BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Teori

2.1.1 Pengertian Sistem

Sistem adalah suatu kumpulan elemen yang saling berhubungan menjadi satu kesatuan untuk memudahkan pencapaian suatu tujuan dengan memanfaatkan data, bahan, atau energi, atau bisa disebut juga Pengertian Sistem adalah suatu unsur atau elemen yang saling bekerjasama untuk mencapai tujuan tertentu guna melaksanakan proses atau fungsi untuk mencapai tujuan sistem [7]. Element atau bagian yang berhubungan satu sama lain dan bekerja sama untuk mencapai suatu tujuan tertentu [8].

2.1.2 Pengertian Informasi

Informasi diperoleh dari sebuah kumpulan data yang telah diolah menjadi penting bagi mereka yang menerima informasi karena mereka dapat menggunakannya sebagai referensi untuk membuat keputusan yang berdampak baik secara langsung maupun tidak langsung. [9]

Informasi juga merupakan arti dan manfaat yang dihasilkan dari pengolahan data. [8].

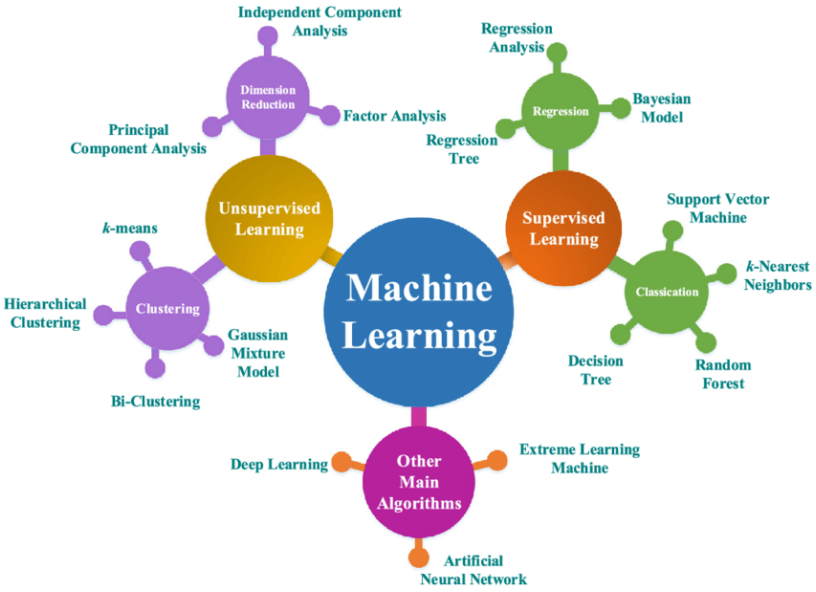
2.1.3 Pengertian Sistem Informasi

Sistem informasi adalah kumpulan perangkat yang memproses data menggunakan perangkat keras dan lunak komputer serta perangkat manusia. [9] Sistem informasi juga merupakan berbagai kumpulan komponen yang saling berhubungan, mengumpulkan atau mendapatkan informasi, memprosesnya, menyimpannya, dan menyebarkannya untuk membantu pengambilan keputusan dan pengawasan organisasi [8].

2.1.4 Pengertian *Machine Learning*

Machine Learning, juga dikenal sebagai mesin pembelajar, adalah cabang dari kecerdasan buatan yang berfokus pada belajar dari data. Fokusnya adalah membuat sistem yang dapat belajar secara mandiri tanpa perlu diprogram lagi oleh manusia. *Machine Learning* memerlukan sebuah data sebagai dasar bahan belajar yang akan digunakan pada saat proses *training* sebelum digunakan selama pengujian untuk mendapatkan hasil yang akurat dan optimal [3]. Secara umum *Machine Learning* di bagi menjadi 4, yaitu *supervised learning, unsupervised learning, semi unsupervised learning,* dan *reinforcement learning.*

