**PENERAPAN METODE *CONVOLUTIONAL NEURAL NETWORK* (CNN) DALAM APLIKASI PENDETEKSI PENYAKIT DAUN TANAMAN KENTANG**

**BERBASIS ANDROID**

**SKRIPSI**

Diajukan Sebagai Salah Satu Syarat Memperoleh Gelar

Strata Satu (S1) Pada Program Studi Teknik Informatika

****

**Oleh:**

**SONNYA GHANDI**

**NIM: 201351131**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TINGGI TEKNOLOGI WASTUKANCANA**

**PURWAKARTA**

**2024**

**LEMBAR PENGESAHAN DOSEN PEMBIMBING**

**PENERAPAN METODE *CONVOLUTIONAL NEURAL NETWORK* (CNN) DALAM APLIKASI PENDETEKSI PENYAKIT DAUN TANAMAN KENTANG BERBASIS ANDROID**

**Oleh:**

**Sonnya Ghandi**

**201351131**

**Program Studi Teknik Informatika**

Sekolah Tinggi Teknologi Wastukancana Purwakarta

Menyetujui,

Tim Pembimbing

Tanggal ………………..

|  |  |
| --- | --- |
| Pembimbing Utama  Imam Ma’ruf Nugroho, S.T., M.Kom.  NIDN: 0425019105 | Pembimbing Pendamping  Yudhi Raymond Ramadhan, S.Si., M.Kom.  NIDN: 0422078006 |

Mengetahui,

Ketua Program Studi Teknik Informatika

Teguh Imam Hermanto, S.Kom., M.Kom.

NIDN: 0425049102

**LEMBAR PENGESAHAN DOSEN PENGUJI**

**PENERAPAN METODE *CONVOLUTIONAL NEURAL NETWORK* (CNN) DALAM APLIKASI PENDETEKSI PENYAKIT DAUN TANAMAN KENTANG BERBASIS ANDROID**

**Oleh:**

**Sonnya Ghandi**

**201351131**

**Program Studi Teknik Informatika**

Sekolah Tinggi Teknologi Wastukancana Purwakarta

Menyetujui,

Tim Penguji

Tanggal ………………..

|  |  |
| --- | --- |
| Ketua Penguji  Irsan Jaelani, S.T., M.Kom  NIDN: 0405019001 | Anggota Penguji  Yusuf Muhyidin, S.T., M.Kom  NIDN: 0420059003 |

Mengetahui,

Ketua Program Studi Teknik Informatika

Teguh Imam Hermanto, S.Kom., M.Kom.

NIDN: 0425049102

**PERNYATAAN**

Saya Sonnya Ghandi menyatakan dengan sesungguhnya, bahwa karya tulis yang berjudul “**Penerapan Metode *Convolutional Neural Network* (CNN) Dalam Aplikasi Pendeteksi Penyakit Daun Tanaman Kentang Berbasis Android**” adalah benar hasil karya sendiri. Serta tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan disuatu perguruan tinggi, dan sepengetahuan saya tidak terdapat karya atau pendapat yang pernah ditulis/diterbitkan orang lain, kecuali yang tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

|  |  |
| --- | --- |
|  | Purwakarta,…………………….. |
|  | Oleh,  Sonnya Ghandi  201351131 |

ABSTRAK

**PENERAPAN METODE *CONVOLUTIONAL NEURAL NETWORK* (CNN) DALAM APLIKASI PENDETEKSI PENYAKIT DAUN TANAMAN KENTANG BERBASIS ANDROID**

**Oleh:**

**Sonnya Ghandi**

**201351131**

**Program Studi Teknik Informatika**

Kentang, tumbuhan dikotil dari keluarga *Solanaceae*, menjadi bahan pokok penting dalam industri pangan. Namun, pertanian kentang sering kali menghadapi tantangan penyakit pada daun, yang dapat signifikan mengurangi hasil panen. Penelitian ini bertujuan mengembangkan aplikasi Android menggunakan *Convolutional Neural Network* (CNN) untuk dapat membantu petani atau pengelola pertanian dalam mengidentifikasi penyakit pada daun kentang.

Penelitian ini menggunakan pendekatan CRISP-DM, meliputi tahap *Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, dan Deployment*. CNN digunakan untuk *supervised learning* dengan *dataset* citra penyakit pada daun tanaman kentang.

Model CNN dilatih selama 50 *epoch* dengan hasil yang sangat baik dengan menghasilkan *training accuracy* 99.69%, *validation accuracy* 99.65%, *training loss* 1.76%, dan *validation loss* 2.64%. Ini menunjukkan kemampuan model dalam mengenali penyakit pada daun kentang dengan tingkat akurasi yang tinggi. Dalam evaluasi menggunakan 320 sampel data tes, model mencapai *accuracy* 99.68%. Hasil ini menegaskan keberhasilan aplikasi dalam deteksi dan klasifikasi penyakit daun tanaman kentang menggunakan CNN.

**Kata Kunci:** *Convolutional Neural Network, Deep learning,* Daun kentang.

*ABSTRACT*

***IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK (CNN) METHOD IN ANDROID-BASED APPLICATION FOR DETECTING POTATO LEAF DISEASE***

***By*:**

**Sonnya Ghandi**

**201351131**

***Informatics Engineering***

*Potatoes, dicot plants belonging to the Solanaceae family, are a crucial staple in the food industry. However, potato farming often faces challenges such as leaf diseases, which can significantly reduce yields. This research aims to develop an Android application using Convolutional Neural Network (CNN) to assist farmers or agricultural managers in identifying potato leaf diseases.*

*This study adopts the CRISP-DM approach, encompassing Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment phases. CNN is employed for supervised learning using datasets of leaf disease images from potato plants.*

*The CNN model is trained for 50 epochs, achieving excellent results with a training accuracy of 99.69%, validation accuracy of 99.65%, training loss of 1.76%, and validation loss of 2.64%. These metrics demonstrate the model's ability to accurately identify potato leaf diseases. In evaluation using 320 test samples, the model achieves an accuracy of 99.68%. These findings underscore the success of the application in detecting and classifying potato leaf diseases using CNN.*

***Keywords****: Convolutional Neural Network, Deep learning, Potato leaves.*

PEDOMAN PENGGUNAAN SKRIPSI

Skripsi S1 yang tidak dipublikasikan terdaftar dan tersedia di Perpustakaan Sekolah Tinggi Teknologi Wastukancana, dan terbuka untuk umum dengan ketentuan bahwa hak cipta ada pada pengarang dengan mengikuti aturan HaKI yang berlaku di Sekolah Tinggi Teknologi Wastukancana. Referensi kepustakaan diperkenankan dicatat, tetapi pengutipan atau peringkasan hanya dapat dilakukan seizin pengarang dan harus disertai dengan kebiasaan ilmiah untuk menyebutkan sumbernya.

Memperbanyak atau menerbitkan sebagian atau seluruh Skripsi haruslah seizin Ketua Sekolah Tinggi Teknologi Wastukancana Purwakarta.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah Subhanahu Wa Ta’ala yang telah melimpahkan rahmat serta hidayah-Nya, sehingga penulis dapat merampungkan skripsi yang berjudul: “Penerapan Metode *Convolutional Neural Network* (CNN) Dalam Aplikasi Pendeteksi Penyakit Daun Tanaman Kentang Berbasis Android”.

Dalam penyusunan laporan ini, tentu tidak terlepas dari arahan, bimbingan, bantuan serta doa dari berbagai pihak. Oleh karena itu, ucapan terimakasih sebanyak-banyaknya penulis sampaikan kepada:

1. Bapak Ir. Apang Djafar Shieddique,S.T., M.T., IPM., ASEAN Eng. sebagai Ketua Sekolah Tinggi Teknologi Wastukancana Purwakarta.
2. Bapak Teguh Iman Hermanto, S.Kom., M.Kom. selaku Ketua Program Studi Teknik Informatika .
3. Bapak Imam Ma’ruf Nugroho,S.T., M.Kom. Selaku pembimbing utama.
4. Bapak Yudhi Raymond Ramadhan, S.Si., M.Kom. Selaku pembimbing pendamping.
5. Orang tua, yang telah memberikan dukungan kepada penulis baik secara materi ataupun moril.
6. Pihak-pihak yang telah membantu penulis menyelesaikan laporan ini, yang tidak bisa penulis sebutkan satu per satu.

Semoga semua pihak yang telah membantu mendapat balasan dan pahala yang setimpal dari Allah Subhanahu Wa Ta’ala, Aamiin.

|  |  |
| --- | --- |
|  | Purwakarta,…………………….. |
|  | Oleh,  Sonnya Ghandi  201351131 |

**DAFTAR ISI**

[DAFTAR GAMBAR x](#_Toc174104725)

[DAFTAR TABEL xi](#_Toc174104726)

[DAFTAR LAMPIRAN xii](#_Toc174104727)

[BAB I PENDAHULUAN 1](#_Toc174104728)

[1.1 Latar Belakang Masalah 1](#_Toc174104729)

[1.2 Rumusan Masalah 4](#_Toc174104730)

[1.3 Batasan Masalah 4](#_Toc174104731)

[1.4 Tujuan Penelitian 5](#_Toc174104732)

[1.5 Manfaat Penelitian 5](#_Toc174104733)

[1.5.1 Bagi Pengguna 5](#_Toc174104734)

[1.5.2 Bagi Peneliti 5](#_Toc174104735)

[1.6 Sistematika Penulisan 5](#_Toc174104736)

[BAB II TINJAUAN PUSTAKA 7](#_Toc174104737)

[2.1 Penerapan 7](#_Toc174104738)

[2.2 *Cross Industry Standard Process for Data Mining* (CRISP-DM) 7](#_Toc174104739)

[2.3 Aplikasi 9](#_Toc174104740)

[2.4 Kentang 9](#_Toc174104741)

[2.4.1 Penyakit Daun Tanaman Kentang 10](#_Toc174104742)

[2.5 *Machine Learning* 12](#_Toc174104743)

[2.6 *Deep Learning* 12](#_Toc174104744)

[2.7 *Convolutional Neural Network* 14](#_Toc174104745)

[2.7.1 *Convolution Layer* 15](#_Toc174104746)

[2.7.2 *Pooling Layer* 16](#_Toc174104747)

[2.7.3 *Fully Connected Layer* 16](#_Toc174104748)

[2.8 TensorFlow 17](#_Toc174104749)

[2.9 Tensorflow Lite 17](#_Toc174104750)

[2.10 Keras 18](#_Toc174104751)

[2.11 Android 19](#_Toc174104752)

[2.12 Kotlin 19](#_Toc174104753)

[2.13 Python 19](#_Toc174104754)

[2.14 Penelitian Terdahulu 20](#_Toc174104755)

[BAB III METODOLOGI PENELITIAN 26](#_Toc174104756)

[3.1 Kerangka Penelitian 26](#_Toc174104757)

[3.2 Studi Literatur 27](#_Toc174104758)

[3.3 *Business Understanding* 27](#_Toc174104759)

[3.4 *Data Understanding* 27](#_Toc174104760)

[3.5 *Data Preparation* 29](#_Toc174104761)

[3.6 *Modeling* 30](#_Toc174104762)

[3.7 *Evaluation* 30](#_Toc174104763)

[3.8 *Deployment* 31](#_Toc174104764)

[BAB IV PENGOLAHAN DATA DAN PEMBAHASAN 32](#_Toc174104765)

[4.1 *Data Preparation* 32](#_Toc174104766)

[4.2 *Modeling* 35](#_Toc174104767)

[4.2.1 Proses *Modeling* 35](#_Toc174104768)

[4.2.2 Penentuan parameter *modeling* 38](#_Toc174104769)

[4.3 *Evaluation* 40](#_Toc174104770)

[4.4 *Devlopyment* 42](#_Toc174104771)

[4.4.1 Tamplian Aplikasi 42](#_Toc174104772)

[4.4.2 Tautan Unduh Aplikasi 45](#_Toc174104773)

[BAB V KESIMPULAN DAN SARAN 46](#_Toc174104774)

[5.1 Kesimpulan 46](#_Toc174104775)

[5.2 Saran 46](#_Toc174104776)

[DAFTAR PUSTAKA 47](#_Toc174104777)

[LAMPIRAN 50](#_Toc174104778)

# DAFTAR GAMBAR

[Gambar 1. 1 Grafik konsumsi kentang di Indonesia 2](#_Toc174104779)

[Gambar 1. 2 Grafik Produksi kentang di Indonesia 2](#_Toc174104780)

[Gambar 2. 1 Diagram CRISP-DM 8](#_Toc174104815)

[Gambar 2. 2 Penyakit *Early Blight* 10](#_Toc174104816)

[Gambar 2. 3 Penyakit *Late Blight* 11](#_Toc174104817)

[Gambar 2. 4 Kategori dalam *Deap Learning* 13](#_Toc174104818)

[Gambar 2. 5 *Convlutional Neural Network Layers* 15](#_Toc174104819)

[Gambar 2. 6 Proses Konvolusi 16](#_Toc174104820)

[Gambar 2. 7 Proses *Pooling* 16](#_Toc174104821)

[Gambar 3. 1 Kerangka penelitian 26](#_Toc174104788)

[Gambar 3. 2 *Dataset Class Early Blight* 28](#_Toc174104789)

[Gambar 3. 3 *Dataset Class Late Blight* 29](#_Toc174104790)

[Gambar 3. 4 *Dataset Class Healthy* 29](#_Toc174104791)

[Gambar 3. 5 Proses konversi model 31](#_Toc174104792)

[Gambar 4. 1 Data gambar yang menjadi numerik 33](#_Toc174104793)

[Gambar 4. 2 Contoh citra daun kentang hasil *rotation* dan *flip* 35](#_Toc174104794)

[Gambar 4. 3 Grafik *loss* 30dan40 *epoch* 39](#_Toc174104795)

[Gambar 4. 4 Grafik *loss* 50 *epoch* 40](#_Toc174104796)

[Gambar 4. 5 Grafik *training* model 41](#_Toc174104797)

[Gambar 4. 6 Nilai *evaluation* model 41](#_Toc174104798)

[Gambar 4. 7 *Confusion matrix* 42](#_Toc174104799)

[Gambar 4. 8 Halaman *onboarding* 43](#_Toc174104800)

[Gambar 4. 9 Halaman utama 43](#_Toc174104801)

[Gambar 4. 10 Halaman kamera 44](#_Toc174104802)

[Gambar 4. 11 Halaman klasifikasi penyakit 44](#_Toc174104803)

[Gambar 4. 12 Halaman detail penyakit 45](#_Toc174104804)

[Gambar 4. 13 *QR Code* unduh aplikasi 45](#_Toc174104805)

# DAFTAR TABEL

[Tabel 2. 1 Penelitian terdahulu 20](#_Toc174104806)

[Tabel 3. 1 *Class* *dataset* 28](#_Toc174104807)

[Tabel 4. 1 *Source code* memuat data 32](#_Toc174104808)

[Tabel 4. 2 *Source code split dataset* 33](#_Toc174104809)

[Tabel 4. 3 *Source code resize* dan *rescale* 34](#_Toc174104810)

[Tabel 4. 4 *Source code sequential* model 36](#_Toc174104811)

[Tabel 4. 5 *Source code compile* model 37](#_Toc174104812)

[Tabel 4. 6 *Source code model fit* 38](#_Toc174104813)

[Tabel 4. 7 Perbandingan *epoch* proses *training* model 39](#_Toc174104814)

# DAFTAR LAMPIRAN

[Lampiran A Letter Of Acceptance (LOA) 50](#_Toc174104822)

# PENDAHULUAN

## Latar Belakang Masalah

Penerapan ilmu komputer semakin meluas ke berbagai sektor, termasuk bidang geografis, pertanian, pariwisata, dan kedokteran. Perkembangan teknologi informasi juga memberikan dampak positif terhadap bidang pertanian, khususnya dalam manajemen dan identifikasi penyakit tanaman. Salah satu contohnya adalah dalam klasifikasi penyakit pada tanaman kentang. Tanaman kentang (*Solanum Tuberosum*) merupakan jenis umbi-umbian yang penting dalam pangan global, di samping gandum, jagung, dan beras (Auliatuzahra, 2024).

