BAB 1

PENDAHULUAN

1.1 Latar Belakang

Tanah gambut adalah tanah yang memiliki potensi cukup besar untuk dimanfaatkan sebagai budidaya pertanian, karena kandungan bahan organik yang tinggi [1]. Tanah ini terbentuk dari akumulasi sisa-sisa tanaman yang mengalami dekomposisi sebagian atau seluruhnya, sehingga kaya akan karbon dan berperan penting dalam ekosistem[2].

Kabupaten Rokan Hulu yang terletak di provinsi Riau, memiliki lahan gambut yang berkontribusi terhadap produktivitas pertanian dan pemanfaatan lahan. Namun, tanah gambut sendiri terdiri dari beberapa jenis dengan ciri dan karakteristik berbeda-beda, yang mempengaruhi pertumbuhan tanaman [3]. Hingga saat ini, belum ada penelitian yang secara khusus melakukan klasifikasi jenis tanah gambut di Rokan Hulu menggunakan metode *Convolutional Neural network (CNN).* Oleh karena itu, identifikasi dan klasifikasi jenis tanah gambut menjadi hal yang penting untuk menentukan strategi pengolahan lahan yang optimal, salah satu cara yang dapat dilakukan adalah dengan menganalisis citra tanah gambut untuk mengklasifikasikannya berdasarkan karakteristik visualnya [3].

Seiring dengan perkembangan kecerdasan buatan *(AI)* terkhususnya metode *Deep Learning* mulai banyak dimanfaatkan dalam berbagai bidang, termasuk klasifikasi citra [4]. Metode yang digunakan untuk klasifikasi adalah metode *Convolutional Neural network (CNN), CNN* terkenal sebagai salah satu algoritma *Deep Learning* yang cukup baik dalam mengenali pola visual gambar [5].

Convolutional Neural network (CNN) adalah salah satu jenis *neural network* yang termasuk ke dalam *Deep Learning* yang biasa digunakan pada pengolahan citra gambar. *Convolutional Neural network (CNN)* dapat dimanfaatkan untuk mendeteksi dan mengidentifikasi objek dalam sebuah gambar [6] yang membuatnya sangat cocok digunakan untuk klasifikasi tanah gambut. Dengan menerapkan *CNN* dalam klasifikasi tanah gambut, diharapkan dihasilkan model yang mampu melakukan klasifikasi dengan tingkat akurasi yang tinggi.

Beberapa penelitian sebelumnya telah membahas klasifikasi jenis tanah menggunakan berbagai algoritma sebelum diterapkan metode *Convolutional Neural network (CNN).* Salah satu contohnya adalah yang memanfaatkan algoritma *Support Vector Machine (SVM)* untuk mengklasifikasikan jenis tanah berdasarkan data gambar yang diambil di India. Dalam penelitian ini, proses pengolahan gambar dilakukan dengan mempertimbangkan warna, energi, HSV, dan faktor lain. Hasil akhir penelitian ini berupa aplikasi dengan fitur seperti nutrisi tanah, rekomendasi pupuk, dan saran tanaman [3].

Penelitian yang dilakukan oleh dan Donny Avianto berfokus pada klasifikasi jenis tanah di wilayah Langensari, Kota Banjar, menggunakan gambar tanah sebagai objek utama. Metode yang diterapkan dalam penelitian ini adalah algoritma *K-Nearest Neighbor (KNN),* dengan pendekatan ekstraksi fitur histogram untuk mendapatkan karakteristik tanah. Fitur-fitur yang diekstraksi meliputi intensitas, standar deviasi, *skewness*, energi, entropi, dan *smoothness*. Hasil penelitian menunjukkan bahwa nilai akurasi tertinggi sebesar 60% dicapai dengan nilai K=3 pada 20 data gambar tanah yang diuji [7].

Penelitian lainnya juga menggunakan gambar tanah sebagai objek klasifikasi untuk membedakan berbagai jenis tanah dengan algoritma Jaringan Syaraf Tiruan Radial Basis *Function (RBF).* Karakteristik tanah diperoleh melalui ekstraksi fitur menggunakan *Matriks Co-Occurrence Level Gray (GLCM),* yang kemudian digunakan untuk proses klasifikasi. Dari total 36 data gambar tanah, 21 data digunakan untuk pelatihan dan 15 data untuk pengujian. Hasilnya menunjukkan bahwa metode *GLCM* mampu menonjolkan fitur tanah dengan sangat baik, menghasilkan tingkat akurasi hingga 90,4% [3].