a. *Supervised Learning*, merupakan teknik dalam *machine learning* dimana sebuah model dilatih dengan data yang telah diberi label, yang berarti setiap data memiliki masukan dan keluaran yang dikenal sebelumnya.
b. *Unsupervised Learning*, metode pembelajaran mesin dimana model dilatih dengan data tanpa label, yang berarti model tidak memiliki pasangan masukan dan keluaran yang sudah diketahui. Model bertugas mengetahui struktur atau pola yang tersembunyi dalam data.
c. Semi *Unsupervised Learning*, metode yang menggabungkan pendekatan *supervised* dan *unsupervised learning*, di mana sebagian data berlabel dan sebagian lagi tidak berlabel.
d. *Reinforcement Learning*, metode *machine learning* di mana agen (model) belajar untuk beradaptasi di lingkungan tertentu untuk memaksimalkan suatu reward (hadiah) yang dihasilkan dari tindakan tersebut. Berbeda dengan supervised atau unsupervised learning, dalam RL, model tidak diberi pasangan input-output yang jelas. Sebaliknya, ia belajar melalui proses coba-gagal dengan umpan balik (*feedback*) dari lingkungan.



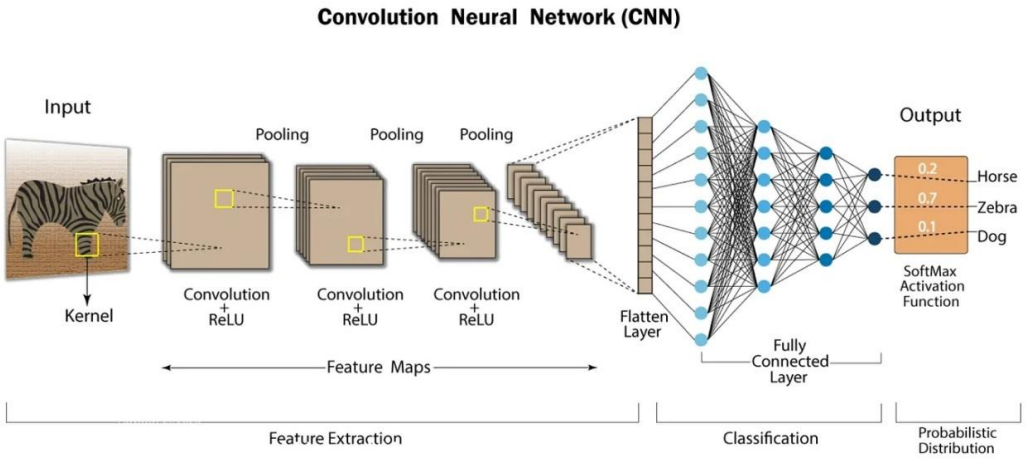
Gambar 2.1 Map Cakupan *Machine Learning*

2.1.5 Pengertian CNN (*Convolutional Neural Network*)

CNN (Convolutional Neural Network) adalah algroitma atau arsitektur dari pembelajaran mesin jaringan saraf tiruan yang sangat efektif dalam memproses data seperti citra gambar. CNN mempunyai kinerja yang lebih baik untuk menangani data gambar [10].

Salah satu jenis neural network yang sering digunakan untuk mengolah atau mengklasifikasikan pola dalam data gambar adalah convolutional neural network [4].

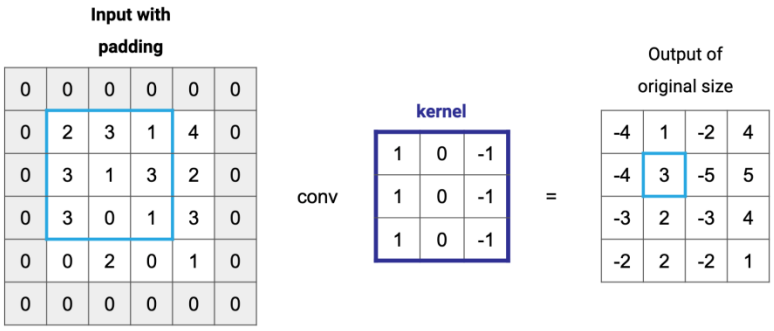
Convolutional Neural Network merupakan beberapa rangkaian lapisan proses yang berfungsi menjalankan operasi konvolusi, dan bertugas secara parallel, serta memiliki beberapa elemen. Algoritma ini terinspirasi dari sistem saraf biologis [11].