Kentang, sebagai salah satu tumbuhan dikotil dari keluarga *Solanaceae*, memiliki ciri khas yang mempengaruhi cara reproduksinya. Tanaman kentang umumnya berkembang biak melalui umbinya, yang memungkinkan penyebaran dan pertumbuhan yang efisien di lingkungan yang cocok. Faktor iklim yang lebih dingin cenderung mendukung pertumbuhan optimal tanaman kentang (Fuadi & Suharso, 2022). Selain itu, kentang memiliki peran vital sebagai sumber karbohidrat dalam memenuhi kebutuhan pangan manusia(Auliatuzahra, 2024). Dalam industri pangan, kentang menjadi bahan pokok dalam berbagai produk, menegaskan kepentingannya dalam menjaga ketahanan pangan. Rata-rata konsumsi kentang per kapita di Indonesia menunjukkan tren peningkatan yang signifikan dari tahun 2020 hingga 2022. Pada tahun 2020, total konsumsi kentang per kapita tercatat mencapai 2,547 kilogram. Angka ini mengalami kenaikan yang cukup berarti pada tahun berikutnya, yaitu pada tahun 2021, di mana konsumsi per kapita meningkat menjadi 2,820 kilogram. Peningkatan ini tidak berhenti di situ, karena pada tahun 2022, total konsumsi kentang per kapita terus bertambah hingga mencapai 3,167 kilogram. Persentase pertumbuhan konsumsi kentang dari tahun 2021 hingga 2022 tercatat sebesar 12,28%. Angka ini mencerminkan adanya peningkatan permintaan yang stabil terhadap kentang di kalangan masyarakat Indonesia. Pertumbuhan konsumsi ini dapat disebabkan oleh berbagai faktor, termasuk meningkatnya kesadaran akan manfaat gizi dari kentang, serta diversifikasi produk makanan berbahan dasar kentang yang semakin banyak tersedia di pasaran. (Kementrian Pertanian, 2022). Detailnya bisa dilihat pada grafik berikut.

Gambar 1. 1 Grafik konsumsi kentang di Indonesia

Sumber: (Kementrian Pertanian, 2022)

Namun, meskipun memiliki peran yang signifikan, pertanian kentang sering dihadapkan pada berbagai tantangan dan masalah yang menyebabkan penurunan hasil produksi. Pada tahun 2022 hasil produksi tanaman kentang di Indonesia adalah sebesar 1,5 juta ton tetapi pada tahun 2023 terjadi penurunan produksi menjadi 1,2 juta ton (Badan Pusat Statistik, 2024). Detailnya bisa dilihat pada grafik berikut.

Gambar 1. 2 Grafik Produksi kentang di Indonesia

Sumber: (<https://www.bps.go.id/id/statistics-table/2/NjEjMg==/produksi-tanaman-sayuran.html> Diakses tanggal 26-Juni-2024)

Penurunan produksi kentang 80% disebabkan oleh penyakit yang dapat menyebabkan penurunan hasil secara signifikan karena menghambat pertumbuhan (Fuadi & Suharso, 2022). Diduga bahwa jamur penyebab penyakit ini berasal dari umbi-umbi bibit. Pertumbuhan jamur pada umbi kentang dipengaruhi oleh suhu lingkungan. Jamur yang dimaksud adalah jamur *Patogen* dan jamur *Alternaria Solani* (Aumatullah et al., 2021). Secara umum, terdapat dua jenis penyakit yang disebabkan oleh jamur yang menyerang tanaman kentang, yakni *Early Blight* dan *Late Blight*(Fuadi & Suharso, 2022).

Gejala awal dari penyakit busuk daun (*Late Blight*) ini adalah adanya bercak basah yang muncul di tepi daun, yang juga dapat terjadi di bagian tengahnya. Bercak ini kemudian akan meluas dan menyebabkan perubahan warna daun menjadi coklat atau abu-abu. Sementara itu, gejala penyakit bercak kering (*Early Blight*) ditandai oleh munculnya bercak kering berwarna coklat dalam bentuk lingkaran di bagian bawah daun (Rozaqi et al., 2021). Dengan mengamati gejala yang muncul pada tanaman kentang, petani dapat segera mengatasi masalah penyakit daun yang membusuk. Namun, proses identifikasi gejala penyakit ini seringkali menghadapi kendala karena tidak semua petani memiliki pengetahuan yang memadai tentang cara mengenali gejala penyakit ini secara komprehensif.

Penanganan masalah penyakit pada daun kentang telah mengalami perkembangan yang signifikan, tidak hanya terbatas pada sektor pertanian, melainkan juga melibatkan kontribusi dari bidang teknologi. Salah satu kontribusi tersebut datang dari bidang informatika, yang mengimplementasikan teknik pengolahan citra digital atau yang lebih dikenal sebagai *image processing* dalam mengidentifikasi penyakit yang menginfeksi tanaman kentang. Pemanfaatan *image processing* dalam proses identifikasi ini membantu para pengelola pertanian untuk memberikan penanganan yang lebih efektif dan efisien terhadap tanaman yang terkena penyakit atau menunjukkan gejala tidak normal. Seiring dengan kemajuan teknologi, banyak penelitian yang mengembangkan pengolahan citra digital dalam konteks pertanian, baik untuk identifikasi penyakit maupun untuk menganalisis hasil produksi pertanian. Salah satu tujuan penelitian ini adalah untuk memanfaatkan citra digital dalam mengidentifikasi penyakit pada daun tanaman kentang.

Penelitian ini bertujuan untuk mengembangkan sebuah aplikasi berbasis android yang dapat membantu petani atau pengelola pertanian dalam mengidentifikasi penyakit pada daun kentang dengan memanfaatkan data gambar daun kentang. Identifikasi penyakit pada daun tanaman kentang dibagi menjadi tiga kategori, yaitu daun sehat, *Late Blight*, dan *Early Blight*. Penelitian ini menggunakan algoritma *Convolutional Neural Network* (CNN), sebuah metode dalam bidang *Deep Learning*, untuk melakukan identifikasi tersebut.

Berdasarkan latar belakang yang telah diuraikan di atas, Peneliti mengambil judul "PENERAPAN METODE *CONVOLUTIONAL NEURAL NETWORK* (CNN) DALAM APLIKASI PENDETEKSI PENYAKIT DAUN TANAMAN KENTANG BERBASIS ANDROID” sebagai fokus utama penelitian ini.

## Rumusan Masalah

Berdasarkan latar belakang yang sudah dijelaskan, maka dapat dirumuskan permasalahan yang dapat diambil pada penelitian ini adalah bagaimana mengimplementasikan metode CNN dalam aplikasi pendeteksi penyakit daun tanaman kentang berbasis Android?

## Batasan Masalah

Berdasarkan rumusan masalah yang telah diuraikan, penting untuk menetapkan batasan masalah agar penelitian tetap terfokus pada rencana yang telah disusun, dengan demikian tujuan utama penelitian dapat tercapai sesuai yang direncanakan. Oleh karena itu, penelitian ini dibatasi dengan beberapa aspek, yakni:

1. Aplikasi yang dibangun hanya untuk *platform* Android.
2. Metode yang digunakan dalam aplikasi ini adalah *Convolutional Neural Network.*
3. Menggunakan metode *Cross Industry Standard Process for Data Mining* (CRISP-DM) sebagai kerangka penelitian.
4. Bahasa pemrograman yang digunakan adalahKotlin dan Python*.*
5. Luaran dari aplikasi ini adalah menampilkan hasil deteksi penyakit daun tanaman kentang.
6. Aplikasi yang dibangun hanya untuk mendeteksi jenis penyakit pada daun tanaman kentang, jenis tanaman lain tidak termasuk dalam cakupan deteksi aplikasi ini.
7. Penyakit daun tanaman kentang yang dideteksi hanya dua jenis yaitu *Early Blight* dan *Late Blight.*

## Tujuan Penelitian

Mengacu pada rumusan masalah, maka tujuan yang ingin dicapai oleh peneliti adalah membangun aplikasi pendeteksi penyakit daun pada tanaman kentang berbasis android dengan mengimplementasikan metode *Convolutional Neural Network.*

## Manfaat Penelitian

Adapun beberapa manfaat pada penelitian ini terbagi menjadi dua yaitu bagi pengguna, dan bagi peneliti.

### Bagi Pengguna

1. Mempermudah pendeteksian penyakit daun pada tanaman kentang.
2. Memudahkan petani atau pengelola pertanian untuk segera melakukan penanganan terhadap penyakit *Early Blight* dan *Late Blight.*

### Bagi Peneliti

1. Meningkatkan kompetensi akademik peneliti, yang dapat diterapkan dalam studi lanjutan atau karir di masa depan.
2. Memberikan kesempatan bagi peneliti untuk menerapkan pengetahuan teoritis dalam konteks yang praktis dan relevan

## Sistematika Penulisan

Sistematika penulisan laporan yang disusun terbagi dalam beberapa bab, antara lain:

**BAB I PENDAHULUAN**

Bab ini menguraikan berbagai hal mengenai latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian dan sistematika penulisan. Uraian bab ini dimaksudkan untuk menjelaskan latar belakang penelitian sehingga dapat memberi masukan sesuai dengan tujuan penelitian dengan batasan-batasan masalah yang ada.

**BAB II TINJAUAN PUSTAKA**

Pada bab ini peneliti akan memaparkan dasar teori yang digunakan, serta rujukan dari penelitian terdahulu yang akan digunakan dalam perancangan aplikasi, dan membantu penulis untuk memahami algoritma *convolutional neural network* selama proses penelitian.

**BAB III METODOLOGI PENELITIAN**

Bab ini berisi tentang langkah-langkah terstruktur dan sistematis yang dilakukan dalam penelitian, dan mengungkapkan proses dan tahapan penelitian yang dimulai dari mendapatkan dataset hingga proses klasifikasi data uji yang ada.

**BAB IV HASIL DAN PEMBAHASAN**

Bab ini berisi tentang hasil penelitian, dimulai dengan hasil akhir aplikasi, klasifikasi data uji, dan akurasi dari aplikasi.

**BAB V KESIMPULAN DAN SARAN**

Bab ini berisi tentang kesimpulan dari penelitian yang telah dilakukan dan saran perbaikan yang perlu dilakukan pada objek penelitian.

# TINJAUAN PUSTAKA

## Penerapan

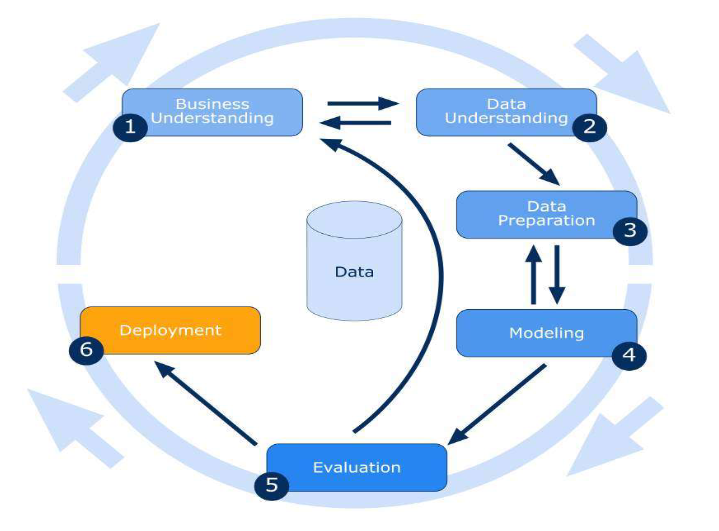
Menurut Kamus Besar Bahasa Indonesia (KBBI), penerapan adalah proses, cara, perbuatan menerapkan.

Penerapan merujuk pada pelaksanaan suatu teori, metode, atau konsep tertentu dengan tujuan mencapai hasil yang diinginkan. Tindakan ini dilakukan atas dasar rencana yang telah disusun sebelumnya untuk mencapai tujuan tertentu, dan dapat melibatkan penggunaan sumber daya yang tersedia secara efektif. Dalam konteks ini, penerapan adalah langkah konkret untuk menerapkan ide atau gagasan dalam praktik, dengan tujuan untuk memenuhi kebutuhan atau kepentingan tertentu(Salam et al., 2021).

## *Cross Industry Standard Process for Data Mining* (CRISP-DM)

CRISP-DM (*Cross Industry Standard Process for Data Mining*) didirikan pada akhir 1990-an oleh empat perusahaan utama: Integral Solutions Ltd. (perusahaan solusi data mining komersial), NCR (perusahaan database), DaimlerChrysler (produsen mobil), dan OHRA (perusahaan asuransi). CRISP-DM diakui sebagai proses standar yang digunakan untuk mengatasi masalah dalam praktik data mining di berbagai industri(Pambudi, 2023).