Implementasi Metode *Convolutional Neural network (CNN)* Untuk Klasifikasi Tanaman pada Citra Resolusi Tinggi oleh Arrofiqoh dan Harintaka. Penelitian ini menggunakan pendekatan klasifikasi semantik otomatis yang mampu mengenali jenis tumbuhan dengan menggunakan algoritma *Convolutional Neural network* (*CNN*). Data citra diambil menggunakan teknologi *UAV* (*Unmanned Aerial Vehicle*). Jenis tanaman yang diteliti adalah padi, bawang merah, kelapa, pisang, dan cabai. Jumlah total dataset yang digunakan sebanyak 500 citra yang dibagi menjadi 3 bagian. Akurasi yang dicapai pada proses *training* sebesar 100% [8].

Sedangkan penelitian lain membahas cara mengkategorikan gambar daun untuk menentukan jenis tanaman obat. Metode *Convolutional Neural network (CNN)* digunakan untuk mengklasifikasikan gambar dalam penelitian ini. Data yang digunakan terdiri dari 34123 data pelatihan dari berbagai sumber dari Bangladesh. Hasil penelitian ini menunjukkan bahwa sistem dapat mengklasifikasikan jenis tanaman obat dengan akurasi 71,3% [9].

Tujuan utama penelitian ini adalah untuk menjawab pertanyaan "Apakah metode *CNN* mampu mengenali jenis tanah gambut tersebut?" melalui pendekatan berbasis *Deep Learning*, diharapkan hasil klasifikasi dapat memberikan kontribusi dalam pengelolaan lahan gambut yang lebih efektif serta mendukung pemanfaatan tanah gambut secara berkelanjutan.

Tantangan dalam klasifikasi tanah gambut tidak hanya diselesaikan dengan membangun model *Convolutional Neural network (CNN)* untuk membangun model klasifikasi, tetapi juga mengintegrasikan ke dalam aplikasi berbasis web menggunakan *Framework Flask*. Aplikasi ini dirancang untuk memberikan kemudahan bagi pengguna dalam memanfaatkan hasil klasifikasi secara langsung melalui antarmuka interaktif. Selain itu, hasil klasifikasi ini juga akan diimplementasikan dalam *User Interface* yang tersedia di *GitHub*, sehingga mempermudah aksesibilitas serta memungkinkan kolaborasi dan pengembangan lebih lanjut oleh pengguna atau peneliti lain.

Berdasarkan masalah yang telah diuraikan, maka judul yang diangkat untuk judul proposal skripsi ini adalah **"Penerapan Metode *Convolutional Neural network (CNN)* Untuk Klasifikasi Jenis Tanah Gambut di Rokan Hulu"**

1.2 Rumusan Masalah

Rumusan masalah dalam penelitian ini adalah bagaimana menerapkan metode *Convolutional Neural network (CNN)* untuk melakukan klasifikasi jenis tanah gambut di Rokan Hulu?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk menerapkan metode *Convolutional Neural network (CNN)* untuk klasifikasi jenis tanah gambut di Rokan Hulu.

1.4 Batasan Masalah

Agar penelitian ini lebih terarah dan fokus, beberapa batasan masalah yang ditetapkan adalah sebagai berikut:

1. Penelitian ini hanya memanfaatkan citra visual tanah yang diambil melalui pemotretan secara langsung sehingga menghasilkan dataset yang lebih sesuai dengan kondisi lapangan.
2. Model klasifikasi yang digunakan dalam penelitian ini terbatas pada metode *Convolutional Neural network (CNN).*
3. Klasifikasi jenis tanah pada penelitian ini dibatasi pada beberapa jenis tanah yang telah teridentifikasi dalam dataset, mencakup tiga kategori tanah gambut yaitu fibrik, hemik, dan saprik.
4. Penelitian ini akan menggunakan *Jupyter Notebook* sebagai *platform* utama untuk melatih, dan menguji model *CNN* .
5. Dataset yang digunakan dalam penelitian ini berjumlah 800 gambar dengan format gambar JPEG/JPG, yang mencakup 4 (empat) kelas yaitu bukan gambut, tanah gambut Saprik, Hemik, dan Fibrik.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut:

a. Memberikan wawasan praktis mengenai penerapan metode *Convolutional Neural network (CNN)* untuk klasifikasi jenis tanah

gambut, sehingga memperlihatkan bagaimana teknologi kecerdasan buatan dapat diterapkan dalam pengelolaan dan analisis data tanah.

b. Memperluas pengetahuan dan pemahaman dalam bidang kecerdasan buatan, khususnya dalam penerapan metode *Convolutional Neural network (CNN)* terutama untuk klasifikasi citra tanah gambut.