Gambar 2.2 *Layer Convolutional Nerural Network*

Pada gambar 2.2 menunjukkan bahwa data yang diinputkan mengalami beberapa proses konvolusi pada fitur extraksi (Feature Extraction). Fitur, ini merupakan elemen-elemen didalam gambar yang diextraksi oleh filter, dan dijadikan kedalam fitur map (Feature Maps). Setelah melalui fase extraksi fitur, terdapat fase klasifikasi (Classification), yang mana fitur diubah ke dalam bentuk satu dimensi melalui Flatten Layer. Setelah fitur melalui flatten, maka kemudian diklasifikasikan melalui sejumlah *neuron,* mencapai *neuron* keluaran dengan jumlah yang setara dengan jumlah kelas dalam rangka klasifikasi. Berikut beberapa lapisan yang ada dimetode CNN:

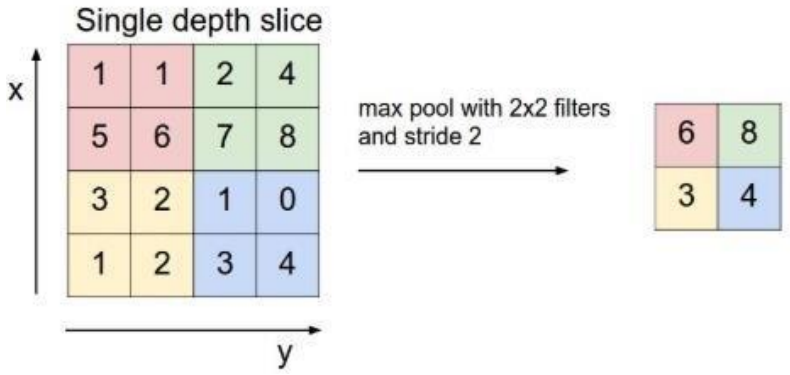
a. *Convolution Layer*, merupakan kunci elemen dalam arsitektur *Convolutional Neural Network.* Komponen ini merupakan kumpulan dari filter yang telah diterapkan dari operasi konvolusi pada input yang disajikan. Hasil dari operasi ini yaitu menghasilkan *feature maps*. Didalam konteks *convolution layer,* matriks akan mewakilkan setiap filter yang mengandung serangkaian nilai yang unik. Nilai inilah yang fungsinya mirip dengan bobot, yang akan mengalami pembaruan disepanjang fase pelatihan ketika selesai dilakukannya konvolusi, dari output operasi ini kemudian diakumulasikan agar menghasilkan *feature maps* yang diinginkan.



Gambar 2.3 Contoh Proses Konvolusi

Gambar 2.3 merupakan proses konvolusi pada CNN yang mana meliputi input, kernel/filter, dan output. Input merupakan gambar matriks 2D yang menunjukkan bahwa original *image*, lalu kernel merupakan matriks kecil yang berguna untuk memproses gambar dari *input*. Dari contoh diatas kernel dengan nilai-nilai yang sudah ditentukan untuk diterapkan pada bagian tertentu dari gambar input dengan cara mengalihkan elemen-elemen yang sesuai lalu menjumlahkannya. Dari hasil tersebut nilai-nilai yang diperoleh ditempatkan pada *output*. Proses konvolusi ini sangat umum digunakan pada pengolahan gambar digital, khususnya dalam arsitektur *convolutional neural network (CNN)* untuk ekstraksi fitur dan deteksi pola.

b. *Pooling Layers*, merupakan komponen yang penting dan berada diantara lapisan konvolusi. Fungsi dari lapisan ini adalah untuk mengurangi volume keluaran dari feature maps, yang mempercepat hasil komputasi di lapisan selanjutnya.