Menurut F. Krose (dalam Kurniawan & Yasir, 2022) Dalam siklus pengembangannya, CRISP-DM diakui sebagai metodologi *data mining* yang paling komprehensif dalam memenuhi kebutuhan proyek industri. Metodologi ini telah menjadi yang paling umum digunakan dalam proyek analitik, data mining, dan ilmu data. CRISP-DM terdiri dari enam tahapan, dimulai dari *Business Understanding*, kemudian *Data Understanding*, *Data Preparation*, *Modeling*, *Evaluation*, dan *Deployment*.



Gambar 2. 1 Diagram CRISP-DM

(Sumber: Hasanah et al., 2021)

Berikut 6 tahapan CRISP-DM (Hasanah et al., 2021):

1. *Business Understanding*

Pada tahap ini, kegiatan yang dilakukan meliputi memahami kebutuhan dan tujuan dari perspektif bisnis, menguraikan pengetahuan menjadi definisi masalah dalam konteks *data mining*, dan merencanakan serta menetapkan strategi untuk mencapai tujuan *data mining* tersebut.

1. *Data Understanding*

Data understanding merupakan bagian dari tahap persiapan di mana dilakukan pemeriksaan terhadap data yang akan digunakan, pengumpulan data awal, dan identifikasi terhadap kualitas data. Pada tahap ini, data akan dijelaskan secara rinci untuk setiap fitur yang ada.

1. *Data Preparation*

*Data Preparation* adalah langkah yang dilakukan setelah data terkumpul. Pada tahap ini, data akan menjalani proses identifikasi, seleksi, pembersihan, dan transformasi.

1. *Modeling*

*Modeling* adalah langkah implementasi algoritma untuk mencari, mengidentifikasi, dan menghasilkan pola yang akan diterapkan pada data penelitian.

1. *Evaluation*

*Evaluation* adalah proses untuk mengukur hasil evaluasi dari model yang telah diimplementasikan pada tahap sebelumnya, yaitu *modeling*. Hasil evaluasi tersebut mencerminkan proses *data mining* yang telah dilakukan dan menilai model yang paling optimal untuk digunakan.

1. *Deployment*

Menggunakan model yang dihasilkan dapat meliputi pembuatan laporan berdasarkan hasil analisis dari *data mining* atau penerapan proses *data mining* dalam konteks praktis.

## Aplikasi

Aplikasi adalah program yang terdapat di komputer atau ponsel yang berfungsi untuk menjalankan tugas-tugas tertentu sesuai dengan yang telah diprogramkan (Dewi et al., 2021).

Menurut Pramana (dalam Rohayah et al., 2015), aplikasi merupakan jenis perangkat lunak yang dirancang untuk memenuhi berbagai kebutuhan terkait aktivitas tertentu, seperti bisnis, permainan, layanan publik, dan periklanan, serta banyak lagi. Perangkat lunak aplikasi sendiri adalah subkategori dari perangkat lunak komputer yang bertujuan untuk memanfaatkan kapasitas komputer secara langsung dalam menyelesaikan tugas yang dibutuhkan oleh pengguna. Hal ini berbeda dengan perangkat lunak sistem yang berfungsi mengintegrasikan kapabilitas perangkat, namun tidak langsung aplikatif dalam melaksanakan tugas-tugas yang spesifik menguntungkan pengguna. Beberapa contoh perangkat lunak aplikasi meliputi pengolah kata, *spreadsheet*, dan pemutar media.

## Kentang

Kentang (*Solanum tuberosum L.*) adalah jenis tanaman sayuran yang tumbuh setiap musim, dengan siklus hidup antara 90 hingga 180 hari, dan masuk dalam kategori tanaman perdu. Tanaman ini memerlukan tanah yang telah diolah dengan baik dan memiliki tekstur yang lembut. Idealnya, kentang ditanam di area yang berada di dataran tinggi atau wilayah pegunungan, di tempat yang memiliki ketinggian di atas 700 meter dari permukaan laut(Utami et al., 2015).

Kentang adalah contoh dari jenis sayuran yang masuk dalam kelompok sayuran yang membutuhkan iklim sejuk untuk tumbuh dan berkembang, dengan kebutuhan suhu berkisar antara 10 hingga 18 derajat *Celsius*. Tanaman ini memiliki asal usul dari kawasan Pegunungan Andes, khususnya di Peru dan Bolivia, sebelum akhirnya menyebar ke negara-negara seperti Chili, Kolombia, Ekuador, Spanyol, dan meluas ke seluruh wilayah Eropa(Vincent E. Rubatzky & Mas Yamaguchi, 1998). Kentang yang diperkenalkan ke Indonesia, yaitu varietas *Eigenheimer*, berasal dari Amerika. Varietas ini pertama kali ditemukan di wilayah sekitar Cimahi, Bandung, pada tahun 1794. Setelah itu, penyebarannya meluas ke berbagai daerah seperti Karo, Aceh, Padang, Bengkulu, Minahasa, Bali, Seram, dan Timor, menandakan adaptasi dan penerimaan terhadap tanaman ini di berbagai kondisi geografis Indonesia(Utami et al., 2015).

### Penyakit Daun Tanaman Kentang

Berikut ini beberapa penyakit yang dapat menyerang daun tanaman kentang:

1. *Early Blight*

Gejala dari penyakit bercak kering atau disebut juga *Early Blight* dapat ditemukan pada daun, batang, dan buah yang berusia lebih tua. Munculnya bercak dengan warna dari abu-abu ke coklat pada daun yang secara bertahap menyebar ke luar secara konsentris dari pusat yang tetap bersih, menghasilkan pola yang mirip dengan mata banteng. Sekitar luka ini, terdapat lingkaran dengan warna kuning terang yang mencolok. Seiring perkembangan penyakit, daun bisa menguning secara keseluruhan (klorosis) dan akhirnya rontok, yang dapat menyebabkan kehilangan daun secara signifikan (Plantix, 2023a). Berikut contoh penyakit *Early Blight* dapat dilihat pada gambar 2.2.



Gambar 2. 2 Penyakit *Early Blight*

Sumber: (<https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset> Diakses tanggal 25-Maret-2024)

*Early Blight* disebabkan oleh dua jenis jamur yang memiliki kekerabatan dekat, yaitu *Alternaria tomatophila* dan *Alternaria solani*. Kedua jamur ini bertahan hidup di dalam tanah dan pada reruntuhan tanaman (L. Meno dalam Fuadi & Suharso, 2022). Banyak fungisida tersedia di pasaran untuk mengatasi penyakit bercak kering, termasuk produk yang mengandung *azoxystrobin*, *pyraclostrobin*, *difenoconazole*, *boscalid*, *chlorothalonil*, *fenamidone*, *maneb*, *mancozeb*, *trifloxystrobin*, dan *ziram*, baik secara tunggal maupun dalam kombinasi. Sangat dianjurkan untuk berganti-ganti antar senyawa kimia yang digunakan untuk mencegah resistensi. Aplikasi fungisida harus dilakukan secara tepat waktu, dengan memperhatikan prakiraan cuaca. Penting juga untuk memperhatikan interval waktu yang aman antara aplikasi terakhir fungisida dan panen, agar produk dapat dipanen dengan aman (Plantix, 2023a).

1. *Late Blight*

Penyakit busuk daun, dikenal juga sebagai *late blight*, umumnya muncul pada tanaman selama periode pertumbuhan minggu kelima hingga minggu keenam. Tanda-tanda awal dari penyakit ini mencakup kemunculan noda-noda lembap pada pinggiran daun, yang dapat juga terjadi di bagian tengahnya. Kemudian bercak ini akan melebar dan warna daun berubah menjadi coklat/abu-abu.(Rozaqi et al., 2021). Berikut contoh penyakit *Late Blight* dapat dilihat pada gambar 2.3.



Gambar 2. 3 Penyakit *Late Blight*

Sumber: (<https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset> Diakses tanggal 25-Maret-2024)

*Late Blight* disebabkan oleh *Phytophthora infestans* yaitu suatu mikroorganisme yang menyukai lingkungan lembab dan sejuk (A. Majeed dalam Fuadi & Suharso, 2022). Tanda awal penyakit busuk daun yang paling jelas adalah bercak coklat atau bercak penyakit pada batang tanaman. Aplikasi fungisida penting untuk mengendalikan penyakit busuk daun, terutama di daerah lembab. Fungisida kontak yang melapisi daun dapat efektif sebelum infeksi dan tidak memicu kekebalan pada jamur. Fungisida yang mengandung *mandipropamid*, *chlorothalonil*, *fluazinam*, atau *mancozeb* juga dapat digunakan sebagai perlakuan pencegahan. Perlakuan benih sebelum disemai dengan fungisida seperti *mancozeb* juga berhasil(Plantix, 2023b).

## *Machine Learning*

*Machine learning* adalah salah satu cabang ilmu komputer yang bertujuan untuk mengembangkan algoritma yang memungkinkan komputer untuk belajar dari data yang ada. Dengan kata lain, komputer diajarkan untuk memahami pola dan informasi dari data historis sehingga dapat meningkatkan kinerja dalam menganalisis dan mengolah informasi yang diberikan. Hal ini membuat machine learning sering disebut sebagai proses "belajar dari data", di mana komputer memperoleh pengetahuan atau keterampilan baru melalui pengalaman yang diperoleh dari data yang ada. Tujuan utamanya adalah untuk menciptakan model atau program yang dapat mengoptimalkan performa dalam memahami dan menginterpretasikan data yang diberikan (Alpaydin dalam Sidik & Ansawarman, 2022).

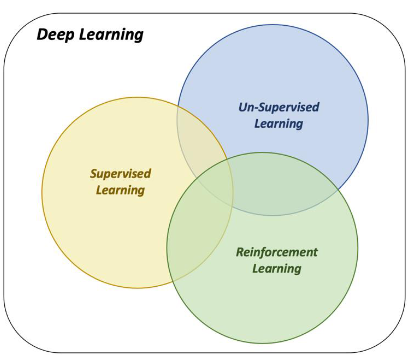
Menurut Tom M. Mitchell dalam (Sidik & Ansawarman, 2022) *Machine Learning* merupakan program komputer yang meningkatkan kemampuannya melalui pengalaman yang didapat dari melaksanakan tugas-tugas tertentu, di mana efektivitasnya dapat diukur secara objektif.

## *Deep Learning*

*Deep Learning* merupakan metode dalam *machine learning* yang memproses informasi kompleks secara nonlinier dengan memanfaatkan berbagai lapisan. Teknik ini bertujuan untuk mengidentifikasi pola, mengekstrak ciri, dan melakukan klasifikasi melalui model komputasi yang dibangun dari berlapis-lapis struktur pemrosesan. Dengan memanfaatkan data pada berbagai tingkat abstraksi, *deep learning* memungkinkan pembelajaran yang mendalam dari informasi yang diberikan (Radikto et al., 2022).

*Deep Learning*, yang mulai dikenal luas sejak tahun 2006, menggunakan mekanisme *deep architecture of learning* atau *hierarchical learning*. Pembelajaran dalam konteks ini merujuk pada prosedur yang mencakup proses estimasi parameter-parameter suatu model, sehingga model (algoritme) yang dikembangkan mampu menyelesaikan tugas atau permasalahan tertentu (Alom et al., 2019).

Menurut J. Schmidhuber dalam (Muhammad Haris Diponegoro et al., 2021) *Deep Learning* menggunakan berbagai lapisan (*layers*) di antara lapisan masukan (*input layer*) dan lapisan keluaran (*output layer*). Arsitektur ini memungkinkan pemrosesan nonlinier melalui beberapa tahap, yang hasilnya bisa digunakan untuk *feature learning* dan *pattern classification*. Berikut pengkategorian *deep learning* dapat dilihat pada gambar 2.4



Gambar 2. 4 Kategori dalam *Deap Learning*

Sumber: (Alom et al., 2019)

Gambar 2.4 menjelaskan teknik dalam *Deep Learning* dapat dikategorikan menjadi *supervised*, *semi-supervised*, dan *unsupervised*. Kategori lainnya, seperti *Reinforcement Learning* (RL) atau *Deep* *Reinforcement Learning* (DRL), sering kali dianggap sebagai bagian dari *semi-supervised* atau *unsupervised*. Beberapa contoh dalam masing-masing kategori:

1. *Deep Supervised Learning*

Teknik *learning* yang digunakan dalam kategori ini menggunakan data yang telah diberi label sebelumnya (*labeled data*). Contoh populer dalam kategori ini adalah *Deep Neural Networks* (DNN), *Convolutional Neural Network* (CNN), *Recurrent Neural Networks* (RNN), termasuk juga LSTM dan *Gated Recurrent Unit* (GRU) (Alom et al., 2019).

1. *Deep Semi-Supervised Learning*

*Semi-supervised learning* menggunakan teknik *learning* yang memanfaatkan sebagian data yang telah diberi label sebelumnya (*partially labeled data*). Dalam beberapa kasus, DRL, *Generative Adversarial Networks* (GAN), serta RNN, termasuk LSTM dan GRU, juga menggunakan teknik learning ini (Alom et al., 2019).

1. *Deep Unsupervised Learning*

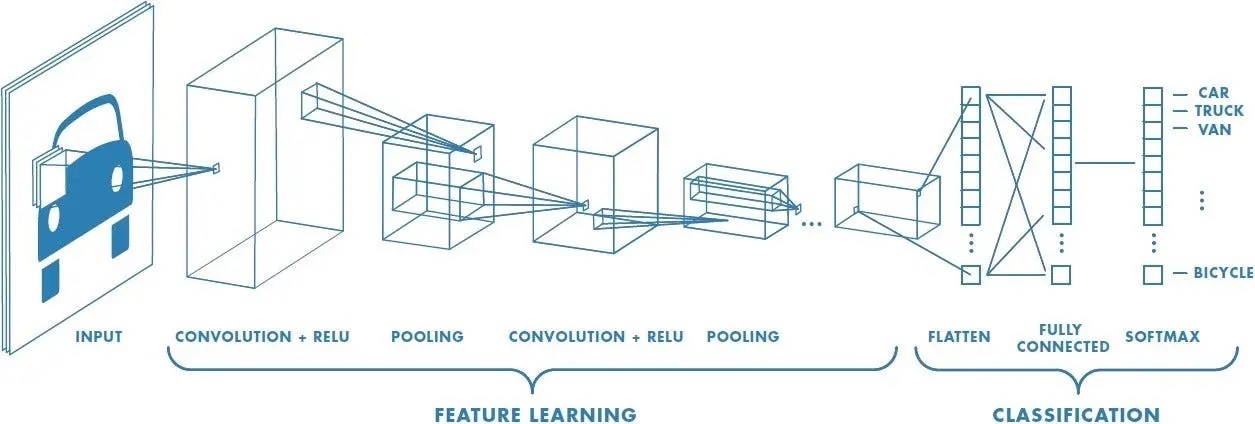
Teknik *learning* ini menggunakan data yang tidak diberi label sebelumnya (*unlabeled data*). *Auto Encoders* (AE), *Restricted Boltzmann Machines* (RBM), dan generasi terbaru dari GAN menggunakan teknik learning ini dalam implementasinya (Alom et al., 2019).

