembentukan interval kepercayaan untuk mengestimasi rentang perbedaan proporsi populasi. Pendekatan ini memberikan informasi tambahan mengenai ketidakpastian estimasi dan banyak digunakan dalam analisis statistik modern berbasis proporsi [56]

2.14.2 Analisis Ketidakpastian dan Estimasi Interval

Analisis ketidakpastian dalam statistika inferensial digunakan untuk mengukur tingkat kepercayaan terhadap suatu estimasi parameter populasi. Dua konsep utama yang umum digunakan adalah standar error dan interval kepercayaan. Standar error (standard error) merupakan ukuran penyebaran dari distribusi sampling suatu statistik, khususnya rata-rata. Nilai ini menunjukkan seberapa besar variasi estimasi terhadap parameter populasi yang sebenarnya. Standar error dipengaruhi oleh simpangan baku dan ukuran sampel, sehingga semakin besar

jumlah sampel maka nilai standar error akan semakin kecil dan estimasi menjadi lebih presisi[57].

Secara matematis, standar error dari rata-rata dirumuskan sebagai berikut[58].

$$SE = \frac{s}{\sqrt{n}}$$

di mana:

SE = standar error

s = simpangan baku sampel

n = ukuran sampel

Interval kepercayaan (confidence interval) adalah rentang nilai yang digunakan untuk mengestimasi parameter populasi dengan tingkat keyakinan tertentu, seperti 95%. Interval ini memberikan informasi yang lebih komprehensif dibandingkan estimasi titik karena memperhitungkan ketidakpastian dari data. Secara umum, interval kepercayaan untuk rata-rata dirumuskan sebagai berikut[59].

$$CI = \bar{x} \pm z \cdot \frac{s}{\sqrt{n}}$$

atau jika menggunakan distribusi t (untuk sampel kecil):

$$CI = \bar{x} \pm t \cdot \frac{s}{\sqrt{n}}$$

di mana:

$\bar{x}$= rata-rata sampel

z= nilai dari distribusi normal (misalnya 1,96 untuk 95%)

t= nilai dari distribusi t (berdasarkan derajat kebebasan)

s= simpangan baku sampel

n= jumlah sampel

Penggunaan interval kepercayaan sangat dianjurkan dalam analisis data modern karena mampu memberikan gambaran mengenai stabilitas dan reliabilitas hasil estimasi. Selain itu, pelaporan standar error dan interval kepercayaan meningkatkan transparansi dan kualitas interpretasi hasil penelitian[57].

2.14.3 Analisis Interpretabilitas

1. 90% Cumulative Feature Importance

Feature importance merupakan salah satu pendekatan utama dalam Explainable AI (XAI) untuk memahami kontribusi setiap variabel terhadap keputusan model. Pendekatan ini bertujuan mengidentifikasi fitur-fitur dominan yang paling berpengaruh dalam proses prediksi. Interpretasi model berbasis fitur memungkinkan identifikasi faktor utama yang memengaruhi hasil prediksi, sehingga membantu pengguna memahami alasan di balik keputusan model. Dalam konteks ini, tidak semua fitur memiliki kontribusi yang sama, sehingga diperlukan seleksi terhadap fitur yang paling signifikan[60].

Konsep *cumulative feature importance* merupakan pendekatan yang digunakan untuk mengidentifikasi fitur-fitur dominan dalam suatu model dengan cara mengakumulasi kontribusi masing-masing fitur hingga mencapai ambang tertentu, umumnya sebesar 90% dari total kontribusi. Pendekatan ini didasarkan pada asumsi bahwa sebagian besar keputusan model tidak ditentukan oleh seluruh fitur, melainkan hanya oleh sejumlah kecil fitur yang memiliki pengaruh paling besar. 90% Cumulative Feature Importance digunakan untuk menyaring fitur-fitur utama yang berkontribusi secara signifikan terhadap prediksi model. Fitur-fitur yang secara kumulatif mencapai atau melebihi 90% dari total nilai feature importance dikategorikan sebagai fitur dominan, sedangkan fitur lainnya dianggap memiliki kontribusi rendah atau bersifat redundan[60]. Dengan demikian, pendekatan ini tidak hanya membantu dalam memahami struktur keputusan model, tetapi juga berperan dalam penyederhanaan model tanpa mengorbankan performa secara signifikan.

2. Top 20 Feature Importance

Dalam analisis interpretabilitas model machine learning, identifikasi fitur-fitur paling berpengaruh menjadi langkah penting untuk memahami bagaimana model menghasilkan prediksi. Salah satu pendekatan yang umum digunakan adalah dengan mengurutkan fitur berdasarkan nilai kontribusinya, kemudian memilih sejumlah fitur teratas, seperti top 20 fitur paling penting (top 20 feature importance). Pendekatan ini bertujuan untuk memberikan gambaran ringkas mengenai struktur keputusan model tanpa harus menganalisis seluruh fitur yang jumlahnya besar [61].

Penggunaan top 20 fitur memberikan keseimbangan antara kompleksitas dan keterbacaan model. Dari banyak fitur yang dihasilkan model hanya sebagian kecil fitur yang memiliki kontribusi signifikan terhadap prediksi[61]. Hasil analisis menunjukkan bahwa:

- Top 20 fitur mampu merepresentasikan sebagian besar perilaku model,
- Sebagian besar fitur lainnya memiliki kontribusi yang sangat kecil,
- Terdapat konsistensi antara metode SHAP dan metode konvensional (gain), di mana 18 dari 20 fitur teratas memiliki kesamaan .

Hal ini menunjukkan bahwa pemilihan sejumlah fitur teratas, seperti top 20, merupakan pendekatan yang cukup representatif untuk memahami model tanpa kehilangan informasi penting.

2.15 *Flask*

Flask merupakan *micro web framework* berbasis Python yang digunakan untuk membangun aplikasi web secara ringan, fleksibel, dan mudah dikembangkan[62]. Flask menyediakan fungsi dasar seperti manajemen *routing*, *request handling*, dan *templating*, tanpa mengharuskan pengembang mengikuti struktur proyek yang kaku. Fleksibilitas ini menjadikan Flask populer dalam pengembangan prototipe maupun aplikasi yang memerlukan integrasi dengan model *machine learning* atau *deep learning*. Sebagai *microframework*, Flask tidak membawa banyak modul bawaan, melainkan memungkinkan pengembang menambahkan pustaka sesuai kebutuhan. Struktur minimalis ini membuat Flask mudah diadaptasi untuk berbagai tujuan, seperti membangun *RESTful API* yang dapat diakses oleh aplikasi eksternal melalui protokol HTTP [62]. Flask memiliki beberapa keunggulan yang menjadikannya banyak digunakan dalam penelitian maupun industri, antara lain:

1. Ringan dan modular : komponen dapat ditambahkan sesuai kebutuhan tanpa memengaruhi kinerja inti.
2. Dukungan API dan integrasi model : memudahkan penerapan model yang telah dilatih ke dalam layanan web atau aplikasi berbasis jaringan.
3. Mudah digunakan dan terdokumentasi dengan baik : komunitas Flask yang besar memungkinkan pengembang memperoleh dukungan, contoh kode, serta integrasi dengan pustaka populer seperti TensorFlow, PyTorch, dan scikit-learn.