Gambar 2.4 Contoh Proses *Pooling*

Pada gambar 2.4 menunjukkan proses *max pooling* yang merupakan bagian dari algoritma CNN (*Convolutional Neural*

Network), yang kegunaanya untuk mengurangi dimensi data dan mengekstrak fitur penting. Pada contoh gambar diatas input berukuran 4x4 diproses dengan pooling yang berukuran 2x2. Max pooling yang diambil dari nilai maksimalnya yaitu berukuran 2x2 untuk hasilnya.

c. *Fully Connected Layers*, merupakan struktur yang terdapat pada bagian *Classification* yang bermanfaat untuk menghubungkan setiap *neuron* aktivasi dari lapisan sebelumnya ke *neuron* pada lapisan selanjutnya. Pada lapisan ini fitur-fitur yang telah dipelajari oleh jaringan dari gambar akan berguna untuk proses klasifikasi. Lapisan ini terhubung penuh, dianggap sebagai lapisan dari jaringan saraf tiruan tradisional (multilayer perceptrons) yang berguna untuk mengklasifikasikan gambar berdasarkan fitur-fitur dan pola yang telah diidentifikasi oleh lapisan sebelumnya.

2.1.6 Pengertian *TensorFlow*

TensorFlow adalah salah satu contoh arsip *open-source* yang dikembangkan oleh *Google* untuk komputasi numerik dan pembelajaran mesin *(Machine Learning)*, khususnya pembelajaran mendalam *(Deep Learning).*

TensorFlow dimanfaatkan untuk menguji model *deep learning*, melatih model dengan data berukuran besar, serta mempersiapkannya untuk produksi. Selain itu, *TensorFlow* mendukung pelatihan dan inferensi berskala besar, memanfaatkan ratusan server dengan *Graphic Processing Unit (GPU)* guna meningkatkan efisiensi pelatihan [6].

TensorFlow memungkinkan para peneliti dan pengembang untuk membangun dan menerapkan model pembelajaran mesin dengan berbagai tingkat kompleksitas. Fokus utama dari *TensorFlow* adalah pembelajaran mendalam *(deep learning)* tetapi juga mendukung metode pembelajaran mesin lainnya. *TensorFlow* dapat digunakan untuk berbagai keperluan, mulai dari penelitian

pembelajaran mesin tingkat lanjut hingga implementasi model dalam produksi di berbagai perangkat, seperti ponsel dengan *TensorFlow Lite*.

2.1.7 Pengertian *Python*

Python adalah sebuah bahasa pemrograman interpretatif dengan banyak fungsi yang berfokus pada kejelasan kode dan kemudahan pemahaman. *Python* juga berguna untuk membuat aplikasi atau pemrograman *script* [12]. Bahasa pemrograman ini dianggap sebagai bahasa pemrograman yang menggabungkan kejelasan sintaks kode dan kemampuan, juga dirancang untuk *programmer* agar memudahkan membuat program seefisien mungkin, kemudahan pengembangan, serta kompatibilitas dengan sistem yang dibuat. *Python* merupakan salah satu bahasa pemrograman serbaguna yang mampu mengeksekusi berbagai instruksi secara langsung (interpretatif) menggunakan pendekatan berorientasi objek. *Python* juga dikenal sebagai bahasa pemrograman yang paling mudah untuk dipelajari. Bahasa ini dikembangkan oleh seorang *programmer* asal *Belanda* bernama Guido Van Rossum [13].

2.1.8 Pengertian *Kotlin*

Kotlin merupakan bahasa pemrograman statis yang berorientasi objek dan fungsional, dirancang untuk sepenuhnya dapat dioperasikan dengan bahasa pemrograman *Java* yang mana dikembangkan oleh *JetBrains*. *Kotlin* juga sudah dioptimalkan untuk pengembangan aplikasi-aplikasi modern, baik server-side maupun client-side. Didesain dengan sintaksis yang lebih sederhana dan efisien daripada *Java*, bahasa pemrograman ini bertujuan untuk meningkatkan produktivitas para pengembang aplikasi dengan cara mengurangi kode yang berlebihan tetapi tetap kompatibilitas penuh dengan bahasa pemrograman *Java*.

Bahasa pemrogram *Kotlin* memiliki performa yang lebih baik dibandingkan bahasa pemrograman *Java* untuk membangun aplikasi berbasis *Android* [14].