1. *Deep Reinforcement Learning*

Teknik *learning* ini digunakan pada lingkungan atau environments yang tidak diketahui (*unknown environments*). Pada tahun 2013, DRL dimulai dengan hadirnya *Google DeepMind* ( V. Mnih dalam Muhammad Haris Diponegoro et al., 2021).

## *Convolutional Neural Network*

*Convolutional Neural Network* (CNN) adalah serangkaian lapisan pemrosesan yang menjalankan operasi konvolusi, bekerja secara paralel, dan terdiri dari beberapa elemen. Model ini terinspirasi oleh sistem saraf biologis(Romario et al., 2020). Pengolahan citra merupakan salah satu domain dalam deep learning yang dapat memecahkan masalahnya melalui metode *Convolutional Neural Network* (CNN). CNN efektif dalam menangani masalah yang kompleks, menawarkan kinerja yang unggul, dan mampu melakukan klasifikasi pada data citra yang diwakili dalam bentuk matriks. CNN juga termasuk dalam kategori *supervised learning*, di mana modelnya dilatih terlebih dahulu untuk mengidentifikasi pola antara input data dan label output(Harani et al., 2019).



Gambar 2. 5 *Convlutional Neural Network Layers*

Sumber: (Saha, 2018)

Gambar 2.5 menunjukan bahwa inputan akan mengalami proses konvolusi pada segmen pembelajaran fitur (*feature learning*). Fitur, yang merupakan elemen-elemen dalam citra yang diekstraksi oleh filter, kemudian dijadikan peta fitur (*feature map*). Dalam fase pembelajaran fitur, terdapat lapisan tersembunyi (*hidden layer*) yang mencakup lapisan-lapisan konvolusi dan pooling maksimum (*max pooling*). Menyusul segmen pembelajaran fitur, terdapat fase klasifikasi, di mana transformasi fitur menjadi bentuk satu dimensi dijalankan menggunakan teknik pelandaian (*flattening*). Selanjutnya, fitur tersebut diklasifikasikan melalui sejumlah *neuron* dan mencapai *neuron* keluaran, dengan jumlah yang setara dengan jumlah *class* yang ada dalam rangka klasifikasi.

### *Convolution Layer*

*Convolution Layer* atau lapisan konvolusional adalah elemen kunci dalam arsitektur CNN. Komponen ini terdiri dari kumpulan filter yang diterapkan melalui operasi konvolusi pada input yang disajikan. Operasi ini menghasilkan apa yang dikenal sebagai peta fitur. Dalam konteks lapisan konvolusional, setiap filter diwakili oleh matriks yang mengandung serangkaian nilai yang unik. Nilai-nilai ini, yang berfungsi mirip dengan bobot, akan mengalami proses pembaruan sepanjang fase pelatihan sesudah dilakukannya konvolusi. Operasi konvolusi itu sendiri melibatkan perkalian matriks antara filter dan lapisan konvolusional, dengan output dari operasi ini kemudian diakumulasikan untuk menghasilkan peta fitur yang diinginkan (Denata et al., 2021).

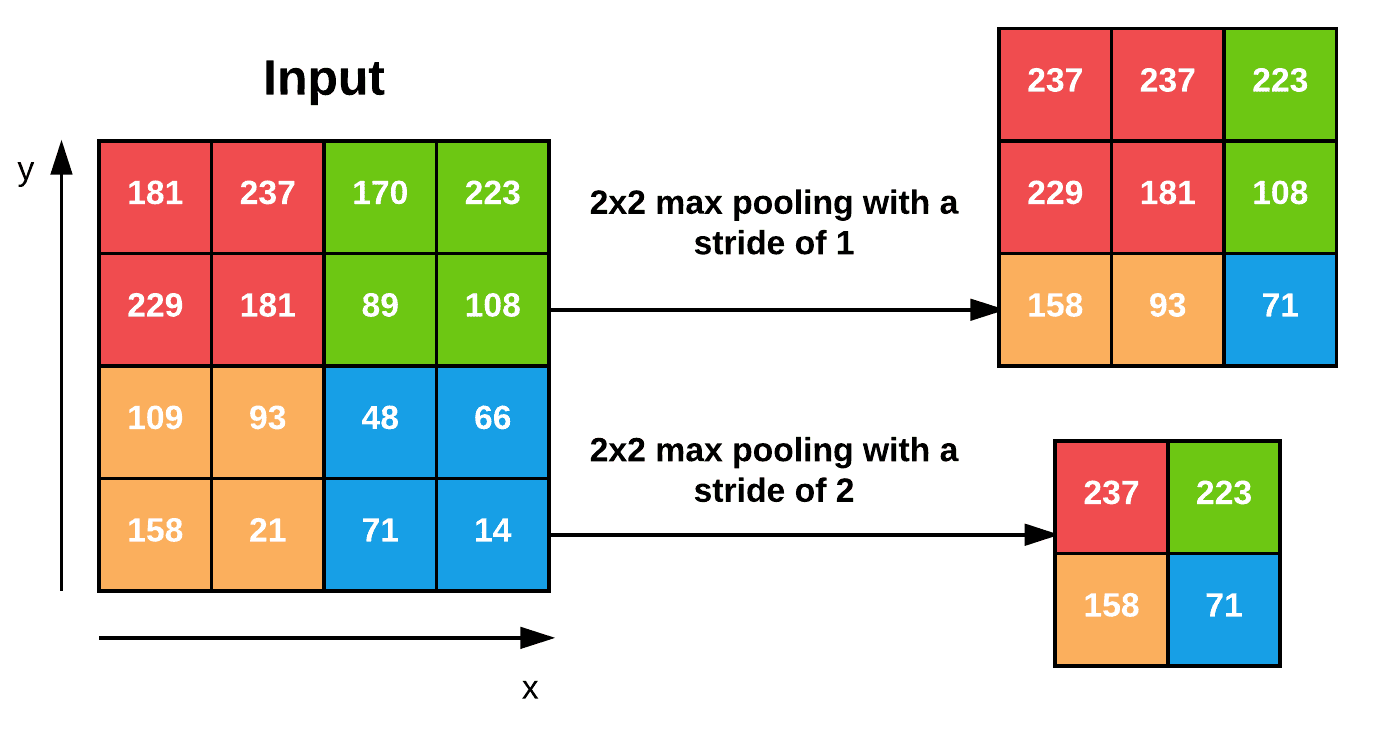


Gambar 2. 6 Proses Konvolusi

Sumber: (Mostafa & Fang-Xiang Wu, 2021)

### *Pooling Layer*

*Pooling Layer* adalah sebuah komponen penting yang terletak antara lapisan konvolusi dalam arsitektur jaringan saraf tiruan. Fungsi utama dari layer ini adalah untuk melakukan pengurangan volume pada keluaran dari peta fitur, yang dapat menghasilkan percepatan komputasi pada lapisan berikutnya (Denata et al., 2021).



Gambar 2. 7 Proses *Pooling*

Sumber: (<https://pyimagesearch.com/2021/05/14/convolutional-neural-networks-cnns-and-layer-types/> Diakses tanggal 23-Maret-2024)

### *Fully Connected Layer*

*Fully connected layer* dalam jaringan saraf tiruan adalah struktur yang menghubungkan setiap *neuron* aktivasi dari lapisan sebelumnya secara langsung dengan setiap *neuron* pada lapisan berikutnya. Perbedaan mendasar antara lapisan terhubung penuh dan lapisan konvolusi adalah bahwa dalam lapisan konvolusi, *neuron-neuron* hanya terhubung dengan wilayah tertentu dari input, sementara dalam lapisan terhubung penuh, setiap *neuron* terhubung dengan semua *neuron* pada lapisan sebelumnya(Denata et al., 2021). Pada titik ini, semua fitur yang telah dipelajari oleh jaringan dari gambar akan digunakan untuk melakukan klasifikasi. Lapisan terhubung penuh dapat dianggap sebagai lapisan dari jaringan saraf tiruan tradisional (*multilayer perceptrons*) yang digunakan untuk mengklasifikasikan gambar berdasarkan fitur-fitur yang telah diidentifikasi oleh lapisan sebelumnya.

## TensorFlow

TensorFlow adalah sebuah *framework* komputasi yang terkenal dalam pembangunan model pembelajaran mesin, menawarkan serangkaian alat yang beragam yang memfasilitasi pembangunan model pada berbagai tingkat abstraksi. Dengan kemampuannya yang luar biasa, TensorFlow mampu menjalankan grafik komputasi pada berbagai platform hardware seperti CPU, GPU, dan TPU (Hikmatia A. E. & Ihsan Zul, 2021). Fleksibilitas ini memungkinkan para peneliti dan praktisi untuk memanfaatkan sumber daya komputasi yang tersedia dengan cara yang paling efisien dan efektif untuk kebutuhan pengembangan model mereka. Sebagai tambahan, *TensorFlow* juga memungkinkan adaptabilitas yang tinggi dalam merancang dan mengimplementasikan model pembelajaran mesin, memastikan bahwa pengguna dapat menyesuaikan model mereka dengan kebutuhan khusus dan infrastruktur yang tersedia dengan optimal.

## Tensorflow Lite

TFLite atau TensorFlow Lite merupakan versi ringan dari TensorFlow yang dibuat khusus untuk penggunaan pada perangkat *mobile* dan sumber daya terbatas lainnya. TensorFlow Lite memungkinkan model *machine learning* beroperasi dengan efisien di perangkat seluler, mikrokontroler, dan perangkat *Internet of Things* (Simbolon & Maulany, 2024). Fitur TensorFlow Lite:

1. Dioptimalkan untuk pembelajaran mesin di perangkat, dengan mengatasi 5 kendala utama: latensi (tidak ada perjalanan pulang pergi ke server), privasi (tidak ada data pribadi yang keluar dari perangkat), konektivitas (konektivitas internet tidak diperlukan), ukuran (model yang diperkecil dan ukuran biner), dan konsumsi daya (kesimpulan yang efisien dan kurangnya koneksi jaringan).
2. Dukungan berbagai platform, mencakup perangkat Android dan iOS, *embedded Linux*, dan *microcontroller.*
3. Dukungan bahasa yang beragam, yang mencakup *Java, Swift, Objective-C, C++,* dan *Python.*
4. Performa tinggi, dengan akselerasi perangkat keras dan optimalisasi model.
5. Contoh, untuk tugas-tugas *machine learning* yang umum seperti klasifikasi gambar, deteksi objek, estimasi pose, menjawab pertanyaan, klasifikasi teks, dan lain-lain di berbagai platform.

## Keras

Keras merupakan perangkat lunak berbasis Python yang dirancang untuk fasilitasi komputasi cerdas, termasuk dalam hal ini adalah pengolahan citra, konversi formula, dan proses klasifikasi. Sebagai pustaka jaringan saraf yang *user-friendly*, Keras beroperasi dengan menggunakan TensorFlow dan Theano sebagai *backend*, memungkinkan penggunaannya secara efisien baik pada unit pemrosesan sentral (CPU) maupun unit pemrosesan grafis (GPU) (Ihsan, 2021). *Framework* ini menawarkan dukungan luas terhadap berbagai arsitektur jaringan saraf, mulai dari jaringan yang sepenuhnya terkoneksi, jaringan konvolusional, lapisan *pooling*, jaringan berulang, hingga lapisan *embedding*. Keras memungkinkan pengguna untuk mengintegrasikan model-model tersebut dalam pembuatan sistem yang lebih kompleks dan multifungsi. Berkat desainnya yang berbasis Python, Keras tidak hanya mudah untuk dipasang dan digunakan tetapi juga mendukung eksplorasi dan pembelajaran yang mendalam terhadap pemrograman jaringan saraf. Lebih jauh, integrasi Keras dengan *backend* seperti TensorFlow dan Theano menjamin skalabilitas dan optimasi kinerja, memastikan bahwa aplikasi dapat dikembangkan dengan fleksibilitas tinggi sesuai dengan kebutuhan spesifik pengguna.

## Android

Android merupakan sistem operasi untuk perangkat bergerak yang dikembangkan di atas kernel Linux, mencakup tidak hanya sistem operasi itu sendiri tetapi juga perangkat lunak antara (*middleware*) dan berbagai aplikasi. Sistem ini menawarkan sebuah platform yang bersifat terbuka, memungkinkan para developer untuk merancang dan mengembangkan aplikasi mereka dengan lebih leluasa. Sebagai sebuah ekosistem yang komprehensif, Android dirancang untuk mendukung inovasi dan kreativitas pengembang aplikasi, dengan menyediakan akses ke fungsi-fungsi perangkat keras perangkat mobile melalui *API* yang kaya dan fleksibel. Melalui pendekatan ini, Android berkontribusi pada evolusi teknologi *mobile*, memperkaya pengalaman pengguna dengan beragam aplikasi yang dapat disesuaikan dan ditingkatkan sesuai dengan kebutuhan dan preferensi individu (Azis et al., 2020).

## Kotlin

Kotlin adalah bahasa pemrograman yang berbasis *Java Virtual Machine* (JVM) dan dikembangkan oleh JetBrains. Bahasa ini dirancang secara pragmatis untuk pengembangan Android, menggabungkan konsep *object oriented programming* (OOP) dengan pemrograman fungsional. Kotlin memiliki interoperabilitas tinggi, sehingga memungkinkan penggunaannya bersamaan dengan bahasa pemrograman Java dalam satu proyek. Selain itu, Kotlin juga dapat digunakan untuk mengembangkan aplikasi desktop, web, dan bahkan backend (Arafat Febriandirza, 2019).