BAB 2

LANDASAN TEORI

2.1 Klasifikasi Tanah Gambut

Dalam taksonomi tanah, tanah gambut termasuk dalam *Ordo Histosol*, yang berasal dari bahasa Yunani "*histos*", yang berarti "jaringan." Tanah gambut memiliki lapisan organik yang tebal (lebih dari 40 cm) dan tingkat bahan organik yang tinggi [10]. Tanah gambut dapat diklasifikasikan menjadi tiga jenis berdasarkan tingkat kematangannya: fibrik, hemik, dan saprik [11].

Gambar 2. 1 Tanah Gambut [12]

2.1.1 Tanah Gambut Saprik

Gambut Saprik adalah gambut yang paling matang, dicirikan oleh kandungan serat paling rendah yakni <33%, bobot isi >0,2 g/cm³, kandungan air <450%, dan warna gelap kehitam-hitaman. Jenis tanah gambut ini memiliki tingkat dekomposisi bahan organik yang tinggi, sehingga memiliki tekstur yang lebih padat dan berwarna lebih gelap [13].

Kandungan bahan organik yang tinggi, dan memiliki pH yang cenderung asam. Tanah gambut saprik umumnya ditemukan di daerah dengan curah hujan tinggi dan drainase yang buruk.

Gambar 2. 2 Gambut Saprik [11]

2.1.2 Tanah Gambut Hemik

Gambut Hemik adalah gambut transisi dengan tingkat kematangan sedang, memiliki kandungan serat antara 33–66%, bobot isi 0,1–0,19 g/cm³, kandungan air 450–850%, dan warna coklat kelabu hingga coklat kemerahan. Tanah gambut hemik umumnya ditemukan di daerah dengan curah hujan sedang dan drainase yang sedang [13].

Gambar 2. 3 Gambut Hemik [14]

2.1.3 Tanah Gambut Fibrik

Gambut Fibrik adalah gambut yang paling muda dan belum matang dengan kandungan serat >66%, bobot isi <0,1 g/cm³, kandungan air lebih dari 850%, serta warna kuning kecoklatan. Jenis tanah gambut ini memiliki tingkat dekomposisi bahan organik yang rendah, sehingga memiliki tekstur yang lebih berserat dan berwarna coklat muda. Kandungan bahan organik yang rendah, memiliki pH yang cenderung asam[13].

Gambar 2. 4 Gambut Fibrik [15]

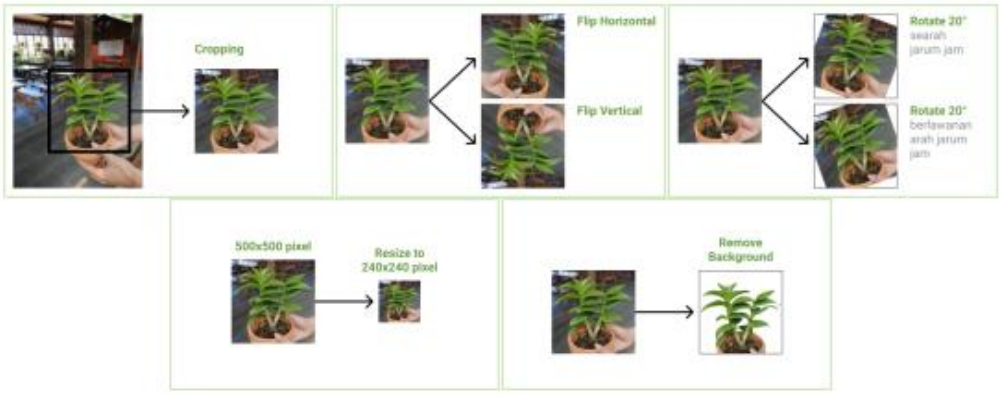
Klasifikasi tanah bertujuan membedakan jenis tanah, termasuk tanah gambut[3]. Dalam pertanian, ini membantu menentukan kesesuaian tanah untuk tanaman, sementara dalam konstruksi, menilai stabilitas tanah. Selain metode fisik, kimia, dan visual, teknologi kecerdasan buatan seperti *Convolutional Neural network (CNN)* kini banyak digunakan untuk klasifikasi tanah melalui pengolahan citra digital [3].

2.2 Pengolahan Citra Digital

Pengolahan Citra Digital (PCD) mempelajari teknik peningkatan kualitas citra, seperti penyesuaian kontras, perubahan warna, dan restorasi. Selain itu, PCD mencakup transformasi dan augmentasi citra untuk mengoptimalkan fitur (*feature*

map) dalam analisis. Tujuannya mencakup identifikasi objek, ekstraksi informasi, dan kompresi data untuk efisiensi penyimpanan. PCD menerima citra asli sebagai *input* dan menghasilkan citra yang telah dimodifikasi.[16].