Dari hal tersebut kelebihan utama Flask terletak pada kemudahan implementasi, skalabilitas, dan kesesuaian dengan proyek berukuran kecil hingga menengah. Selain itu, Flask memudahkan integrasi langsung dengan model yang ditulis dalam Python tanpa perlu framework tambahan. Namun, untuk sistem berskala besar dengan kebutuhan *multi-user* dan kompleksitas tinggi, Flask memerlukan penambahan pustaka eksternal untuk manajemen autentikasi, database, dan keamanan [63].

2.16 *Streamlit*

Streamlit adalah framework open-source berbasis Python yang dirancang untuk mempermudah pembuatan aplikasi web interaktif khusus untuk data science, machine learning, dan analisis data[64]. Dengan Streamlit, pengguna tidak perlu menulis HTML, CSS, atau JavaScript, cukup menulis kode Python biasa. Hal ini sangat menguntungkan bagi peneliti atau developer ML/AI yang ingin membuat antarmuka pengguna (UI) untuk model dengan cara yang lebih efektif dan sederhana [65]. Streamlit menyediakan berbagai komponen interaktif seperti tombol, *file uploader*, *slider*, dan *input text* yang dapat digunakan untuk membuat antarmuka pengguna dengan cepat. Selain itu, framework ini terintegrasi secara langsung dengan berbagai pustaka populer dalam ekosistem Python, seperti TensorFlow, PyTorch, Scikit-learn, NumPy, dan Pandas, sehingga memudahkan proses implementasi dan visualisasi hasil prediksi model [66].

Keunggulan utama Streamlit terletak pada kemudahan dan kecepatan pengembangannya. Framework ini memungkinkan proses *deployment* model dilakukan secara langsung tanpa perlu membuat arsitektur server yang kompleks. Selain itu, karena bersifat ringan, Streamlit cocok digunakan untuk membuat prototipe aplikasi maupun sistem yang membutuhkan *real-time interaction* antara pengguna dan model[67]. Streamlit sangat sesuai untuk digunakan dalam penelitian yang berfokus pada implementasi model berbasis *machine learning*.

Meskipun memiliki banyak kelebihan, Streamlit juga memiliki beberapa keterbatasan. Framework ini lebih sesuai digunakan untuk aplikasi berskala kecil hingga menengah, karena belum dirancang untuk menangani sistem dengan jumlah pengguna besar atau kebutuhan *backend* yang kompleks seperti manajemen basis data dan autentikasi pengguna. Selain itu, antarmuka Streamlit bersifat sederhana sehingga kurang fleksibel jika dibutuhkan desain web yang lebih kompleks dan dinamis [66].

2.17 Perangkat Lunak (*Software*)

Perangkat lunak atau *software* merupakan komponen penting dalam sistem komputer yang berfungsi sebagai perantara antara pengguna (*user*) dan perangkat keras (*hardware*). *Software* memungkinkan pengguna untuk mengoperasikan dan mengendalikan perangkat keras secara efektif melalui antarmuka yang telah dirancang sedemikian rupa. Dalam konteks penelitian ini, berbagai jenis perangkat lunak dimanfaatkan untuk menunjang kelancaran proses penelitian, mulai dari pengolahan data, visualisasi hasil, hingga pengembangan sistem atau aplikasi yang menjadi objek kajian.

2.17.1 Python

Python adalah bahasa pemrograman yang sederhana namun sangat kuat. Karena sintaksnya yang mudah dipahami, Python menjadi pilihan banyak orang, baik pemula maupun profesional. Dalam dunia pengolahan data dan kecerdasan buatan, Python sangat diandalkan karena memiliki banyak pustaka pendukung seperti NumPy, Pandas, dan Matplotlib. Dalam penelitian ini, Python digunakan sebagai dasar untuk membangun sistem, mengolah data, serta membuat dan menguji model AI yang dikembangkan.

2.17.2 Google Drive

Google Drive adalah layanan penyimpanan berbasis cloud dari Google yang memungkinkan pengguna menyimpan, mengelola, dan berbagi file secara daring. Dengan kapasitas penyimpanan yang cukup besar serta integrasi dengan berbagai layanan Google lainnya, Google Drive menjadi solusi praktis untuk kolaborasi dan akses data kapan saja serta di mana saja. Dalam penelitian ini, Google Drive digunakan sebagai media penyimpanan dataset, hasil pelatihan model, serta dokumen pendukung lainnya. Integrasinya dengan Google Colab juga mempermudah proses pemanggilan data secara langsung saat pengolahan maupun pelatihan model.

2.17.3 Google Colab

Google Colab adalah layanan berbasis cloud yang sangat membantu, terutama bagi pengguna yang tidak memiliki perangkat dengan spesifikasi tinggi. Di sini, pengguna bisa menjalankan kode Python langsung di browser, lengkap dengan akses ke GPU dan TPU secara gratis. Ini tentu sangat menguntungkan ketika melatih model yang membutuhkan waktu dan sumber daya besar. Google Colab juga terintegrasi dengan Google Drive, sehingga proses penyimpanan data dan kerja sama tim jadi lebih praktis. Dalam penelitian ini, Google Colab menjadi lingkungan kerja utama untuk proses coding dan pelatihan model.

2.17.4 Tensorflow

TensorFlow adalah library *open-source* yang banyak digunakan dalam pengembangan kecerdasan buatan, khususnya deep learning. Library ini mendukung komputasi di berbagai perangkat, dan juga menyediakan antarmuka Keras yang membuat proses pembuatan model menjadi lebih cepat dan mudah. TensorFlow cocok digunakan untuk proyek berskala besar, namun tetap fleksibel bagi kebutuhan skala kecil. Dalam penelitian ini, TensorFlow menjadi salah satu pilihan framework dalam membangun dan menguji model deteksi objek berbasis gambar.

2.17.5 Keras

Keras merupakan library high-level yang berjalan di atas TensorFlow dan dirancang untuk mempermudah pembangunan model deep learning. Dengan antarmuka yang sederhana, modular, dan intuitif, Keras memungkinkan peneliti merancang neural network secara cepat tanpa harus menuliskan kode yang terlalu kompleks. Library ini juga menyediakan berbagai lapisan, *optimizers*, dan fungsi aktivasi yang siap digunakan. Keunggulan Keras terletak pada kemampuannya melakukan prototyping dengan cepat, sehingga peneliti dapat menguji berbagai arsitektur model sebelum menentukan yang paling optimal. Dalam penelitian ini,

Keras digunakan sebagai kerangka kerja utama dalam membangun arsitektur model serta melakukan proses training dan evaluasi.

2.17.6 Jupyter Notebook

Jupyter Notebook adalah aplikasi open-source berbasis web yang memungkinkan peneliti menulis kode program, menampilkan hasil eksekusi, menambahkan penjelasan berupa teks, serta menyisipkan visualisasi dalam satu dokumen interaktif. Alat ini sangat berguna untuk eksplorasi data, dokumentasi penelitian, maupun eksperimen awal sebelum implementasi penuh. Jupyter Notebook juga mendukung berbagai bahasa pemrograman, namun Python adalah yang paling dominan digunakan. Dalam penelitian ini, Jupyter Notebook dimanfaatkan sebagai sarana untuk melakukan eksperimen awal, uji coba preprocessing, serta mencatat hasil pengujian model secara sistematis sebelum dipindahkan ke Google Colab untuk proses pelatihan skala besar.