## Python

Python adalah bahasa pemrograman yang menggunakan interpreter untuk mengeksekusi kodenya secara langsung. Python dapat dijalankan di berbagai platform seperti Windows, Linux, dan lainnya. Bahasa ini mengadopsi berbagai paradigma pemrograman dari bahasa lain, termasuk pemrograman prosedural seperti C, pemrograman berorientasi objek seperti Java, dan pemrograman fungsional seperti Lisp. Kombinasi berbagai paradigma ini membuat Python menjadi pilihan yang fleksibel bagi para programmer untuk mengembangkan berbagai jenis proyek (Rahman et al., 2023).

## Penelitian Terdahulu

Berikut ini adalah acuan penelitian terdahulu.

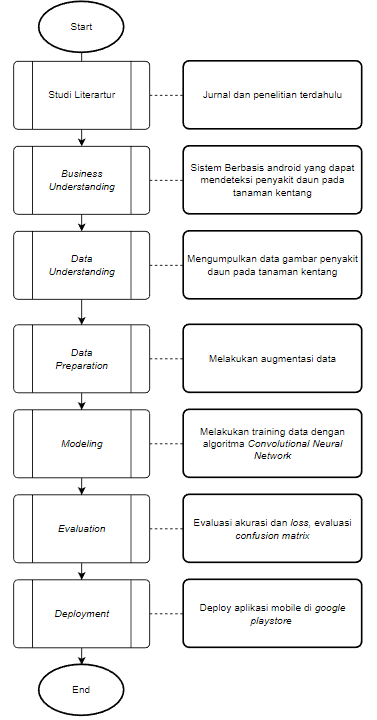
Tabel 2. 1 Penelitian terdahulu

| Judul, Penulis | Metode | Hasil Penelitian |
| --- | --- | --- |
| Perbandingan Arsitektur *Mobilenet* Dan  *NASNetMobile* Untuk Klasifikasi Penyakit Pada Citra Daun Kentang (Fuadi & Suharso, 2022) | Menggunkan metode *Convlutional Neural Network* dengan arsitektur *Mobilenet dan NASNetMobile* | Data yang digunakan pada penelitian tersebut terbagi menjadi gambar daun kentang sehat, gambar daun kentang terinfeksi *Early Blight*, dan gambar daun kentang terinfeksi *Late Blight*. Di akhir penelitian ditemukan bahwa pada skema pengujian menggunakan arsitektur *NASNetMobile* dengan perbandingan data latih dan data uji sebesar 90:10 menghasilkan *accuracy* sebesar 90.96%, *precision* sebesar 90.86%, *recall* sebesar 91.11%, dan f1 score sebesar 92.93%. |
| Deteksi Penyakit pada Daun Kentang Menggunakan Pengolahan Citra  dengan Metode *Convolutional Neural Network* (Rozaqi et al., 2021) | Metode yang digunakan adalah *Convolutional Neural Network* | Dengan pembagian dataset 80% dan 20% dan gambar yang digunakan ukuranya dirubah 150x150. Pada *epoch* ke 10 dengan *batch\_size* 20 dengan total data *training* 922 gambar dan data *testing* 230 gambar menghasilkan nilai akurasi 95% dan untuk akurasi validasi menghasilkan 94% |
| Identifikasi Penyakit Daun Kentang Berdasarkan Fitur Tekstur dan Warna Dengan Menggunakan Metode *K-Nearest Neighbor* (Aumatullah et al., 2021) | Menggunakan metode *K- Nearest Neighbors* dan *Gray Level Co-occurrence Matrix* (GLCM) untuk melakukan ekstraksi fitur tekstur | Penelitian ini mengidentifikasi penyakit pada daun kentang melalui ekstraksi fitur tekstur menggunakan *Gray Level Co-occurrence Matrix* (GLCM) dan fitur warna menggunakan metode K-*Nearest Neighbor* (KNN) pada implementasi sistem. Dataset yang digunakan terdiri dari 270 citra daun kentang, yang dibagi menjadi tiga kelompok: 90 citra untuk data pelatihan, 90 citra untuk data pengujian, dan 90 citra untuk data validasi. Klasifikasi KNN diuji dengan nilai K sebesar 2, 3, 5, 7, 9, dan 10. Hasil pengujian menunjukkan bahwa nilai K terbaik adalah 3, dengan akurasi mencapai 80%. Namun, terdapat 21 data citra uji yang diklasifikasikan secara berbeda dari kelas sebenarnya. |
| Implementasi *Deep Learning* untuk Klasifikasi Jenis Jeruk dengan Menggunakan Metode *Convolutional Neural Network* (Denata et al., 2021) | Metode yang digunakan adalah *Convolutional Neural Network* | Penelitian ini menunjukkan bahwa meskipun jumlah epoch dalam pelatihan model meningkat hingga *epoch* ke-8, akurasi latih tidak selalu mencerminkan akurasi uji yang optimal. Model pada *epoch* ke-9, yang memiliki nilai *loss* terendah, menghasilkan akurasi uji tertinggi (94,81%). Oleh karena itu, peningkatan jumlah epoch tidak menjamin hasil yang lebih baik. Model paling optimal adalah yang memiliki nilai *loss* terendah, seperti yang terjadi pada epoch ke-9, dengan nilai *presisi, recall*, dan F1-measure tertinggi sebesar 94,90%, 94,81%, dan 94,85% serta akurasi pengujian sebesar 94,81%. |
| Sistem Hitung Dan Klasifikasi Objek Dengan Metode Convolutional *Neural Network* (Romario et al., 2020) | Metode yang digunakan adalah *Convolutional Neural Network* | Hasil penelitian menunjukkan bahwa program ini efektif dalam memproses gambar dengan dimensi tertentu, meskipun terdapat kesalahan dalam mendeteksi pola wajah manusia pada gambar yang sebenarnya tidak mengandung wajah manusia. Namun, program berhasil membedakan objek manusia dengan tingkat keberhasilan sekitar 85%. Program juga mampu mengenali detail wajah manusia dengan baik, bahkan pada gambar karikatur. Selain itu, ketika diuji dengan gambar kartun SpongeBob SquarePants, program berhasil membedakan tanpa menampilkan frame objek manusia, mencapai tingkat keberhasilan 95%. |
| Klasifikasi Penyakit Tanaman Padi Mengunakan Metode *Convolutional Neural Network* Melalui Citra Daun (*Multilayer Perceptron*) (Jinan et al., 2022) | Metode yang digunakan adalah *Convolutional Neural Network* | Implementasi algoritma CNN untuk mengklasifikasikan citra daun padi yang terserang penyakit melibatkan pencarian arsitektur terbaik dengan membandingkan beberapa parameter. Arsitektur terbaik menggunakan citra 100x100 piksel, *kernel* 3x3, *learning* *rate* 0,01, *optimizer* Adam, epoch 150, *batch size* 30, dan skenario dataset 90%:10% dengan citra RGB. Hasil pengujian menunjukkan akurasi 91,7% pada data testing. |
| Klasifikasi Penyakit Daun Jagung Menggunakan Metode *Convolutional Neural Network AlexNet* (Azizah, 2023) | Metode yang digunakan adalah *Convolutional Neural Network* denganarsitektur *AlexNet* | Sistem yang memanfaatkan arsitektur *AlexNet* berhasil mengklasifikasi penyakit daun jagung dengan akurasi mencapai 90%. Hal ini menunjukkan bahwa metode yang diusulkan efektif dalam pengenalan jenis penyakit pada tanaman daun jagung. |
| Klasifikasi Penyakit *Early Blight* Dan *Late Blight* Pada  Tanaman Tomat Berdasarkan Citra Daun Menggunakan Metode  Cnn Berbasis Website (Ningsih et al., 2022) | Metode yang digunakan adalah *Convolutional Neural Network* | Metode *Convolutional Neural Network* (CNN) digunakan untuk mengklasifikasi penyakit pada daun tomat. Proses training menggunakan learning rate 0,0001 dan batch size 20. Pada epoch 100, tercapai loss 5%, akurasi 99%, dan recall 85%, hasil terbaik dari percobaan. Training melibatkan 4.000 citra dan 2 jenis penyakit. Akurasi dan recall pada data uji mencapai 80% dan 85% secara berturut-turut, menunjukkan kemampuan CNN dalam mengidentifikasi penyakit pada daun tomat secara efektif. |
| Implementasi Deteksi Objek Penyakit Daun Kentang dengan Metode *Convolutional Neutral Network* (Nauval & Lestari, 2022) | Metode yang digunakan adalah *Convolutional Neural Network* | Dalam penelitian ini, setelah melakukan praproses data pada dataset citra daun kentang, peneliti mengembangkan sebuah model menggunakan *Convolutional Neural Network* (CNN) dan melakukan pengujian yang menghasilkan akurasi sebesar 94%. Kemudian, dilakukan upaya peningkatan akurasi melalui ekstraksi fitur menggunakan metode VGG16, namun ini malah menurunkan akurasi menjadi 93,3%. Dari hasil tersebut, disimpulkan bahwa penggunaan metode CNN dalam mendeteksi citra daun kentang telah memenuhi ekspektasi. Berdasarkan *confusion matrix*, tercatat bahwa 88 citra daun dengan label late blight, 97 citra dengan label *early blight*, dan 97 citra dengan label *healthy* terdeteksi dengan benar. |
| *Convolutional Neural Network* Untuk Perbandingan Optimizer Pada Citra Batang Pohon (Zuzzaifa & Rianto, 2023) | Metode yang digunakan adalah *Convolutional Neural Network* | Dalam penelitian ini, model klasifikasi citra batang pohon Pinus dan Tabebuya menggunakan *Convolutional Neural Network* (CNN) yang meliputi tiga lapisan konvolusi, lapisan *pooling* dengan *max pooling*, *flatten*, dan *dense* layer. Dari hasil pengujian, terlihat bahwa SGD sebagai *optimizer* memberikan kinerja terbaik dengan akurasi pelatihan 0.9453 dan validasi 0.9160 setelah 80 *epochs*. Sementara itu, penggunaan RMSProp pada 20 *epochs* menghasilkan akurasi pelatihan yang sempurna 1.0000 dan validasi 0.9360. Berdasarkan hasil ini, CNN dengan optimizer SGD dan RMSProp terbukti efektif untuk klasifikasi citra batang pohon. Untuk penelitian lebih lanjut, disarankan untuk meningkatkan kualitas pengambilan citra, mencoba berbagai kombinasi dalam desain model CNN, dan menguji model dengan dataset baru. |

# METODOLOGI PENELITIAN

## Kerangka Penelitian

Berikut gambar 3.1 kerangka dari penelitian ini.



Gambar 3. 1 Kerangka penelitian

Pengembangan model pembelajaran dalam pada penelitian ini menerapkan kerangka kerja CRISP-DM dan ditambah satu tahapan lainnya, yang dibagi menjadi 7 tahap. Tahap-tahap tersebut dimulai dengan Studi Literatur, *Business Understanding*, diikuti oleh *Data Understanding*, kemudian berlanjut ke *Data Preparation*, *Modeling, Evaluation*, dan berakhir dengan *Deployment*.

## Studi Literatur

Studi literatur dalam penelitian ini dilakukan dengan beberapa tahapan. Pertama, mengidentifikasi serta mengumpulkan sumber literasi yang relevan, seperti jurnal ilmiah, buku, artikel, laporan penelitian, tesis, dan disertasi. Literatur tersebut diperoleh dari berbagai sumber tepercaya, termasuk database jurnal *online* dan sumber lainnya. Setelah itu, dilakukan evaluasi untuk menyeleksi literatur berdasarkan kredibilitas, relevansi dengan topik, kontribusi teoritis, dan kebaruan informasi. Literasi yang terpilih dianalisis mendalam, di mana peneliti mencatat poin-poin penting dan membandingkan berbagai pandangan serta temuan. Tahapan berikutnya adalah sintesis dan integrasi informasi untuk membangun landasan teori yang solid, yang digunakan untuk merumuskan hipotesis dan pertanyaan penelitian. Terakhir, peneliti menyusun tinjauan pustaka yang merangkum teori, konsep, dan hasil penelitian yang relevan, memberikan konteks serta menunjukkan celah dalam literatur yang ada, yang kemudian menjadi dasar penelitian ini.

## *Business Understanding*

Penelitian ini bertujuan untuk mengembangkan sebuah aplikasi seluler yang mampu mengklasifikasikan penyakit pada daun tanaman kentang menggunakan teknik pembelajaran terarah dengan memanfaatkan algoritma *Convolutional Neural Network*. Aplikasi ini akan mengidentifikasi penyakit pada daun tanaman kentang dengan mengambil gambar melalui kamera ponsel, dan selanjutnya akan menghasilkan output berupa tipe penyakit berdasarkan gambar yang diperoleh.