Dalam Data *Science*, khususnya *Deep Learning* dengan *Convolutional Neural network (CNN),* PCD digunakan untuk augmentasi citra. Teknik ini memperbanyak data dengan menghasilkan variasi citra baru. Operasi yang umum dilakukan meliputi rotasi, *flipping*, *cropping*, *scaling*, dan penyesuaian pencahayaan. [16].



Gambar 2.5 Operasi *Cropping*. *lopping, Flip, Rotate, Resize* dan *Remove Background* [16]

2.2.1 Citra Digital

Citra digital merupakan representasi visual dunia nyata dalam bentuk digital yang dapat diproses oleh komputer. Terdiri dari piksel-piksel berbentuk baris dan kolom, setiap piksel memiliki nilai numerik yang merepresentasikan tingkat kecerahan atau warna. Citra ini dapat diperoleh dari kamera digital, *scanner*, atau simulasi komputer. [17].

2.2.2 Jenis Citra

Dalam pemrosesan citra, terdapat 3 jenis citra yang umumnya digunakan yakni citra berwarna, citra skala keabuan, dan citra biner sebagai berikut:

1. Citra berwarna (RGB)

   Citra berwarna menggunakan tiga saluran utama, yaitu merah (*Red*), hijau (*Green*), dan biru (*Blue*), dikenal sebagai model RGB. Setiap piksel memiliki nilai intensitas untuk ketiga warna tersebut yang dikombinasikan membentuk berbagai spektrum warna. Citra jenis ini umum digunakan dalam fotografi digital, pengenalan objek, dan *augmented reality*.

2. Citra Skala Keabuan (*Grayscale*)

   Citra skala keabuan memiliki satu saluran warna dengan intensitas bervariasi dari hitam (0) hingga putih (255). Karena hanya memiliki satu kanal, citra ini lebih sederhana dan efisien dalam pemrosesan. Citra *grayscale* sering digunakan untuk deteksi tepi, segmentasi objek, dan analisis tekstur.

3. Citra Biner

   Citra biner hanya memiliki dua nilai piksel, yaitu hitam (0) dan putih (1), yang dihasilkan melalui proses thresholding pada citra skala keabuan. Setiap piksel diklasifikasikan berdasarkan ambang batas tertentu. Citra ini banyak digunakan dalam pemrosesan dokumen (OCR), deteksi bentuk, dan analisis pola.

2.3 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan (JST) atau *Artificial Neural network (ANN)* adalah model komputasi yang meniru cara kerja otak manusia. Model ini digunakan untuk

menyelesaikan masalah kompleks seperti pengenalan pola, klasifikasi, dan prediksi melalui pembelajaran mesin (*machine learning*)[18].

2.3.1 Konsep Jaringan Syaraf Tiruan

Setiap informasi yang masuk dan keluar dalam Jaringan Syaraf Tiruan (JST) diproses oleh neuron yang tersusun dalam tiga lapisan utama [19]:

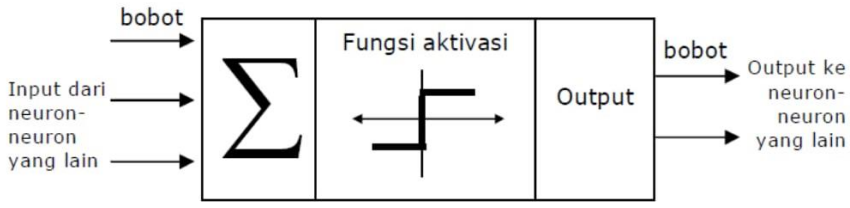
1. Lapisan *Input* : Menerima data dari luar sebagai representasi masalah
2. Lapisan Tersembunyi : Memproses data secara internal tanpa *output* langsung yang dapat diamati.
3. Lapisan *Output* : Menghasilkan solusi berdasarkan pemrosesan jaringan.

2.3.2 Komponen Jaringan Syaraf Tiruan

Jaringan syaraf tiruan terdiri dari beberapa komponen berikut [19]:

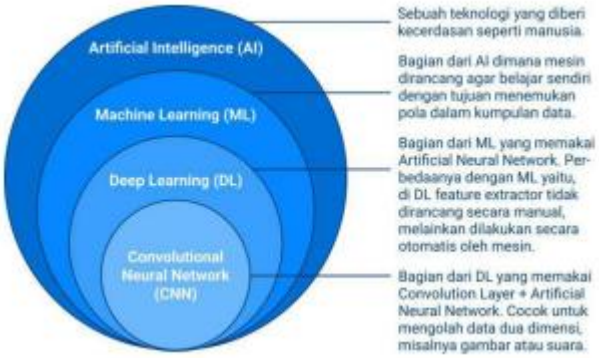
1. Neuron yaitu sel syaraf yang mentransmisikan informasi ke neuron lain.
2. Bobot yaitu hubungan antar neuron.
3. Neuron *layers* yaitu tempat dimana neuron-neuron akan dikumpulkan dalam lapisan-lapisan (*layers*).