2.17.7 Github

GitHub adalah platform berbasis web untuk menyimpan, mengelola, dan berbagi kode program yang menggunakan sistem *version control* berbasis Git. Melalui GitHub, setiap perubahan kode terekam secara rinci siapa, kapan, dan apa yang diubah sehingga memungkinkan kolaborasi yang transparan dan terstruktur. Bagi penelitian maupun pengembangan perangkat lunak, GitHub mendukung kolaborasi tim lintas lokasi melalui fitur-fitur seperti *repository*, *branching*, dan *pull request*.

2.18 Penelitian Terdahulu

Penelitian ini menggunakan beberapa studi literatur sebagai referensi untuk menambah pemahaman dan mengembangkan wawasan baru dari penelitian sebelumnya. Beberapa penelitian penting yang menjadi dasar pembuatan penelitian ini terdapat pada tabel 3.

Tabel 3. Penelitian Terdahulu

Peneliti	Relevansi Penelitian	Perbedaan
Indra Bakti & Mohamad Firdaus (2023)	klasifikasi 4 kelas menggunakan Inception-ResNet-V2; mereka menggunakan 4.000 citra dan akurasi 98%; relevan sebagai validasi efektivitas arsitektur residual pada klasifikasi multi kelas	Dataset 4.000 citra X-ray grayscale; tidak ada implementasi web
Marina Litvak (2022)	Klasifikasi 15 spesies dari 1.500 gambar; menggunakan transfer learning bertahap	Tidak menggunakan satu backbone spesifik;; tanpa implementasi sistem
Yo-Ping Huang & Haobijam Basanta (2021)	760 gambar + augmentasi; 29 kelas; menggunakan Inception-ResNet-V2; akurasi 98,39%; sama-sama menggunakan augmentasi untuk meningkatkan generalisasi	29 kelas (lebih banyak); juga melakukan deteksi objek
Mostafa Ahmed & Ali Ahmed (2023)	2.631 gambar; ResNet & InceptionResNet; akurasi 99,62% dan 100% (augmented); menunjukkan efektivitas residual network dan augmentasi	Objek daun lebih besar dan pola kontras jelas
Yunfeng Chen (2022)	Menggunakan Inception-ResNet + self-attention; akurasi 88,23%; menunjukkan backbone dapat dikembangkan lebih lanjut	Menggunakan attention mechanism; teknik augmentasi offline; domain CT grayscale
Vallent A.T. Kurniawan (2024)	3.264 citra; 4 kelas; InceptionResNetV2 + transfer learning + augmentasi; akurasi 96,63%;	Dataset lebih besar; citra medis
Prima Nugraha (2022)	49.282 gambar; 325 kelas; menggunakan Inception-	Dataset sangat besar; fokus deteksi + klasifikasi;

	ResNet-V2; sistem identifikasi otomatis spesies hewan	tanpa implementasi web
Muhammad Ichwan & Hanifah Sumantri (2024)	614 gambar; 6 kelas; InceptionResNetV2 akurasi 91,87%; mendukung performa arsitektur pada dataset kecil-menengah	Tidak disebutkan penggunaan teknik augmentasi on-the-fly; objek statis tanpa variasi pose kompleks
Febri Yalda Sulistia & Arie Vatresia (2024)	1.200 citra; 2 kelas; menunjukkan pentingnya pemilihan arsitektur; skala dataset hampir sama dengan 1.280 citra	Menggunakan ResNet-50; akurasi 55%
Fitriana Masruroh (2023)	4.500 gambar; 5 kelas; InceptionResNetV2 akurasi 92,00%; mendukung pemilihan arsitektur yang sama	Studi komparatif; tidak fokus pada augmentasi on-the-fly; tidak ada implementasi sistem
William Kelley (2021)	3 spesies lebah; akurasi 91%; domain paling dekat dengan penelitian Anda; sama-sama klasifikasi lebah berbasis deep learning	Menggunakan Faster R-CNN (object detection); tidak menggunakan InceptionResNetV2
Rico Yoga Pradana (2024)	Menggunakan pendekatan OSEMN seperti penelitian Anda; akurasi 98%; relevan secara metodologi pipeline	Data numerik; Random Forest; tidak ada CNN dan augmentasi citra
Slamet Cahyo Edy Sahputro & Beny Riswanto (2025)	35.887 gambar; 7 kelas; membahas overfitting pada CNN; relevan dengan penggunaan augmentasi untuk generalisasi pada dataset 1.280 citra	Tidak menggunakan InceptionResNetV2
Qorry Aina Fitroh & Shofwatul ‘Uyun (2023)	2.000 gambar; 2 kelas; transfer learning meningkatkan akurasi hingga 94%; mendukung strategi fine tuning	Menggunakan VGG16 & ResNet50; klasifikasi biner
Ibrahem Kandel & Mauro Castelli (2020)	220.000 citra; batch size 16 dan learning rate 0,0001 menghasilkan AUC 0,9677; relevan untuk konfigurasi dan pipeline training model	Fokus eksperimen hyperparameter

Beberapa penelitian sebelumnya telah menunjukkan bahwa arsitektur deep learning, khususnya InceptionResNet-V2, dapat digunakan secara efektif dalam klasifikasi citra pada berbagai bidang. Model ini terbukti mampu mengklasifikasikan pneumonia dari citra rontgen dada dengan akurasi mencapai 98%, serta presisi tinggi pada masing-masing kelas seperti COVID dan Lung Opacity [31]. Selain itu, penggunaan arsitektur yang sama pada klasifikasi CT scan paru-paru penderita COVID-19 juga menunjukkan kinerja yang baik, meskipun dengan pendekatan tambahan seperti self-attention dan menghasilkan akurasi sebesar 88,23% [45]. Temuan ini menegaskan bahwa InceptionResNet-V2 efektif dalam konteks medis yang kompleks dan membutuhkan tingkat akurasi tinggi.

Dalam ranah klasifikasi objek biologis non-medis, model InceptionResNet-V2 juga menunjukkan performa yang sangat baik dengan akurasi mencapai 98,39% dalam identifikasi burung endemik. Pada bidang pertanian, arsitektur ResNet dan InceptionResNet mampu mendeteksi penyakit pada daun pohon palem dengan akurasi hingga 100% setelah dilakukan augmentasi data [44]. Selain itu, model InceptionResNet-V2 juga menunjukkan kinerja yang unggul dalam klasifikasi kualitas biji kakao dengan akurasi sebesar 91,87% dan F1-score sebesar 92,([60]. Hal ini menunjukkan bahwa arsitektur ini memiliki potensi besar dalam meningkatkan efisiensi pada sektor pertanian dan pengolahan hasil bumi.

Secara keseluruhan, berbagai penelitian tersebut memperlihatkan konsistensi performa tinggi dari arsitektur InceptionResNet-V2 dalam beragam konteks klasifikasi citra, baik medis maupun non-medis. Oleh karena itu, arsitektur ini memiliki potensi yang kuat untuk diterapkan dalam klasifikasi lebah madu tanpa sengat, yang memiliki karakteristik visual yang kompleks dan kemiripan morfologi antar kelas. Dengan kemampuan ekstraksi fitur yang mendalam, model ini diharapkan mampu meningkatkan akurasi klasifikasi serta mendukung penelitian di bidang keanekaragaman hayati dan konservasi serangga.