## *Data Understanding*

*Dataset* yang digunakan dalam penelitian ini terdiri dari gambar penyakit pada daun tanaman kentang, yang diakses dari situs *Kaggle*. Gambar dalam dataset ini menampilkan variasi ukuran piksel dan mencakup tiga *class* atau label berbeda. Berikut ini *class* tersebut:

Tabel 3. 1 *Class* *dataset*

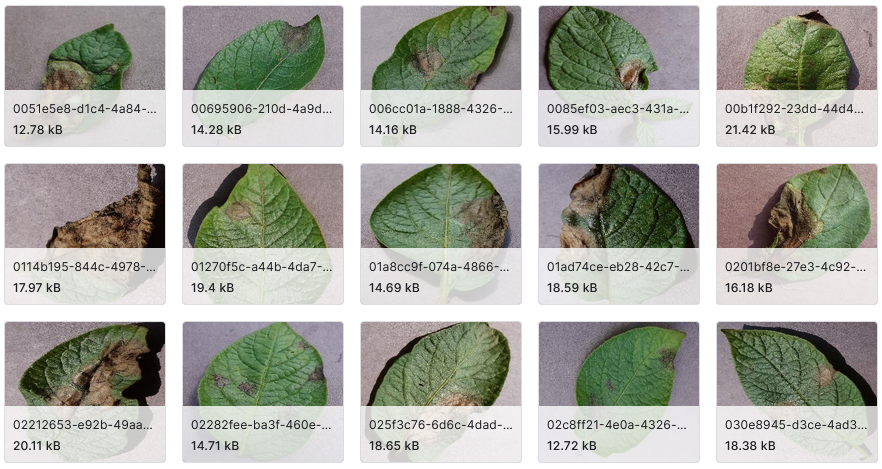
| *Class* | Definisi | Sumber |
| --- | --- | --- |
| *Healthy* | Gambar daun tanaman kentang yang sehat | <https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset> (Ali, 2019) |
| *Late Blight* | Gambar daun tanaman kentang yang terinfeksi penyakit busuk daun |
| *Early Blight* | Gambar daun tanaman kentang yang terinfeksi penyakit bercak kering |

Citra pada dataset tersebut telah diklasifikasikan secara manual oleh para ahli, yang memastikan bahwa setiap gambar memiliki label yang akurat dan sesuai dengan kondisi sebenarnya. Proses klasifikasi manual ini sangat penting untuk memastikan kualitas dan keandalan data yang digunakan dalam pelatihan model, karena label yang akurat memungkinkan model untuk belajar dengan lebih efektif dan menghasilkan prediksi yang lebih tepat. Dengan adanya klasifikasi manual yang dilakukan oleh para ahli, dataset ini menjadi sumber data yang sangat berharga untuk pengembangan dan evaluasi model, khususnya dalam mendeteksi dan mengklasifikasikan penyakit pada daun tanaman kentang. Contoh *dataset* dari setiap *class* dapat dilihat pada gambar 3.2, gambar 3.3 dan gambar 3.4



Gambar 3. 2 *Dataset Class Early Blight*

Sumber: (<https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset> Diakses tanggal 25-Maret-2024)



Gambar 3. 3 *Dataset Class Late Blight*

Sumber: (<https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset> Diakses tanggal 25-Maret-2024)



Gambar 3. 4 *Dataset Class Healthy*

Sumber: (<https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset> Diakses tanggal 25-Maret-2024)

## *Data Preparation*

Sebelum dilakukan training model dataset perlu untuk dilakukan *data augmentation.* Menurut D. Seita (dalam Sanjaya & Ayub, 2020) augmentasi data adalah suatu metode yang membolehkan para praktisi untuk menambah variasi data yang ada untuk pelatihan model tanpa perlu mengumpulkan data baru. Beberapa contoh teknik augmentasi data antara lain adalah pemotongan (*cropping*), penambahan tepi (*padding*), dan pembalikan gambar secara horizontal (*horizontal* *flipping*). Berikut adalah *data augmentation* yang diterapkan pada penelitian ini:

1. *Resize* Dan *Rescale*

Dalam penelitian ini, proses mengubah skala dan ukuran dilakukan pada citra dataset untuk memastikan semua gambar memiliki ukuran yang seragam, yaitu 256 piksel. Hal ini bertujuan untuk memudahkan komputasi selama proses pelatihan model.

1. *Split Dataset*

Dalam penelitian ini, dataset dibagi menjadi dua bagian, yaitu 80% untuk data pelatihan, 10% untuk data validasi dan 10% untuk data uji. Pembagian ini bertujuan untuk memenuhi kebutuhan pelatihan, validasi dan uji model.

1. *Rotation*

Berfungsi untuk melakukan rotasi pada sebuah citra. Penelitian ini dilakukan rotasi sebesar 20 derajat.

1. *Horizontal* dan *Vertical Flip*

*Horizontal* dan *vertical flip* berfungsi untuk membalik sebuah citra menjadi *horizontal* ataupun *vertical*

## *Modeling*

Dalam penelitian ini, algoritma Convolutional Neural Network (CNN) digunakan untuk melatih model menggunakan dataset gambar. Setiap gambar yang menjadi masukan memiliki resolusi 256x256 piksel dan menggunakan tiga kanal warna yaitu Merah (*Red*), Hijau (*Green*), dan Biru (*Blue*). Proses pelatihan dilakukan secara iteratif hingga model mencapai tingkat akurasi yang optimal.

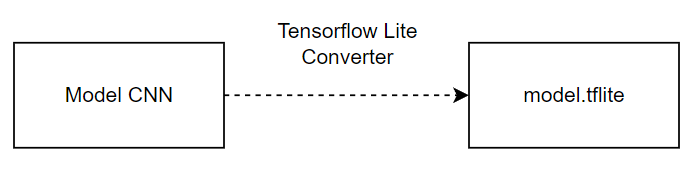
## *Evaluation*

Pada tahap ini, model yang telah dilatih akan dievaluasi dengan melihat nilai dari grafik tingkat akurasi, yang mencakup *training accuracy* dan *validation accuracy*, serta grafik tingkat *error* yang mencakup *training loss* dan *validation loss*. Jika nilai-nilai tersebut rendah atau terjadi *overfitting* atau *underfitting*, akan dilakukan perancangan ulang dan pelatihan model kembali. *Overfitting* adalah kondisi di mana model memiliki kinerja yang sangat baik pada data pelatihan, tetapi kinerjanya menurun ketika diuji dengan data yang berbeda. Ini terjadi karena model terlalu menyesuaikan diri dengan data pelatihan, sehingga kehilangan kemampuan untuk menggeneralisasi pada data baru, sedangkan *underfitting* adalah kondisi di mana model tidak mampu menangkap pola yang ada pada data pelatihan dengan baik, sehingga memiliki kinerja yang buruk baik pada data pelatihan maupun data uji. Ini terjadi karena model terlalu sederhana untuk memodelkan kompleksitas data yang ada. (Rima Dias Ramadhani et al., 2021).

Tahap kedua dari evaluasi melibatkan pembuatan *confusion matrix* untuk model yang telah dibangun. Menurut D. Putra (dalam Normawati & Prayogi, 2021) *confusion matrix* adalah sebuah tabel yang menunjukkan jumlah data uji yang diklasifikasikan secara benar dan jumlah data uji yang diklasifikasikan secara salah.

## *Deployment*

Setelah model melewati tahap pemodelan dan evaluasi, model tersebut siap dikonversi ke format .tflite menggunakan TFLiteConverter yang tersedia dalam *library* TensorFlow.



Gambar 3. 5 Proses konversi model

Langkah selanjutnya adalah melakukan *deployment* model yang telah dikonversi ke dalam aplikasi *mobile* sehingga aplikasi dapat diunduh dan digunakan oleh siapapun yang memiliki *smartphone* dengan sistem operasi Android.

# PENGOLAHAN DATA DAN PEMBAHASAN

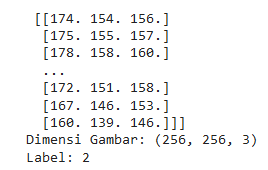
## *Data Preparation*

Dataset yang diperoleh dari situs *Kaggle* akan mengalami *augmentasi* data sebelum memasuki proses pemodelan. Langkah pertama yaitu memuat data gambar dari direktori yang telah disiapkan, proses tersebut dapat dilihat pada tabel 4.1 berikut.

Tabel 4. 1 *Source code* memuat data

|  |
| --- |
| *Source code* |
| BATCH\_SIZE = 32  IMAGE\_SIZE = 256  CHANNELS=3  EPOCHS=50  dataset = tf.keras.preprocessing.image\_dataset\_from\_directory(  "/content/drive/MyDrive/Skripsi/Potato",  seed=123,  shuffle=True,  image\_size=(IMAGE\_SIZE,IMAGE\_SIZE),  batch\_size=BATCH\_SIZE  ) |

Kode pada tabel di atas menggunakan TensorFlow untuk memuat dataset gambar dari direktori, mempersiapkannya untuk pelatihan model. Dataset dibuat menggunakan fungsi `*tf.keras.preprocessing.image\_dataset\_from\_directory*`, yang mengambil gambar dari direktori. Gambar-gambar ini diacak dengan *seed* 123 untuk konsistensi, kemudian diubah ukurannya menjadi 256x256 piksel dan dibagi menjadi *batch* dengan ukuran 32. Setiap *batch* berisi 32 gambar yang akan diproses oleh model secara bersamaan. TensorFlow kemudian mengonversi gambar-gambar ini menjadi tensor, yang merupakan representasi numerik dari gambar tersebut. Setiap piksel dalam gambar dikonversi menjadi nilai numerik, yang biasanya dalam rentang 0 hingga 255 untuk gambar *RGB* dengan 3 *channel.* Contoh data gambar yang telah diubah menjadi numerik bisa dilihat pada gambar 4.1 berikut.



Gambar 4. 1 Data gambar yang menjadi numerik

Setelah data gambar diubah menjadi numerik, data gambar siap untuk digunakan dalam proses *augmentasi* data. Berikut adalah *augmentasi* data yang digunakan dalam penelitian ini:

1. *Split Dataset*

Dalam tahap ini, dataset dibagi menjadi dua bagian, yaitu 80% untuk data pelatihan, 10% untuk data validasi dan 10% untuk data uji.

Tabel 4. 2 *Source code split dataset*

|  |
| --- |
| *Source code* |
| def get\_dataset\_partitions\_tf(  ds, train\_split=0.8,  val\_split=0.1,  test\_split=0.1,  shuffle=True,  shuffle\_size=10000):  assert (train\_split + test\_split + val\_split) == 1  ds\_size = len(ds)  if shuffle:  ds = ds.shuffle(shuffle\_size, seed=12)  train\_size = int(train\_split \* ds\_size)  val\_size = int(val\_split \* ds\_size)  train\_ds = ds.take(train\_size)  val\_ds = ds.skip(train\_size).take(val\_size)  test\_ds = ds.skip(train\_size).skip(val\_size)  return train\_ds, val\_ds, test\_ds |

Berikut penjelasan tabel 4.1, fungsi `get\_dataset\_partitions\_tf` membagi dataset menjadi tiga bagian yaitu pelatihan, validasi, dan pengujian. Pertama, fungsi ini memverifikasi bahwa total pembagian (*train, val, test*) sama dengan 1.0, memastikan persentase yang ditetapkan benar. Jika diatur untuk diacak, dataset akan diacak terlebih dahulu dengan ukuran *buffer* yang ditentukan. Ukuran setiap bagian dataset dihitung berdasarkan fraksi yang diberikan dari total ukuran dataset. Dataset kemudian dibagi menjadi bagian pelatihan (`*train\_ds*`) yang mencakup item pertama sesuai ukuran yang dihitung, bagian validasi (`*val\_ds*`) yang mencakup item berikutnya setelah bagian pelatihan, dan bagian pengujian (`*test\_ds*`) yang mencakup sisa item setelah bagian pelatihan dan validasi. Fungsi ini akhirnya mengembalikan ketiga bagian dataset yang telah dibagi.

1. *Resize* dan *Rescale*

Pada tahap ini gambar akan dilakukan *rescale* sehingga gambar memiliki ukuran 256 piksel.

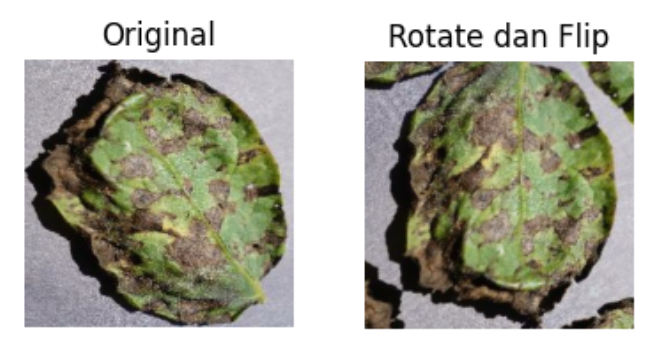
Tabel 4. 3 *Source code resize* dan *rescale*

|  |
| --- |
| *Source code* |
| resize\_and\_rescale = tf.keras.Sequential([  layers.experimental.preprocessing.Resizing(IMAGE\_SIZE, IMAGE\_SIZE),  layers.experimental.preprocessing.Rescaling(1./255),  ]) |

Berikut penjelasan tabel 4.3, kode di atas mendefinisikan sebuah *pipeline preprocessing* menggunakan TensorFlow Keras untuk mempersiapkan gambar sebelum pemodelan. *Pipeline* ini dibuat dengan menggunakan `*tf.keras.Sequential*`, yang memungkinkan penyusunan beberapa layer preprocessing secara berurutan. Pertama, layer `*Resizing*` mengubah ukuran gambar menjadi dimensi yang ditentukan oleh `*IMAGE\_SIZE*`. Selanjutnya, layer `*Rescaling*` melakukan normalisasi piksel gambar dengan membagi setiap nilai piksel dengan 255, sehingga skala piksel berubah dari rentang [0, 255] menjadi [0, 1]. Kombinasi kedua layer ini memastikan bahwa gambar yang diproses memiliki ukuran yang konsisten dan nilai piksel yang terstandarisasi untuk pelatihan model.

1. *Rotation* dan *Flip*

Pada tahap ini dilakukan rotasi sebesar 20% dan membalik citra secara acak antara *horizontal* dan *vertical*



Gambar 4. 2 Contoh citra daun kentang hasil *rotation* dan *flip*

## *Modeling*

*Modeling* terdiri dari dua bagian utama yaitu proses pembuatan model dan penentuan parameter model. Proses pembuatan model melibatkan tahapan-tahapan untuk membangun model, sedangkan penentuan parameter model adalah tahap di mana parameter-parameter model ditentukan dan dibandingkan untuk menilai performa mana yang lebih baik. Berikut ini adalah dua bagian utama dalam *Modeling*:

### Proses *Modeling*

Berikut adalah tahapan yang diambil dalam proses *modeling:*

1. Membuat *sequential* model

*Sequential* model yang bisa dilihat pada Tabel 4.4 akan digunakan sebagai model *training*.

Tabel 4. 4 *Source code sequential* model

|  |
| --- |
| *Source code* |
| input\_shape = (BATCH\_SIZE, IMAGE\_SIZE, IMAGE\_SIZE, CHANNELS)  n\_classes = 3  model = models.Sequential([  resize\_and\_rescale,  layers.Conv2D(32, kernel\_size = (3,3), activation='relu', input\_shape=input\_shape),  layers.MaxPooling2D((2, 2)),  layers.Conv2D(64, kernel\_size = (3,3), activation='relu'),  layers.MaxPooling2D((2, 2)),  layers.Conv2D(64, kernel\_size = (3,3), activation='relu'),  layers.MaxPooling2D((2, 2)),  layers.Conv2D(64, (3, 3), activation='relu'),  layers.MaxPooling2D((2, 2)),  layers.Conv2D(64, (3, 3), activation='relu'),  layers.MaxPooling2D((2, 2)),  layers.Conv2D(64, (3, 3), activation='relu'),  layers.MaxPooling2D((2, 2)),  layers.Flatten(),  layers.Dense(64, activation='relu',kernel\_regularizer = tf.keras.regularizers.l2(0.01)),  layers.Dense(n\_classes, activation='softmax'),  ])  model.build(input\_shape=input\_shape) |

*Model Sequential* yang diberikan terdiri dari beberapa lapisan *convolution* dan *pooling* yang digunakan untuk ekstraksi fitur dari gambar input, diikuti oleh lapisan *fully connected* untuk klasifikasi. Model ini memiliki beberapa lapisan *convolution* (*Conv2D*) dengan 32 dan 64 *filter* yang menggunakan fungsi aktivasi *ReLU* untuk menangkap fitur spasial penting. Setiap lapisan konvolusi diikuti oleh lapisan *pooling* (*MaxPooling2D*) yang mengurangi dimensi spasial untuk mengurangi jumlah parameter dan mencegah overfitting. Setelah serangkaian lapisan konvolusi dan pooling, lapisan Flatten digunakan untuk meratakan output sebelum dimasukkan ke dalam lapisan *dense* (*Dense*) yang terdiri dari 64 *neuron* dengan aktivasi *ReLU*. Terakhir, lapisan *output* menggunakan fungsi aktivasi *Softmax* untuk menghasilkan probabilitas dari masing-masing *class* target dalam tugas klasifikasi.