2.1.9 Pengertian *Android*

Android adalah sistem operasi *open-source* yang telah dibangun sedemikian rupa untuk ponsel, terutama yang memiliki layar sentuh, seperti *tablet* dan *smartphone*. *Android* juga memberikan platform terbuka kepada pengembang aplikasi mobile yang inovatif, para pengembang bisa memanfaatkan berbagai fitur *hardware* dan *software*, serta ekosistem di *Google Play Store* sangat luas untuk aplikasi *Android*.

Android adalah sistem operasi mobile berbasis open-source yang dapat diakses secara global melalui internet, dan dikembangkan dengan kernel *Linux*. *Android* juga memungkinkan untuk para pengembang agar dapat dimodifikasi secara leluasa dan didistribusikan oleh pembuatnya yang bersifat *open-source*, dan selain itu, telah sangat mendorong para pengembang aplikasi untuk menggunakan kode sumber *Android* sebagai dasar dari proyek pengembangan aplikasi mereka. [15]

2.1.10 Pengertian *Android SDK*

Android SDK adalah sebuah *tools API* (*Application Programming Interface*) yang diperlukan untuk memulai pengembangan aplikasi pada platform *Android* dengan menggunakan bahasa pemrograman *Java*. [16] *Android SDK* merupakan seperangkat alat pengembangan *software* yang memungkinkan untuk para pengembang menguji, menerapkan, dan membangun aplikasi untuk sistem operasi *Android.*

Kesimpulan dari *Android SDK* adalah merupakan sekumpulan alat dan *API* yang berguna untuk mengembangkan, membangun, menguji, serta mengimplementasi sebuah aplikasi diplatform *Android*.

2.1.11 Pengertian *Aplikasi Mobile*

Aplikasi adalah suatu program yang dijalankan manusia untuk melakukan suatu proses pada sistem komputer. *Mobile* merupakan istilah yang mengacu pada kemudahan berpindah dari satu tempat ke tempat yang lain, seperti telepon genggam atau telepon seluler, yang berarti terminal telepon dapat berpindah dari satu tempat ke tempat lain tanpa kehilangan komunikasi. [17]

Aplikasi mobile merupakan sebuah program perangkat lunak yang telah dikembangkan untuk *smartphone* dan *tablet*, dirancang untuk memberikan kepada pengguna fungsionalitas yang spesifik dari hiburan, komunikasi, sampai produktivitas.

2.1.12 Pengertian *IDE Android Studio*

Android Studio didefinisikan sebagai *IDE* resmi untuk mengembangkan aplikasi *Android*, yang berbasis pada *IntelliJ IDEA* dari *JetBrains*. *Android Studio* juga menyediakan alat-alat lengkap seperti untuk pengembangan, debugging, dan pengujian aplikasi *Android*, serta mendukung para *developer* untuk menggunakan fitur-fitur canggih seperti *editor layout visual*, *emulator android*, dan sistem *build* berbasis *Gradle.*

Android Studio adalah sebuah program yang memungkinkan pengujian dan publikasi fase-fase proses pengembangan dan perancangan aplikasi *mobile* yang berjalan pada sistem operasi Android. Lingkungan *development* menyediakan kode bawaan untuk contoh fitur aplikasi yang biasa digunakan, alat uji, kerangka kerja, dan sistem pembangunan yang fleksibel [18].

2.1.13 Pengertian Figma

Figma adalah sebuah *software* yang mana memudahkan para *developer* untuk memulai mendesain aplikasi dan merupakan alat *prototyping*. Aplikasi *figma* juga membantu para *developer* mendesain UI/UX yang mana berguna untuk mendesain aplikasi,

situs web yang dapat diintegerasikan ke proyek lain. *Figma* dapat digunakan dimana saja melalui *browser* [19].

2.2 Perancangan Sistem

Dalam perancangan sistem ini digunakan beberapa diagram yang tedapat pada UML (Unified Modeling Languange) yaitu seperti Use Case Diagram, Class Diagram, Activity Diagram, dan Sequence Diagram berikut penjelasannya.