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat

Penelitian dan pembuatan skripsi ini dilakukan pada :

1. Waktu Penelitian : Juli – Desember 2025
2. Tempat : Lembah Suhita dan Laboratorium Komputer Jurusan Teknik Elektro Universitas Lampung

Penelitian akan dilakukan berdasarkan jadwal pada table berikut :

Tabel 4. Waktu Penelitian

No	Aktivitas	2025					
		Juli	Agu	Sep	Okt	Nov	Des
1	Perencanaan						
2	Studi literatur						
3	Persiapan alat dan bahan						
4	Proses pembuatan model (OSEMN)						
5	Penyusunan laporan						

3.2 Alat dan Bahan

3.2.1 Alat

Dalam penelitian ini, alat yang digunakan terdiri dari perangkat keras (*hardware*) dan perangkat lunak (*software*) sebagai pendukung utama dalam proses perancangan dan implementasi sistem. Berikut adalah perangkat keras (*hardware*) yang digunakan pada penelitian ini.

Tabel 5. Hardware

No	Hardware	Spesifikasi	Fungsi
1	Laptop	a. (Dell Latitude E7470) b. Storage: 238 GB c. RAM: 8.00 GB d. Processor : Intel(R) Core(TM) i5- 6300U CPU @ 2.40GHz (2.50 GHz) e. Operating system : Windows 11 / 64-bit	Olah data, pembuatan model dan laporan
2	Kamera Ponsel	a. (iPhone Xr) b. Resolusi: 12 MP c. Lensa: Wide-angle d. Aperture (bukaan): f/1.8 e. Fitur: PDAF (Phase Detection Autofocus), OIS (Optical Image Stabilization), Quad-LED dualtone flash, HDR f. Perekaman Video: 4K hingga 60fps, 1080p hingga 240fps, HDR, stereo sound recording	Mengambil lebah secara langsung
3	Flashdisk	Kapsitas 64 GB	Menyimpan cadangan data

Untuk perangkat lunak (*software*) yang digunakan untuk penelitian ini dijelaskan pada tabel berikut.

Tabel 6. Software

No.	Software	Versi	Keterangan
1.	*Python*	3.13.4	Bahasa pemrograman yang digunakan dalam pembuatan dan pengembangan algoritma klasifikasi. Python mendukung berbagai *library* untuk pemrosesan data, visualisasi, hingga pembuatan model machine learning.
2	*Google Colab*	-	Platform cloud berbasis Jupyter Notebook dari Google yang memungkinkan eksekusi kode Python tanpa instalasi lokal. Mendukung berbagai *library* populer seperti TensorFlow dan PyTorch, serta mempermudah kolaborasi dan eksperimen berbasis GPU/TPU.

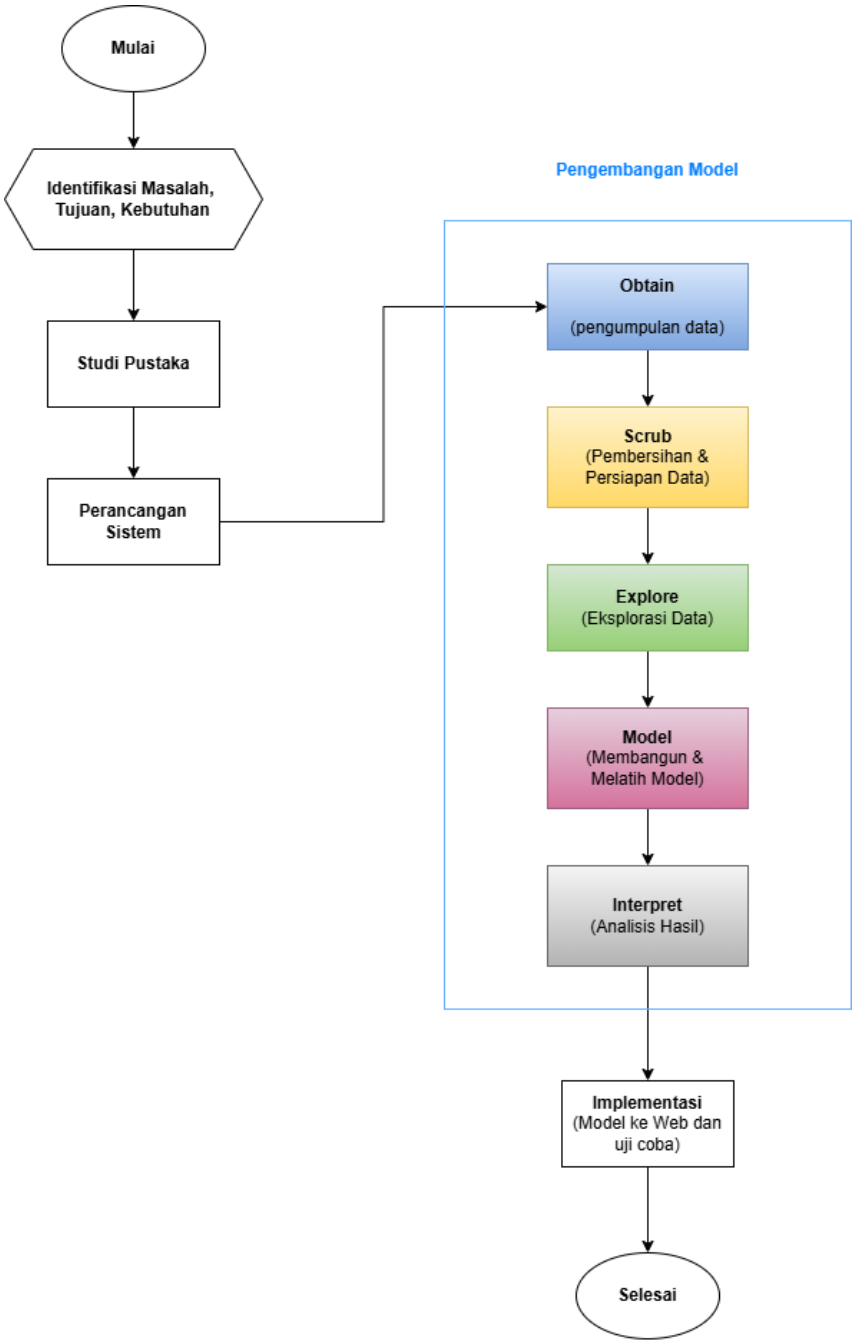
3	*Google Drive*	-	Digunakan sebagai media penyimpanan dataset citra lebah madu tanpa sengat yang dikumpulkan dari lapangan. Dengan penyimpanan cloud, data dapat diakses kapan saja dan terintegrasi langsung dengan Google Colab untuk proses pelatihan model.
4	*Jupiter Notebook*	Built-in environment	Lingkungan interaktif untuk preprocessing citra, pelatihan model, dokumentasi eksperimen, dan visualisasi hasil.
5	*TensorFlow*	TensorFlow 2.18.0 (pre-installed di Colab)	*Framework* open-source untuk komputasi numerik dan machine learning skala besar. Digunakan dalam pembuatan dan pelatihan model jaringan saraf secara efisien.
6	*Keras*	Keras 3.8.0 (pre-installed di Colab)	Dimanfaatkan sebagai *framework* utama dalam pembangunan dan pelatihan model deep learning dengan arsitektur *Inception-ResNetV2*. Keras menyediakan API yang sederhana namun powerful, sehingga memudahkan proses implementasi *transfer learning* untuk klasifikasi spesies lebah.
7	*Github*	Layanan Cloud (2025)	Berfungsi sebagai repositori kode penelitian yang menyimpan script Python, konfigurasi model, dan dokumentasi eksperimen. GitHub juga mendukung *version control* sehingga memudahkan pengelolaan perubahan kode serta memungkinkan kolaborasi dan replikasi penelitian oleh peneliti lain.
8	*Draw.io*	2025 (web-based)	Digunakan untuk membuat diagram alur penelitian, arsitektur model, serta rancangan sistem identifikasi lebah tanpa sengat. Visualisasi ini membantu memperjelas tahapan metodologi dan hubungan antar komponen penelitian.
9	*OpenCV*	4.11.0	*Library* vision yang digunakan untuk pemrosesan dan analisis citra digital. Mendukung pengolahan gambar, deteksi objek, dan berbagai operasi terkait computer vision.