1. *Compile* *model*

Dalam *Convolutional Neural Network* (CNN), proses kompilasi model (*compile model*) berfungsi untuk mengonfigurasi model agar siap untuk pelatihan. Ini mencakup mendefinisikan loss function, seperti *categorical\_crossentropy* untuk klasifikasi multi-kelas atau *mean\_squared\_error* untuk regresi, yang digunakan untuk mengukur seberapa baik model melakukan prediksi. Selain itu, proses ini menentukan optimizer, seperti SGD, *Adam*, *RMSprop*, atau *Adagrad*, yang memperbarui bobot dalam jaringan berdasarkan gradien dari fungsi kerugian, serta menetapkan *metrics* tambahan, seperti *accuracy*, *precision*, dan *recall*, untuk memantau kinerja model selama pelatihan dan evaluasi. Berikut *source code compile* modelyang dapat dilihat pada tabel 4.5.

Tabel 4. 5 *Source code compile* model

|  |
| --- |
| *Source code* |
| model.compile(  optimizer = 'adam',  loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=False),  metrics=['accuracy']  ) |

Model ini dikompilasi menggunakan *optimizer* *Adam* untuk pembaruan parameter, dengan fungsi *loss* *SparseCategoricalCrossentropy* karena dataset memiliki lebih dari dua *class*, serta *from\_logits=False* karena *output* model telah melalui fungsi aktivasi *softmax*. Metrik evaluasi yang dipilih adalah *accuracy* untuk mengukur performa model selama pelatihan dan pengujian.

1. *Model fit*

*Model* *fit* berfungsi untuk melatih model dengan data yang diberikan, yang melibatkan memasukkan data latih dan label, menentukan jumlah *epochs* (jumlah kali seluruh dataset dilalui selama pelatihan), menetapkan ukuran *batch* (untuk membagi data menjadi batch kecil dan memperbarui bobot model setelah setiap *batch*), dan jika disediakan, menggunakan data validasi untuk mengevaluasi kinerja model pada data yang tidak digunakan selama pelatihan. Metode ini juga dapat menerima berbagai *callback* *functions* yang memungkinkan intervensi selama proses pelatihan, seperti menyimpan model terbaik, menyesuaikan laju pembelajaran, dan menghentikan pelatihan lebih awal

Tabel 4. 6 *Source code model fit*

|  |
| --- |
| *Source code* |
| history = model.fit(  train\_ds,  batch\_size = BATCH\_SIZE,  validation\_data = val\_ds,  verbose = 1,  epochs = EPOCHS,  ) |

Model ini dilatih menggunakan metode *fit* dengan data pelatihan *train\_ds* dan data validasi *val\_ds*, menggunakan *batch\_size* sebesar 32, jumlah *epoch* sebesar 50, dan tingkat verbosity diatur ke 1 untuk memberikan output pelatihan yang lebih rinci.

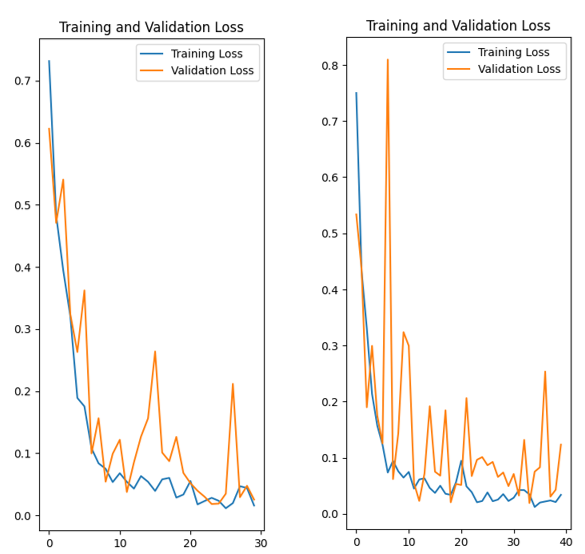
### Penentuan parameter *modeling*

Pada saat melakukan training, model harus menentukan parameter-parameter yang akan mempengaruhi seberapa baik performa model setelah selesai dilatih. Salah satu parameter penting yang perlu ditentukan adalah jumlah *epoch*. Penentuan parameter ini bertujuan untuk membandingkan performa model berdasarkan berbagai nilai parameter dan memilih parameter yang memberikan performa terbaik untuk digunakan dalam model.

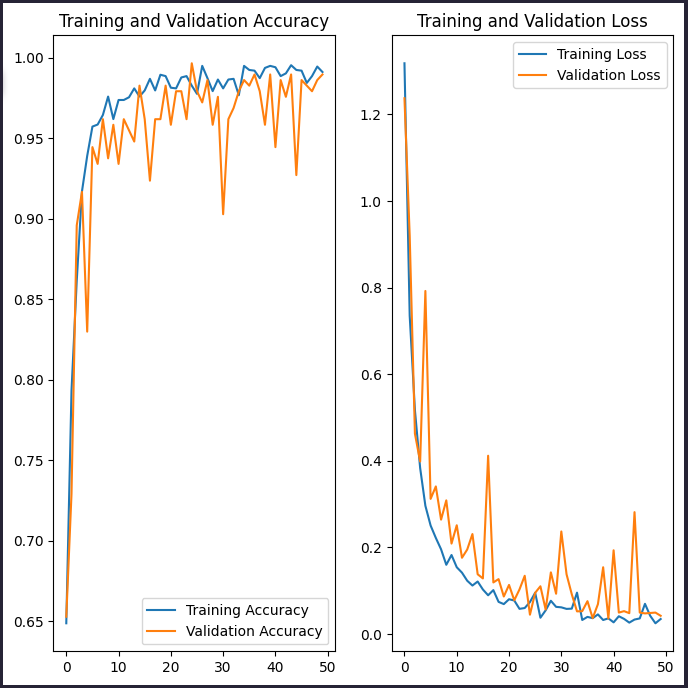
*Epoch* adalah parameter yang mengontrol jumlah perulangan pelatihan setelah seluruh data selesai dilatih(Denata et al., 2021). Berikut ini tabel hasil perbandingan *epoch* dalam proses training model*.*

Tabel 4. 7 Perbandingan *epoch* proses *training* model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Epoch* | *Training Accuracy* | *Training Loss* | *Validation Accuracy* | *Validation Loss* |
| 30 | 0.9945 | 0.0168 | 0.9896 | 0.0219 |
| 40 | 0.9919 | 0.0232 | 0.9896 | 0.0366 |
| 50 | 0.9969 | 0.0176 | 0.9965 | 0.0264 |



Gambar 4. 3 Grafik *loss* 30dan40 *epoch*



Gambar 4. 4 Grafik *loss* 50 *epoch*

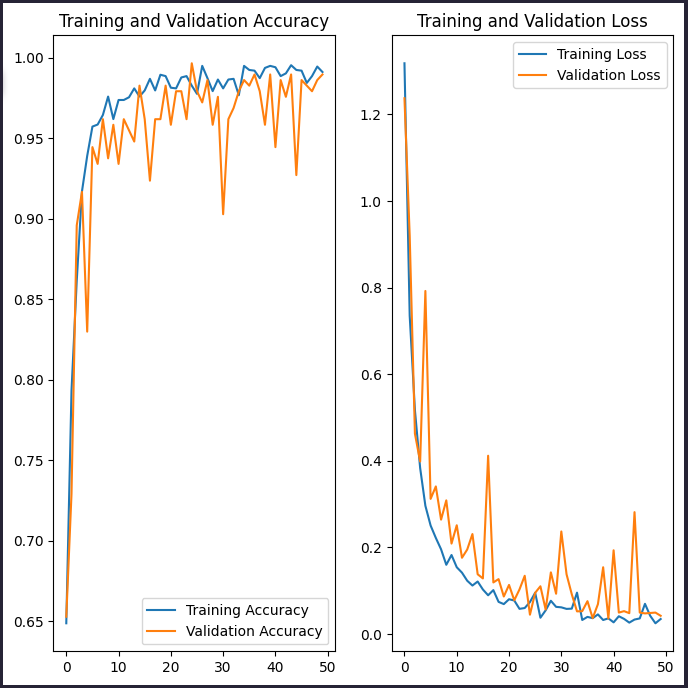
Dari gambar dan tabel di atas menyatakan bahwa modeldengan menggunakan 50 *epoch* memiliki hasil yang lebih baik, dengan nilai *training accuracy* sebesar 99.69%, *validation accuracy* sebesar 99.65%, *training loss* sebesar 1.76% dan *validation loss* sebesar 2.64%.

## *Evaluation*

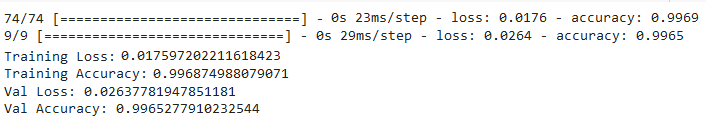
Tahap evaluasi yang dilakukan dalam penelitian ini meliputi analisis grafik dari model yang telah dilatih dan pengamatan hasil dari *confusion matrix*. Setelah proses pelatihan dilakukan, berikut ini adalah hasil grafik pelatihan dan *confusion matrix* dari model yang menggunakan 50 *epoch*.

1. Grafik *training*

Dengan menggunakan grafik ini, dapat diamati apakah model mengalami *overfitting* atau *underfitting*.



Gambar 4. 5 Grafik *training* model

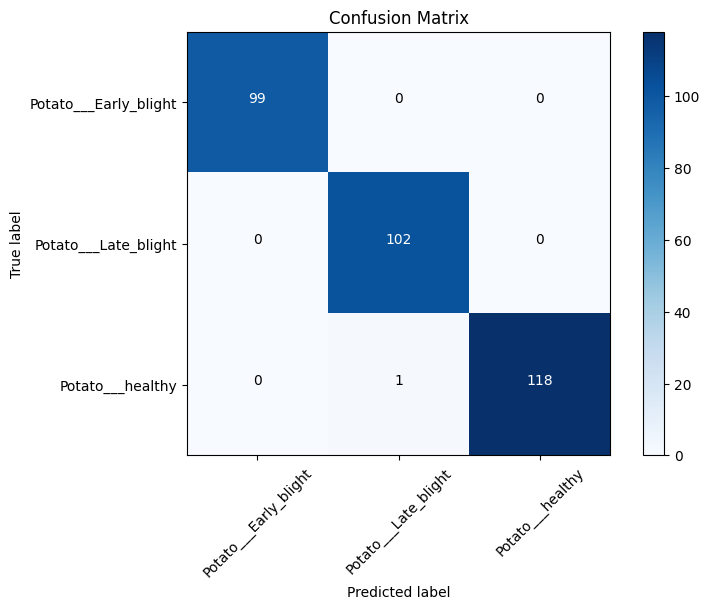


Gambar 4. 6 Nilai *evaluation* model

Dari gambar 4.5 dan 4.6 menyatakan bahwa modelmemiliki akurasi yang bagus yaitu dengan nilai *training accuracy* sebesar 99.69%, *validation accuracy* sebesar 99.65%, *training loss* sebesar 1.76% dan *validation loss* sebesar 2.64%.

1. *Confusion matrix*

*Confusion matrix* memberikan gambaran yang jelas tentang kinerja model dalam mengklasifikasikan data *test*, menunjukkan jumlah prediksi benar (diagonal utama) dan jumlah kesalahan (selain diagonal utama) untuk setiap *class*.



Gambar 4. 7 *Confusion matrix*

Berikut ini perhitungan akurasi dari *confusion matrix* model gambar 4.7:

Maka, hasil akurasi modeldari jumlah data *test* sebanyak 320 data citra adalah 99.68%.

## *Devlopyment*

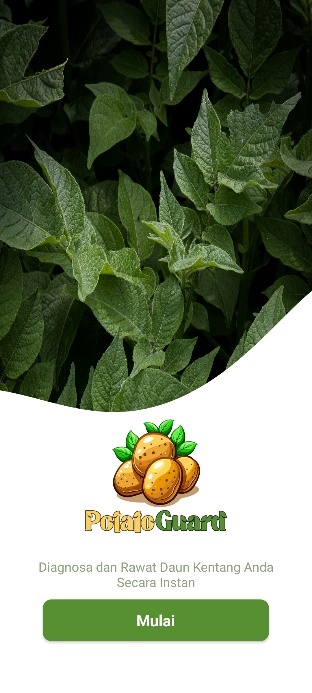
Melakukan penerapan dan *deployment* modelyangtelah dibangun ke dalam aplikasi *mobile,* sehinggaaplikasi dapat diunduh dan dapat digunakan oleh siapapun yang menggunakan *smartphone* dengan sistem operasi Android.

### Tamplian Aplikasi

Berikut ini adalah tampilan dari aplikasi *mobile* yang telah berhasil dikembangkan dan dibangun dengan berbagai fitur yang dirancang untuk memenuhi kebutuhan pengguna secara optimal.

1. Halaman *onboarding*

Halaman *onboarding* adalah halaman pertama yang muncul sebelum ke halaman utama aplikasi. Berikut gambar 4.8 halaman *onboarding.*



Gambar 4. 8 Halaman *onboarding*

1. Halaman utama

Halaman utama menampilkan semua jenis klasifikasi daun kentang, dan terdapat tombol untuk membuka halaman kamera. Berikut gambar 4.9 halaman utama.



Gambar 4. 9 Halaman utama

1. Halaman kamera

Halaman kamera berfungsi untuk mengambil mengambil citra daun kentang yang akan dideteksi dan diklasifikasi. Berikut gambar 4.10 halaman kamera.