2.2.1 UML (*Unified Modeling Languange*)

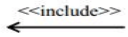
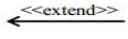
UML (*Unified Modeling Languange*) adalah sebuah bahasa grafis yang digunakan untuk pembuatan model serta komunikasi mengenai sebuah sistem yang digunakan pada diagram-diagram dan tulisan-tulisan pendukung. Yang termasuk kedalam permodelan *UML (Unified Modeling Languange)* yaitu *Use Case Diagram, Class Diagram, Activity Diagram, dan Sequence Diagram* [20]. Karena UML didasarkan pada konsep permodelan *Object Oriented* (OO), yang merupakan sistem mirip dengan kehidupan nyata yang didominasi oleh obyek dan ditunjukkan dengan simbol tertentu, oleh karena itu, *Object Oriented* (OO) memiliki proses yang konsisten dan independen, seperti yang dimiliki oleh sistem kehidupan nyata. Tujuan dari *UML (Unified Modeling Languange)* adalah untuk membantu tim pengembangan proyek berkomunikasi, melihat desain yang mungkin, dan memverifikasi desain arsitektur perangkat lunak atau pembuat program. Salah satu fungsi *UML* yaitu untuk membantu pendeskripsian dan desain sistem pada perangkat lunak, terutama dalam pembuatan sistem yang menggunakan pemrograman berorientasi objek [21].

2.2.2 *Use Case Diagram*

Use Case Diagram merupakan diagram yang menceritakan bagaimana sistem digunakan oleh pengguna.

Use Case Diagram terdiri dari sebuah aktor dan interaksi yang dilakukannya. Aktor dapat berupa manusia, sistem, perangkat keras, atau apa pun yang berinteraksi dengan sistem. [20].

Use Case Diagram juga berfungsi menunjukkan interaksi antara satu atau lebih aktor dengan sistem yang akan dibangun, serta fungsi apa saja yang dilakukan oleh sistem. [22].

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 2.5 Simbol-simbol *Use Case Diagram*

2.2.3 *Activity Diagram*

Activity Diagram merupakan proses langkah-langkah dalam aliran aktivitas yang berada didalam sebuah sistem yang sedang dibangun, bagaimana masing-masing aliran dimulai, keputusan yang mungkin diambil sistem, dan bagaimana sistem itu berakhir. Proses pararel yang mungkin terjadi pada berbagai eksekusi dapat digambarkan dengan *activity diagram*. [20]. *Activity Diagram* merupakan gambaran workflow (aliran kerja) atau aktivitas dari sebuah sistem[22].

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.6 Komponen *Activity Diagram*

2.2.4 *Class Diagram*

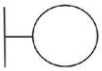
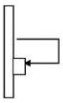
Class Diagram, juga dikenal sebagai diagram kelas, adalah diagram yang digunakan untuk menunjukkan struktur kelas-kelas suatu sistem. *Class diagram* berfungsi memvisualisasikan ikatan pada satu kelas dengan kelas yang lainnya dan memberikan penjelasan pada tiap-tiap kelas didalam model desain (*logical view*). *Class diagram* mempunyai peran untuk menangkap struktur kelas yang akan membentuk arsitektur sistem yang akan dibuat [20]. *Class Diagram* adalah komponen utama sistem berorientasi objek yang menampilkan suatu class, atribut, dan operasinya [22].

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

Gambar 2.7 Komponen *Class Diagram*

2.2.5 *Sequence Diagram*

Sequence Diagram adalah diagram yang menunjukkan interaksi antara objek di dalam dan di sekitar sistem, berisi sebuah pesan yang digambarkan dalam waktu. Diagram *sequance* terdiri dari dua dimensi yaitu, vertical (waktu) dan horizontal (objek yang terkait). [20].