3.2.2 Bahan

Bahan pada penelitian ini berupa data citra lebah madu tanpa sengat yang dikumpulkan secara langsung dari lapangan. Data citra diperoleh dari dokumentasi visual spesies lebah tanpa sengat yang terdapat di Lembah Suhita, menggunakan kamera ponsel dengan format file dominan berupa JPG dan PNG. Data ini digunakan sebagai dataset utama untuk pelatihan dan pengujian model klasifikasi berbasis deep learning *Inception-ResnetV2*.

3.3. Alur Penelitian

Penelitian ini mengadopsi pendekatan data mining yang mengacu pada kerangka kerja OSEMN (Obtain, Scrub, Explore, Model, Interpret) dalam pembuatan modelnya untuk memastikan proses berjalan secara sistematis dan terstruktur. Berikut adalah diagram dari alur penelitian ini.



Gambar 13. Alur Penelitian

Diagram alur kerja diatas (gambar 13) merupakan adalah tahapan keseluruhan dari proses identifikasi hingga implementasi model yang telah dibuat[68]. Berikut adalah penjabaran tiap tahapannya.

1. Identifikasi Masalah, Tujuan, dan Kebutuhan

Tahap awal penelitian dilakukan dengan merumuskan permasalahan utama, yaitu kesulitan dalam membedakan spesies lebah madu tanpa sengat secara kasat mata karena ukuran tubuh yang kecil dan kemiripan morfologi. Dari permasalahan ini ditentukan tujuan penelitian, yaitu merancang model klasifikasi berbasis citra digital yang dapat digunakan tanpa tenaga ahli maupun alat khusus.

2. Studi Pustaka

Selanjutnya dilakukan studi pustaka, yaitu menghimpun referensi ilmiah serta teori yang relevan terkait objek penelitian, teknik *image processing*, arsitektur dan implementasi model, serta teknik lain yang digunakan dalam penelitian ini. Tahap ini penting untuk membangun landasan teori yang kuat.

3. Pengembangan model

Tahap ini Adalah tahap model dibangun dengan *framework* OSEMN agar alur pengembangan model menjadi terstruktur dan mudah dipahami setiap tahapnya. Proses dimulai dari pengumpulan data (*Obtain*) hingga evaluasi model (*interpret*).

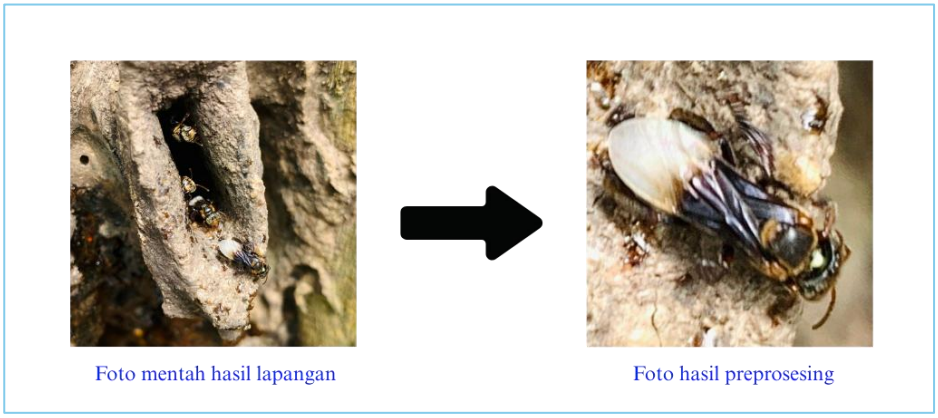
a. Obtain

Tahap *obtain* merupakan proses awal dari tahapan membangun model yang dimulai dari pengumpulan seluruh data citra yang diperlukan untuk membangun sistem klasifikasi. Pada tahap ini dilakukan pengambilan data citra secara langsung dengan kamera ponsel di lembah suhita, pengunduhan data citra pendukung dari iNaturalis dan Flickr. Pada penelitian ini, dari total dua belas spesies lebah madu tanpa sengat yang terdapat di Lembah Suhita, hanya empat spesies yang digunakan sebagai objek penelitian. Pemilihan empat spesies ini dilakukan secara selektif berdasarkan beberapa pertimbangan teknis dalam proses pengumpulan data citra. Alasan Utama keempat spesies tersebut dipilih karena pada saat proses pengumpulan data hanya keempat spesies tersebut yang aktif di sarangnya dan sedang tidak dalam kondisi perawatan sehingga boleh dan dapat dilakukan pemotretan untuk pengambilan data. Keempat spesies ini juga lebih mudah dan mendukung untuk didokumentasikan menggunakan kamera, baik dari

segi ukuran tubuh, lingkungan dan perilaku lebahnya. Dengan pertimbangan tersebut, penggunaan empat spesies ini diharapkan dapat menghasilkan data citra yang lebih representatif dan optimal untuk proses pelatihan model klasifikasi berbasis deep learning.

b. Scrub

Tahap *scrub* dilakukan untuk membersihkan dan menyiapkan data agar layak digunakan dalam pelatihan model. Proses ini mencakup seleksi gambar untuk membuang citra yang rusak atau tidak relevan, pemotongan area citra (cropping) untuk memfokuskan objek utama, pelabelan data dengan mengelompokan data pada folder di Google Drive sesuai kelas spesiesnya, serta pra-pemrosesan seperti *resize*, *rename*, dan *padding*.