Gambar 4. 10 Halaman kamera

1. Halaman klasifikasi penyakit

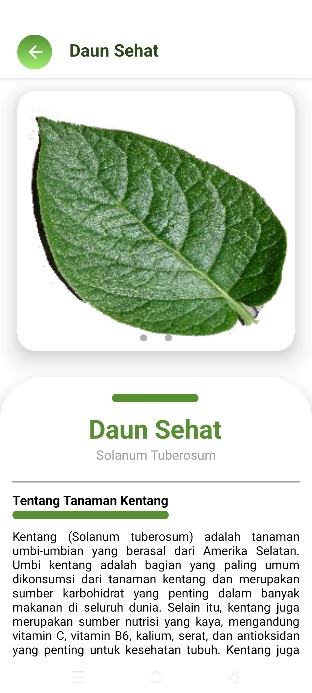
Halaman klasifikasi adalah fitur utama dalam aplikasi ini, fungsinya yaitu menerima inputan citra dari halaman kamera lalu kemudian diklasifikasi jenis penyakitnya. Berikut gambar 4.11 halaman klasifikasi penyakit.



Gambar 4. 11 Halaman klasifikasi penyakit

1. Halaman detail penyakit

Halaman ini menampilkan detail penyakit yang sudah diklasifikasi dari halaman klasifikasi, juga berisi mengenai penyebab, gejala, dan cara pengendalian dari penyakitnya. Berikut gambar 4.12 halaman detail penyakit.



Gambar 4. 12 Halaman detail penyakit

### Tautan Unduh Aplikasi

Berikut ini tautan untuk mengunduh aplikasi **https://bit.ly/potato\_guard** atau melalui *Qr Code* berikut:



Gambar 4. 13 *QR Code* unduh aplikasi

# KESIMPULAN DAN SARAN

## Kesimpulan

Dari hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa metode *Convolutional Neural Network* (CNN) dapat diimplementasikan dengan baik pada aplikasi pendeteksi penyakit daun tanaman kentang berbasis Android. Dengan menggunakan metode ini aplikasi yang dibangun berhasil melakukan deteksi dan klasifikasi penyakit daun tanaman kentang dengan efektif. Pada tahap modeling, model yang telah melalui proses training dengan jumlah *epoch* sebanyak 50 menunjukkan performa yang sangat baik dengan nilai *training accuracy* sebesar 99.69%, *validation accuracy* sebesar 99.65%, *training loss* sebesar 1.76%, dan *validation loss* sebesar 2.64%. Selain itu, pada tahap evaluasi menggunakan data tes sebanyak 320 sampel, model tersebut mencapai akurasi sebesar 99.68%.

## Saran

Berdasarkan hasil penelitian yang telah dilakukan, berikut adalah beberapa saran untuk penelitian selanjutnya:

1. Diharapkan pada penelitian selanjutnya dapat menambah *class* jenis penyakit daun tanaman kentang, ataupun dapat menambahkan *class* untuk melakukan klasifikasi penyakit pada ubi kentang.
2. Mengembangkan sistem berbasis *Internet of Things* (IoT) yang dapat mengintegrasikan aplikasi dengan data dari sensor lingkungan (misalnya kelembaban, suhu) untuk membantu dalam deteksi penyakit, mengingat bahwa kondisi lingkungan juga mempengaruhi kesehatan tanaman.

# DAFTAR PUSTAKA

Ali, A. (2019). *PlantVillage Dataset*. kaggle. https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics (Switzerland)*, *8*(3). https://doi.org/10.3390/electronics8030292

Arafat Febriandirza. (2019). Perancangan Aplikasi Absensi Online Dengan Menggunakan Bahasa Pemrograman Kotlin. *Pseudocode*, *6*(1), 53–59. www.ejournal.unib.ac.id/index.php/pseudocode%0APERANCANGAN

Auliatuzahra, E. (2024). ANALISIS PERAN BIOTEKNOLOGI REKAYASA GENETIKA UNTUK PENINGKATAN KUALITAS PADA TOMAT DAN KENTANG. *HUMANITIS: Jurnal Humaniora, Sosial dan Bisnis*, *2*(1), 113–123.

Aumatullah, L., Ein, I., & Santoni, M. M. (2021). Identifikasi Penyakit Daun Kentang Berdasarkan Fitur Tekstur dan Warna Dengan Menggunakan Metode K-Nearest Neighbor. *Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)*, *April*, 783–791.

Azis, N., Gali Pribadi, & Manda Savitrie Nurcahya. (2020). Analisa dan Perancangan Aplikasi Pembelajaran Bahasa Inggris Dasar Berbasis Android. *IKRA-ITH Informatika*, *4*(3), 251–255.

Azizah, Q. N. (2023). Klasifikasi Penyakit Daun Jagung Menggunakan Metode Convolutional Neural Network AlexNet. *sudo Jurnal Teknik Informatika*, *2*(1), 28–33. https://doi.org/10.56211/sudo.v2i1.227

Badan Pusat Statistik. (2024). *Produksi Tanaman Sayuran, 2021-2023*. bps.go.id. https://www.bps.go.id/id/statistics-table/2/NjEjMg==/produksi-tanaman-sayuran.html

Denata, I., Rismawan, T., Ruslianto, I., Rekayasa Sistem Komputer, J., & MIPA Universitas Tanjungpura, F. (2021). Implementasi Deep Learning untuk Klasifikasi Jenis Jeruk dengan Menggunakan Metode Convolutional Neural Network. *Jurnal Informatika dan Teknologi Informasi*, *18*(3), 297–307. https://doi.org/10.31515/telematika.v18i3.5541

Dewi, N. K., Irawan, B. H., Fitry, E., & Putra, A. S. (2021). Konsep Aplikasi E-Dakwah Untuk Generasi Milenial Jakarta. *Jurnal IKRA-ITH Informatika*, *5*(2), 26–33.

Fuadi, A., & Suharso, A. (2022). Perbandingan Arsitektur Mobilenet Dan Nasnetmobile Untuk Klasifikasi Penyakit Pada Citra Daun Kentang. *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, *7*(3), 701–710. https://doi.org/10.29100/jipi.v7i3.3026

Harani, N. H., Prianto, C., & Hasanah, M. (2019). Deteksi Objek Dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Python. *Jurnal Teknik Informatika*, *11*(3), 47–53. https://ejurnal.ulbi.ac.id/index.php/informatika/article/view/658

Hasanah, M. A., Soim, S., & Handayani, A. S. (2021). Implementasi CRISP-DM Model Menggunakan Metode Decision Tree dengan Algoritma CART untuk Prediksi Curah Hujan Berpotensi Banjir. *Journal of Applied Informatics and Computing*, *5*(2), 103–108. https://doi.org/10.30871/jaic.v5i2.3200

Hikmatia A. E., N., & Ihsan Zul, M. (2021). Aplikasi Penerjemah Bahasa Isyarat Indonesia menjadi Suara berbasis Android menggunakan Tensorflow. *Jurnal Komputer Terapan*, *7*(1), 74–83.

Ihsan, C. N. (2021). Klasifikasi Data Radar Menggunakan Algoritma Convolutional Neural Network (CNN). *DoubleClick: Journal of Computer and Information Technology*, *4*(2), 115. https://doi.org/10.25273/doubleclick.v4i2.8188

Jinan, A., Hayadi, B. H., & Utama, U. P. (2022). Klasifikasi Penyakit Tanaman Padi Mengunakan Metode Convolutional Neural Network Melalui Citra Daun (Multilayer Perceptron). *Journal of Computer and Engineering Science*, *1*(April), 37–44.

Kementrian Pertanian. (2022). Statistik Konsumsi Pangan Tahun 2022. *Pusat Data dan Sistem Informasi Pertanian, Kementrian Pertanian Republik Indonesia*, 1–132.

Kurniawan, D., & Yasir, M. (2022). Optimization Sentimen Analysis using CRISP-DM and Naive Bayes Methods Implemented on Social Media. *Cyberspace: Jurnal Pendidikan Teknologi Informasi*, *6*(2), 74. https://doi.org/10.22373/cj.v6i2.12793

Mostafa, S., & Fang-Xiang Wu. (2021). Neural Engineering Techniques for Autism Spectrum Disorder. *Imaging and Signal Analysis*, *1*, 23–38. https://doi.org/https://doi.org/10.1016/B978-0-12-822822-7.00003-X

Muhammad Haris Diponegoro, Sri Suning Kusumawardani, & Indriana Hidayah. (2021). Tinjauan Pustaka Sistematis: Implementasi Metode Deep Learning pada Prediksi Kinerja Murid. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, *10*(2), 131–138. https://doi.org/10.22146/jnteti.v10i2.1417

Nauval, K. I., & Lestari, S. (2022). Implementasi Deteksi Objek Penyakit Daun Kentang dengan Metode Convolutional Neutral Network. *Jurnal Aplikasi Teknologi Informasi dan Manajemen (JATIM)*, *3*(2), 136–149. https://doi.org/10.31102/jatim.v3i2.1576

Ningsih, N. P., Suryadi, E., Darmawan Bakti, L., & Imran, B. (2022). Klasifikasi Penyakit Early Blight Dan Late Blight Pada Tanaman Tomat Berdasarkan Citra Daun Menggunakan Metode Cnn Berbasis Website Classification of Early Blight and Late Blight Diseases on Tomato Plants Based of Leaf Imagery Using a Website Based Convol. *Jurnal Kecerdasan Buatan dan Teknologi Informasi (JKBTI)*, *1*(3), 27–35.

Normawati, D., & Prayogi, S. A. (2021). Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter. *Jurnal Sains Komputer & Informatika (J-SAKTI)*, *5*(2), 697–711.

Pambudi, A. (2023). Penerapan Crisp-Dm Menggunakan Mlr K-Fold Pada Data Saham Pt. Telkom Indonesia (Persero) Tbk (Tlkm) (Studi Kasus: Bursa Efek Indonesia Tahun 2015-2022). *Jurnal Data Mining dan Sistem Informasi*, *4*(1), 1. https://doi.org/10.33365/jdmsi.v4i1.2462

Plantix. (2023a). *Bercak kering*. Plantix.net. https://plantix.net/id/library/plant-diseases/100321/early-blight/

Plantix. (2023b). *Hawar Daun Kentang*. Plantix.net. https://plantix.net/id/library/plant-diseases/100040/potato-late-blight/

Radikto, Mulyana, D. I., Rofik, M. A., & Zakaria, Mo. Z. Z. (2022). Klasifikasi Kendaraan pada Jalan Raya menggunakan Algoritma Convolutional Neural Network ( CNN ). *Jurnal Pendidikan Tambusai*, *6*(1), 1668–1679.

Rahman, S., Sembiring, A., Siregar, D., Khair, H., Gusti Prahmana, I., Puspadini, R., & Zen, M. (2023). Python : Dasar Dan Pemrograman Berorientasi Objek. In *Penerbit Tahta Media*.

Rima Dias Ramadhani, Nur Aziz Thohari, A., Kartiko, C., Junaidi, A., Ginanjar Laksana, T., & Alim Setya Nugraha, N. (2021). Optimasi Akurasi Metode Convolutional Neural Network untuk Identifikasi Jenis Sampah. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, *5*(2), 312–318. https://doi.org/10.29207/resti.v5i2.2754

Rohayah, S., Sasmito, G. W., & Somantri, O. (2015). Aplikasi Steganografi Untuk Penyisipan Pesan. *Jurnal Informatika*, *9*(1), 975–981. https://doi.org/10.26555/jifo.v9i1.a2038

Romario, M. H., Ihsanto, E., & Kadarina, T. M. (2020). Sistem Hitung dan Klasifikasi Objek dengan Metode Convolutional Neural Network. *Jurnal Teknologi Elektro*, *11*(2), 108. https://doi.org/10.22441/jte.2020.v11i2.007

Rosebrock, A. (2021). *Convolutional Neural Networks (CNNs) and Layer Types*. pyimagesearch. https://pyimagesearch.com/2021/05/14/convolutional-neural-networks-cnns-and-layer-types/

Rozaqi, A. J., Sunyoto, A., & Arief, M. rudyanto. (2021). Deteksi Penyakit Pada Daun Kentang Menggunakan Pengolahan Citra dengan Metode Convolutional Neural Network. *Creative Information Technology Journal*, *8*(1), 22. https://doi.org/10.24076/citec.2021v8i1.263

Saha, S. (2018). *A Guide to Convolutional Neural Networks — the ELI5 way*. Sturn Cloud Blog. https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/

Salam, N. F. S., Manap Rifai, A., & Ali, H. (2021). Faktor Penerapan Disiplin Kerja: Kesadaran Diri, Motivasi, Lingkungan (Suatu Kajian Studi Literatur Manajemen Pendidikan Dan Ilmu Sosial). *Jurnal Manajemen Pendidikan Dan Ilmu Sosial*, *2*(1), 487–508. https://doi.org/10.38035/jmpis.v2i1.503

Sanjaya, J., & Ayub, M. (2020). Augmentasi Data Pengenalan Citra Mobil Menggunakan Pendekatan Random Crop, Rotate, dan Mixup. *Jurnal Teknik Informatika dan Sistem Informasi*, *6*(2), 311–323. https://doi.org/10.28932/jutisi.v6i2.2688

Sidik, A. D., & Ansawarman, A. (2022). Prediksi Jumlah Kendaraan Bermotor Menggunakan Machine Learning. *Formosa Journal of Multidisciplinary Research*, *1*(3), 559–568. https://doi.org/10.55927/fjmr.v1i3.745

Simbolon, S. P., & Maulany, R. (2024). *Development of an Android-Based Waste Type Detection Perancangan Aplikasi Pendeteksi Jenis-jenis Sampah Berbasis Android*. *4*(July), 926–935.

Utami, G. R., Rahayu, M. S., & Setiawan, A. (2015). Penanganan Budidaya Kentang (Solanum tuberosum L.) di Bandung, Jawa Barat. *Buletin Agrohorti*, *3*(1), 105–109. https://doi.org/10.29244/agrob.v3i1.14833

Vincent E. Rubatzky, & Mas Yamaguchi. (1998). *World Vegetables: Principles, Production, and Nutritive Values*. Chapman & Hall.

Zuzzaifa, N., & Rianto, R. (2023). Convolutional Neural Network Untuk Perbandingan Optimizer Pada Citra Batang Pohon. *Jurnal Sistem Cerdas*, *6*(3), 179–188. https://doi.org/10.37396/jsc.v6i3.268

# LAMPIRAN

Lampiran A Letter Of Acceptance (LOA)