Gambar	Nama	Keterangan
	Entity Class	Gambaran sistem sebagai landasan dalam menyusun basis data
	Boundary Class	Menangani komunikasi antar lingkungan sistem
	Control Class	Bertanggung jawab terhadap kelas-kelas terhadap objek yang berisi logika
	Recursive	Pesan untuk dirinya
	Activation	Mewakili proses durasi aktivasi sebuah operasi
	Life Line	Komponen yang digambarkan garis putus terhubung dengan objek

Gambar 2.8 Komponen *Sequence Diagram*

2.3 Penelitian Terkait

2.3.1 Tabel Penelitian Terkait

Tabel 2.1 Penelitian Terkait

NO	PENELITI	JUDUL PENELITIAN	METODE PENELITIAN	TAHUN PENELITIAN	HASIL PENELITIAN
1	Muhamad Ath-Thariq, Teguh Nurhadi Suharsono	*Deteksi Penyakit Kulit Serupa Pada Wajah Berbasis Mobile dengan Metode Convolutional Neural Network*	*CNN (Convolutional Neural Network)*	*2023*	Dengan menggunakan metode Convolutional Neural Network (CNN) dengan arsitektur LeNet-5 yang memiliki akurasi sebesar 81%, penelitian

					ini bertujuan untuk mengembangkan sistem yang dapat mendeteksi penyakit kulit pada wajah. Dengan demikian, hasil penelitian ini dapat dianggap layak digunakan oleh masyarakat Indonesia.
2	Andre Ilham Saputra, Indra Weni, Ulfa Khaira	*Implementasi Metode Convolutional Neural Network Untuk Deteksi Penyakit Pada Tanaman Kopi Arabika Melalui Citra Daun Berbasis Android*	*CNN (Convolutional Neural Network)*	*2023*	Hasil penelitian ini akan digunakan untuk mengukur akurasi metode Convolutional Neural Network yang digunakan untuk menemukan penyakit pada tanaman kopi arabika. Data yang dikumpulkan langsung dari perkebunan kopi arabika PT Perkebunan Nusantara VI dan dari data.mendeley.co

					m
3	*Mawaddah Harahap, Em Manuel Laia, Lilis Suryani Sitanggang, Melda Sinaga, Daniel Franci Sihombing, Amir Mahmud Husein.*	*Deteksi Penyakit Covid-19 Pada Citra X-Ray Dengan Pendekatan Convolutional Neural Network (CNN)*	*CNN (Convolutional Neural Network)*	*2022*	Studi ini menghasilkan berbagai model arsitektur transfer learning VGG19, MobileNetV2, InceptionResNet V2, dan ResNet (ResNet101V2, ResNet152V2, dan ResNet50V2) yang disarankan untuk deteksi COVID-19 berdasarkan gambar sinar X. Semua eksperimen ini dilakukan dengan mengumpulkan data sinar X dari rahim COVID, kemudian dikelompokkan menjadi data latihan, data uji, dan validasi.
4	*Muhammad Yusuf, Syamsudin Aliphadji Talaohu, Jumria Purnamasari*	*Sistem Pakar Diagnosa Penyakit Pada Tanaman Sawi Menggunakan Metode Convolutional Neural Network*	*CNN (Convolutional Neural Network)*	*2024*	Hasil penelitian menunjukkan bahwa algoritma CNN, metode dalam bidang kecerdasan buatan, telah digunakan secara

		Berbasis Android			efektif dalam sistem pakar untuk mengidentifikasi penyakit pada tanaman sawi secara akurat dan cepat. Bahasa pemrograman Java dan Python digunakan untuk mengimplementasikan aplikasi ini pada sistem pakar berbasis android.
5	*Muhammad Yusuf, Dewi Astria Faroek, Moh Saddam S Pattanang, Anggun M, Ristanti Salam*	*Sistem Pakar Diagnosa Penyakit Pada Tanaman Semangka Merah Menggunakan Metode CNN Berbasis Android*	*CNN (Convolutional Neural Network)*	*2024*	Hasil penelitian menunjukkan bahwa implementasi sistem berbasis android yang menggunakan metode convolutional neural network (CNN) mencapai skor usability testing sebesar 86%. Dengan demikian, penelitian ini dapat dianggap berhasil dalam merancang sistem yang dapat mendeteksi penyakit pada tanaman semangka. Hasil pengujian sistem yang dilakukan dengan metode black box testing

					menunjukkan bahwa sistem telah berjalan sesuai dengan harapan.