Gambar 14. Contoh data pre prosesing

Setelah seluruh data bersih dan seragam, dataset kemudian dibagi (*spiliting dataset*) menjadi data pelatihan, validasi, dan pengujian.

c. Explore

Tahap berikutnya adalah *explore*, yaitu eksplorasi data guna memahami karakteristik dataset yang akan dipakai. Pada tahap ini dilakukan analisis distribusi jumlah gambar per kelas, visualisasi sebaran data, serta pemeriksaan format dan dimensi citra. Eksplorasi ini membantu memastikan bahwa data berada dalam kondisi yang stabil dan tidak memiliki masalah yang dapat memengaruhi proses pelatihan.

d. Model

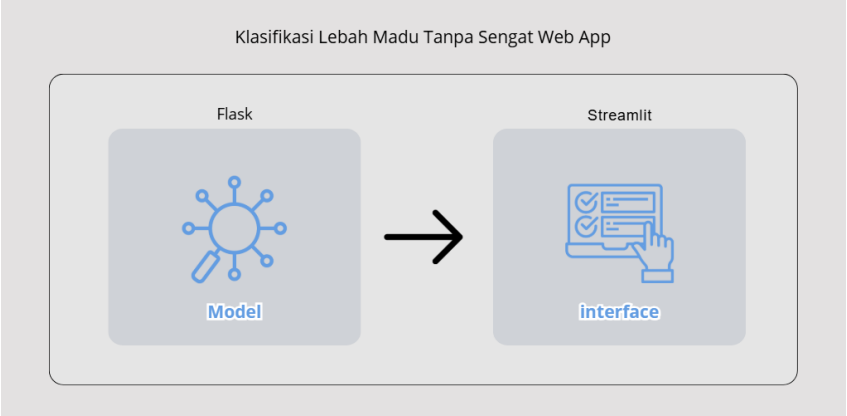
Pada tahap *model*, proses pembangunan model dilakukan mulai dari menyesuaikan lingkungan perangkat pelatihan yang digunakan, penyesuaian arsitektur model, konfigurasi hyperparameter, hingga proses pelatihan model pada dataset yang telah dipersiapkan. Tahap ini juga mencakup penyajian hasil pelatihan berupa grafik akurasi dan loss, serta pengujian model untuk mengetahui kemampuan generalisasinya terhadap data yang belum pernah dilihat sebelumnya

e. Interpret

Tahap terakhir dalam OSEMN adalah interpret, yaitu interpretasi terhadap performa model yang telah dibangun. Analisis dilakukan berdasarkan metrik evaluasi seperti akurasi, precision, recall, dan error yang muncul pada klasifikasi. Selain itu, dianalisis pula kelemahan model serta kesesuaiannya dengan tujuan penelitian. Hasil interpretasi ini menjadi dasar dalam menentukan apakah model sudah layak diterapkan pada sistem dan dapat diintegrasikan ke tahapan implementasi aplikasi.

4. Implementasi

Pada tahap ini, model terbaik hasil pelatihan disimpan dalam format *.keras* agar dapat digunakan kembali tanpa perlu pelatihan ulang. Model tersebut kemudian dipanggil kembali dan dihubungkan dengan aplikasi web sebagai backend API dengan *flask*.



Gambar 15. Implementasi Model

Melalui web, pengguna dapat mengunggah citra lebah madu tanpa sengat, kemudian sistem memproses citra tersebut menggunakan model dan menampilkan hasil klasifikasi dengan antarmuka web yang dikembangkan menggunakan *streamlit*. Dengan demikian, penelitian tidak hanya menghasilkan model, tetapi juga memberikan sarana praktis bagi pengguna untuk memanfaatkan hasilnya secara langsung.

V. PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan pada penelitian yang telah dilakukan, maka disimpulkan bahwa:

1. Model klasifikasi lebah madu tanpa sengat berbasis Inception-ResNetV2 berhasil dibangun dan diimplementasikan dengan performa akurasi akhir 92.5%.
2. Proses *fine tuning* terbukti meningkatkan performa secara signifikan, ditunjukkan dengan kenaikan akurasi validasi dari 79,17% pada *tahap feature extraction* menjadi 97,92% setelah *fine tuning*.
3. Model memiliki kemampuan generalisasi yang baik dengan akurasi hasil pelatihan sebesar 97,92% dan pengujian sebesar 92,5%. Berdasarkan hasil evaluasi performa dan analisis statistik, model tidak mengalami overfitting.

5.2 Saran

Berdasarkan hasil dan temuan penelitian yang telah dilakukan, berikut saran untuk penelitian sejenis:

1. Peningkatan Jumlah dan Variasi Dataset untuk Memperkuat Generalisasi.
   Mengingat terdapat selisih akurasi antara validation dan test serta ukuran data uji yang masih terbatas, penelitian selanjutnya disarankan menambah jumlah dan variasi dataset.
2. Pengembangan Prapemrosesan dan Segmentasi Objek
   Analisis confusion matrix menunjukkan bahwa kesalahan klasifikasi masih terjadi pada spesies dengan kemiripan morfologi tinggi. Oleh karena itu, tahap prapemrosesan dapat dikembangkan melalui segmentasi objek dan

penghilangan latar belakang (background removal) agar model lebih fokus pada fitur morfologi lebah dan tidak terpengaruh oleh elemen latar.

3. Peningkatan Stabilitas Evaluasi dan Interpretabilitas Model.

   Untuk memperoleh estimasi performa yang lebih robust, penelitian selanjutnya dapat menerapkan teknik seperti k-fold cross-validation untuk memahami bagian citra yang memengaruhi keputusan model, sehingga kesalahan prediksi dapat dianalisis dan diperbaiki secara lebih terarah.

4. Optimalisasi Strategi Augmentasi Data

   Mengingat model menunjukkan fitting yang sangat kuat pada data training, strategi augmentasi yang lebih terstruktur termasuk augmentasi statis (offline augmentation) dapat dieksplorasi guna meningkatkan robustness model terhadap variasi data uji dan meminimalkan potensi overfitting.

DAFTAR PUSTAKA

[1] U. Harmain, J. Rudiantho Saragih, M. M. Simarmata, and M. P. J Pasaribu, "Sosialisasi budidaya lebah madu tanpa sengat (stingless bee) dan manfaatnya," *Jurnal Pengabdian Masyarakat*, vol. 2, no. 2, Sep. 2022.

[2] Relung Indonesia Foundation, "Mengenal Lebah Kelulut." Accessed: Jul. 08, 2025. [Online]. Available: https://relungindonesia.org/2024/04/mengenal-lebah-kelulut/?

[3] Food and Agriculture Organization of the United Nations (FAO), "Global Action on Pollination Services for Sustainable Agriculture." Accessed: May 27, 2025. [Online]. Available: www.fao.org/pollination/about/

[4] M. S. Engel, S. Kahono, and D. Peggie, "a key to the genera and subgenera of stingless bees in Indonesia (hymenoptera: apidae)," 2018.

[5] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," 2017. Accessed: Sep. 02, 2025. [Online]. Available: doi.org/10.1609/aaai.v31i1.11231

[6] A. Kensert, P. J. Harrison, and O. Spjuth, "Transfer Learning with Deep Convolutional Neural Networks for Classifying Cellular Morphological Changes," *SLAS Discovery*, vol. 24, no. 4, pp. 466–475, Apr. 2019, doi: 10.1177/2472555218818756.

[7] Y. P. Huang and H. Basanta, "Recognition of endemic bird species using deep learning models," *IEEE Access*, vol. 9, pp. 102975–102984, 2021, doi: 10.1109/ACCESS.2021.3098532.

[8] V. A. T. Kurniawan, E. C. Niswary, C. S.k.aditya, and D. R. Chandranegara, "Brain tumor classification using InceptionResNet-V2 and transfer learning approach," *JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer)*, vol. 10, no. 1, pp. 91–99, Jul. 2024, doi: 10.33480/jitk.v10i1.5223.

[9] W. Liu, X. Wang, J. D. Owens, and Y. Li, "Energy Based Out of Distribution Detection." [Online]. Available: https://github.com/wetliu/

[10] B. Riswanto and S. Cahyo Edy Sahputro, “An analysis of the effectiveness of KAN and CNN algorithms for human facial emotion classification,” *bit-Tech*, vol. 8, no. 1, pp. 1028–1038, Aug. 2025, doi: 10.32877/bt.v8i1.2819.

[11] Claus. Rasmussen, *Catalog of the Indo-Malayan/Australasian stingless bees (Hymenoptera:Apidae:Meliponini)*. Magnolia Press, 2008.

[12] S. Melia *et al.*, “Profile of stingless bee honey and microbiota produced in West Sumatra, Indonesia, by several species (Apidae, Meliponinae),” *Vet. World*, vol. 17, no. 4, pp. 785–795, Apr. 2024, doi: 10.14202/vetworld.2024.785-795.

[13] J. Miharja, T. Atmowidi, W. Priawandiputra, D. Perwitasari, and S. Kahono, “New Distribution Record of Tetrigona apicalis (Smith, 1857) (Hymenoptera: Apidae: Meliponini) in Ujung Kulon National Park, Indonesia,” *Hayati*, vol. 33, no. 1, pp. 1–7, Jan. 2026, doi: 10.4308/hjb.33.1.1-7.

[14] Rinda Mulyani, “Madu Suhita Targetkan Tembus Pasar Ekspor,” PortalLNews.id. Accessed: Feb. 25, 2026. [Online]. Available: https://portallnews.id/headline/madu-suhita-targetkan-pasar-ekspor/

[15] A. S. Rozman, N. Hashim, B. Maringgal, and K. Abdan, “A comprehensive review of stingless bee products: phytochemical composition and beneficial properties of honey, propolis, and pollen,” Jul. 01, 2022, *MDPI*. doi: 10.3390/app12136370.

[16] Z. F. Abror, “Klasifikasi citra kebakaran dan non kebakaran menggunakan convolutional neural network,” *Jurnal Ilmiah Teknologi dan Rekayasa*, vol. 24, no. 2, pp. 102–113, 2019, doi: 10.35760/tr.2019.v24i2.2389.

[17] R. Dijaya, *Buku Ajar Pengolahan Citra Digital*. Sidoarjo: UMSIDA Press, 2023.

[18] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. New York: Pearson, 2018.

[19] S. Yulina, “Penerapan haar cascade classifier dalam mendeteksi wajah dan transformasi citra grayscale menggunakan OpenCV,” *Jurnal Komputer Terapan*, vol. 7, no. 1, 2021, Accessed: Sep. 02, 2025. [Online]. Available: jurnal.pcr.ac.id/index.php/jkt/

[20] A. Arnita, F. Marpaung, F. Aulia, N. Suryani, R. Nabila, and Cyra, *Computer Vision dan Citra Digital*. Surabaya: Pustaka Aksara, 2022.

[21] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *J. Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00444-8.

[22] D. Wicaksono, P. Almeyda, I. Mikola, M. Putra, and L. Malihatuningrum, “Analisis perbandingan metode pra pemrosesan citra untuk deteksi tepi canny pada citra berbagai kondisi jalan menggunakan bahasa pemrograman python,” *Jurnal Teknologi dan Ilmu Komputer Prima (JUTIKOMP)*, 2024.

[23] A. A. Wirabudi, *Dasar Pengolahan Citra Digital menggunakan MATLAB*. Penamuda Media, 2024.

[24] M. I. Prasetya, I. Z. Yadi, Y. N. Kunang, and S. D. Permatasari, “Prapemrosesan untuk klasifikasi gambar aksara OKU Timur,” *Jurnal Teknologi Dan Sistem Informasi Bisnis*, vol. 7, no. 1, pp. 208–215, Feb. 2025, doi: 10.47233/jteksis.v7i1.1629.

[25] N. Rosmawarni, *Pemrosesan Citra (Image Processing)*. Penamuda Media, 2025.

[26] T. Adilah and Q. N. Azizah, “Klasifikasi tumor otak menggunakan ekstraksi fitur HOG dan support vector machine,” *Jurnal Infotech*, vol. 4, no. 1, 2022, Accessed: Sep. 02, 2025. [Online]. Available: ejournal.bsi.ac.id/ejurnal/index.php/infortech45

[27] J. Sanjaya and M. Ayub, “Augmentasi data pengenalan citra mobil menggunakan pendekatan random crop, rotate, dan mixup,” *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 6, no. 2, Aug. 2020, doi: 10.28932/jutisi.v6i2.2688.

[28] K. Malialis, D. Papatheodoulou, S. Filippou, C. G. Panayiotou, and M. M. Polycarpou, “Data augmentation on-the-fly and active learning in data stream classification,” Oct. 2022, doi: 10.1109/SSCI51031.2022.10022133.

[29] Ricky Putra Sardika and W. Widhiarso, “Klasifikasi Otomatis Tingkat Kerusakan Retak Bangunan pada Citra Digital Menggunakan MobileNetV2 dan Augmentasi Data,” *Arcitech: Journal of Computer Science and Artificial Intelligence*, vol. 5, no. 1, pp. 108–124, Jun. 2025, doi: 10.29240/arcitech.v5i1.13938.

[30] S. H. Sudirwo *et al.*, *Artificial Intelegence*. Jambi: PT. Sonpedia Publishing Indonesia, 2025.

[31] K. Athania Purba and T. Dewayanto, “Penerapan artificial intelligence, machine learning dan deep learning pada kurikulum akuntansi: A

systematic literature review," *DIPONEGORO JOURNAL OF ACCOUNTING*, vol. 12, pp. 1–15, 2023, [Online]. Available: http://ejournal-s1.undip.ac.id/index.php/accounting

[32] I. Bakti, M. Firdaus, and S. Artikel, "Arsitektur convolutional neural network InceptionResNet-V2 untuk pengelompokan pneumonia chest X-ray," *Jurnal Komputer dan Teknologi*, vol. 12530, no. 77, Jan. 2023, doi: 10.58290/jukomtek.

[33] J. Jamaaluddin and I. Sulistyowati, *Buku Ajar Mata Kuliah Kecerdasan Buatan(Artificial Intelligence)*. Sidoarjo: UMSIDA Press, 2021.

[34] J. W. G. Putra, *Intro to ML Secured*, 1.4. 2020.

[35] S. Nurmaini, A. Darmawahyuni, A. I. Saputri, M. N. Rachmatullah, F. Firdaus, and B. Tutuko, *Pengenalan Deep Learning dan Implementasinya*. Palembang: UPT. Penerbit dan Percetakan Universitas Sriwijaya, 2021.

[36] M. Litvak, S. Divekar, and I. Rabaev, "Urban plants classification using deep-learning methodology: A case study on a new dataset," *Signals*, vol. 3, no. 3, pp. 524–534, Aug. 2022, doi: 10.3390/signals3030031.

[37] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[38] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*, Institute of Electrical and Electronics Engineers Inc., Jul. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[39] MathWork, "Convolutional Neural Network." Accessed: Jul. 09, 2025. [Online]. Available: https://www.mathworks.com/discovery/convolutional-neural-network.html

[40] R. Cresson, *Deep Learning for Remote Sensing Images with Open Source Software*, 1st ed. Boca Raton: CRC Press, 2020.

[41] B. Zouin, J. Zahir, F. Baletaud, L. Vigliola, and S. Villon, "Improving CNN fish detection and classification with tracking," *Applied Sciences (Switzerland)*, vol. 14, no. 22, Nov. 2024, doi: 10.3390/app142210122.

[42] M. Malik Ibrahim, R. Rahmadewi, and L. Nurpulaela, "Pendeteksian nominal uang pada gambar menggunakan convolutional neural network:

Integrasi metode pra-pemrosesan citra dan klasifikasi berbasis CNN," *Jurnal Mahasiswa Teknik Informatika*, vol. 7, no. 2, 2023.

[43] F. Masruroh, B. Surarso, and B. Warsito, "Perbandingan kinerja Inception-ResNetV2, Xception, Inception-v3, dan ResNet50 pada gambar bentuk wajah," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 10, no. 1, pp. 11–20, Feb. 2023, doi: 10.25126/jtiik.2023104941.

[44] W. Setiawan, *Deep Learning Menggunakan Convolutional Neural Network: Teori dan Aplikasi*. Media Nusa Creative, 2020.

[45] M. Ahmed and A. Ahmed, "Palm tree disease detection and classification using residual network and transfer learning of inception ResNet," *PLoS One*, vol. 18, no. 3 March, Mar. 2023, doi: 10.1371/journal.pone.0282250.

[46] Y. Chen *et al.*, "Classification of lungs infected COVID-19 images based on inception-ResNet," *Comput. Methods Programs Biomed.*, vol. 225, Oct. 2022, doi: 10.1016/j.cmpb.2022.107053.

[47] Mathworks, "inceptionresnetv2." Accessed: Oct. 21, 2025. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/inceptionresnetv2.html

[48] V. Liana *et al.*, "Enzyme dosage detection to degrade feathers in edible bird's nests: A comparative convolutional neural networks study," *Advances in Food Science, Sustainable Agriculture and Agroindustrial Engineering*, vol. 6, no. 4, pp. 382–398, 2023.

[49] N. Kulathunga, N. R. Ranasinghe, D. Vrinceanu, Z. Kinsman, L. Huang, and Y. Wang, "Effects of nonlinearity and network architecture on the performance of supervised neural networks," *Algorithms*, vol. 14, no. 2, Feb. 2021, doi: 10.3390/a14020051.

[50] Y.-Y. Yen, S.-P. Weng, L.-J. Su, J.-H. Kao, and W.-C. Chu, "Application of the Inception-ResNet-V2 algorithm to the analysis of embryo microscope images for the prediction model of assisted reproduction," *Computer Science and Information Systems*, vol. 22, no. 4, pp. 1665–1685, 2025, doi: 10.2298/csis241001066y.

[51] P. Nugraha, A. Komarudin, E. Ramadhan, U. Jenderal, A. Yani, and C. Jl, "Deteksi objek dan jenis burung menggunakan convolutional neural network dengan arsitektur Inception-ResNet-V2," *infotech Journal*, Dec. 2022, doi: 10.31949/infotech.v8I2.2889.

[52] Dimas Saputra, Archamul Fajar Pratama, Muhammad Dawam Fakhri, Muhammad Ahsanur Rafi, and Fetty Tri Anggraeny, "Classification of insect pets in agriculture using Inception-ResNet-V2 architecture," *Antivirus : Jurnal Ilmiah Teknik Informatika*, vol. 19, no. 1, pp. 41–51, May 2025, doi: 10.35457/antivirus.v19i1.4107.

[53] R. Mahendra and E. Angga Laksana, "Pendekatan transfer learning dengan InceptionResNetV2 dan augmentasi mix-up untuk peningkatan klasifikasi tumor otak," *Jurnal Algoritma*, May 2025, doi: 10.33364/algoritma/v.22-1.2194.

[54] S. Saleem, N. Hasan, A. Khattar, P. R. Jain, T. K. Gupta, and M. Mehrotra, "DeLTran15: A deep lightweight transformer-based framework for multiclass classification of disaster posts on X," *IEEE Access*, vol. 12, pp. 153676–153693, 2024, doi: 10.1109/ACCESS.2024.3478790.

[55] G. Shan, X. Y. Lou, and S. S. Wu, "Continuity corrected wilson interval for the difference of two independent proportions," *Journal of Statistical Theory and Applications*, vol. 22, no. 1–2, pp. 38–53, Jun. 2023, doi: 10.1007/s44199-023-00054-8.

[56] Jia Zhou and Chang-Xing Ma, "Testing risk difference of two proportions for combined unilateral and bilateral data," Oct. 2025, [Online]. Available: http://arxiv.org/abs/2510.18834

[57] R. L. Wasserstein, A. L. Schirm, and N. A. Lazar, "Moving to a world beyond 'p < 0.05,'" Mar. 29, 2019, *American Statistical Association*. doi: 10.1080/00031305.2019.1583913.

[58] Douglas G Altman and J Martin Bland, "Standard deviations and standard errors," *Br. Med. J.*, vol. 331, 2005.

[59] S. Greenland *et al.*, "Statistical tests, P values, confidence intervals, and power: a guide to misinterpretations," *Eur. J. Epidemiol.*, vol. 31, no. 4, pp. 337–350, Apr. 2016, doi: 10.1007/s10654-016-0149-3.

[60] K. M. Tawsik Jawad, A. Verma, F. Amsaad, and L. Ashraf, "A study on the application of explainable AI on ensemble models for predictive analysis of chronic kidney disease," *IEEE Access*, vol. 13, pp. 23312–23330, 2025, doi: 10.1109/ACCESS.2025.3535692.

[61] Y. Nohara, K. Matsumoto, H. Soejima, and N. Nakashima, "Explanation of machine learning models using shapley additive explanation and application for real data in hospital," Dec. 2021.

[62] K. Salo, “Comparative study on Python web frameworks: Flask and Django,” Metropolia University of Applied Sciences, 2020.

[63] Elan Suherlan, Shindy Arti, Siti Nabilah, and Zahwah Hazimah, “Penerapan Flask Framework untuk deployment model machine learning dalam mendukung analisis adaptasi mahasiswa pada pembelajaran daring,” *PINTER : Jurnal Pendidikan Teknik Informatika dan Komputer*, vol. 9, no. 1, pp. 110–117, Jun. 2025, doi: 10.21009/pinter.9.1.15.

[64] Streamlit, “Streamlit documentation.” Accessed: Nov. 09, 2025. [Online]. Available: https://docs.streamlit.io/

[65] Codecacademy Team, “What is Streamlit? A Complete Guide for Building Data Apps.” Accessed: Nov. 09, 2025. [Online]. Available: https://www.codecademy.com/article/what-is-streamlit-?

[66] G. Ayu Syafarina and Zaenudin, “Implementasi Framework Streamlit Sebagai Prediksi Harga Jual Rumah Dengan Linear Regresi,” *Metik Jurnal*, vol. 7, 2023, doi: 10.47002/metik.v7i2.680.

[67] R. Buga *et al.*, “Streamlit application and Deep Learning model for brain metastasis monitoring after gamma knife treatment,” *Biomedicines*, vol. 13, no. 2, Feb. 2025, doi: 10.3390/biomedicines13020423.

[68] F. D. Dermawan and Supriyono, “Implementasi artificial intelligence dalam pembuatan website klasifikasi genre buku,” *bit-Tech*, vol. 7, no. 2, pp. 618–627, Dec. 2024, doi: 10.32877/bt.v7i2.1977.