Semua jenis jaringan saraf memiliki komponen dasar serupa yang saling terhubung untuk memproses informasi, seperti cara kerja otak manusia.



Gambar 2.6 Komponen Jaringan Syaraf Tiruan [19]

Sebelum memahami apa itu *Convolutional Neural network* (*CNN*), sebaiknya kita mulai dengan mengenal konsep *Artificial Intelligence (AI), Machine Learning (ML), Deep Learning (DL), dan Artificial Neural network (ANN).* Keempat istilah ini memiliki hubungan langsung dengan *CNN* dan penting untuk dikuasai. Berikut penjelasan singkatnya [16]:



Gambar 2.7 Hubungan *AI, ML , DL* dan *CNN* [16]

2.4 *Artificial Intelligence (AI)*

Kecerdasan Buatan *(AI)* adalah cabang Ilmu Komputer yang mengembangkan sistem cerdas untuk menyelesaikan masalah secara kreatif. *AI* meningkatkan efisiensi melalui otomatisasi tugas berulang. Contohnya meliputi mobil tanpa pengemudi, *Google Translate, Google Maps*, dan asisten virtual seperti *Alexa*. [16].

2.5 *Machine Learning*

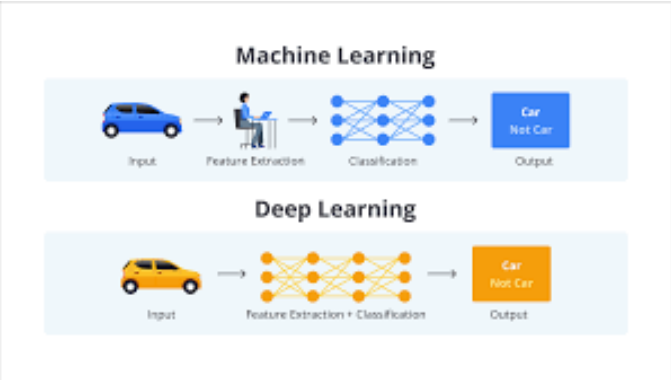
Machine Learning (ML) adalah cabang AI yang memungkinkan mesin belajar dari data tanpa pemrograman eksplisit. Dengan algoritma tertentu, *ML* menganalisis dan menemukan pola secara mandiri. Terdapat tiga jenis utama *ML* yaitu *Supervised Learning, Unsupervised Learning*, dan *Reinforcement Learning*.

Contoh aplikasinya meliputi sistem rekomendasi Amazon, deteksi penipuan transaksi, dan saran teman di media sosial. [16].

2.6 *Deep Learning*

Deep Learning (DL) adalah cabang dari *Machine Learning* yang menggunakan algoritma untuk memodelkan abstraksi data melalui fungsi transformasi non-linear berlapis. *DL* efektif diterapkan pada *supervised, unsupervised, dan semi-supervised learning*, serta *reinforcement learning*. Aplikasi *DL* meliputi pengenalan citra, suara, klasifikasi teks, dan lainnya. [20].

Deep Learning (DL) atau pembelajaran mendalam, adalah cabang dari *Machine Learning (ML)* yang memanfaatkan jaringan saraf tiruan (*Artificial Neural network*/*ANN*) [16] dengan struktur yang umumnya terdiri dari banyak lapisan. Proses pembelajarannya dilakukan dengan mengenali pola tersembunyi dalam data dan mengklasifikasikannya, sehingga mampu menghasilkan keluaran yang sesuai ketika menerima masukan baru. [21]. Perbedaan antara *ML* dan *DL* dapat digambarkan secara lebih jelas melalui Gambar 2.6 di bawah ini.



Gambar 2. 8 Ilustrasi Perbedaan *Machine Learning* dan *Deep Learning* [21]

Perbedaan utama *Machine Learning* dan *Deep Learning* terletak pada ekstraksi fitur. *Machine Learning* membutuhkan fitur yang diekstrak secara manual,

sedangkan *Deep Learning* mengekstraknya otomatis. Namun, *Deep Learning* memerlukan data dalam jumlah besar. Contoh penerapannya adalah pembuatan suara untuk video dan klasifikasi citra. [21].

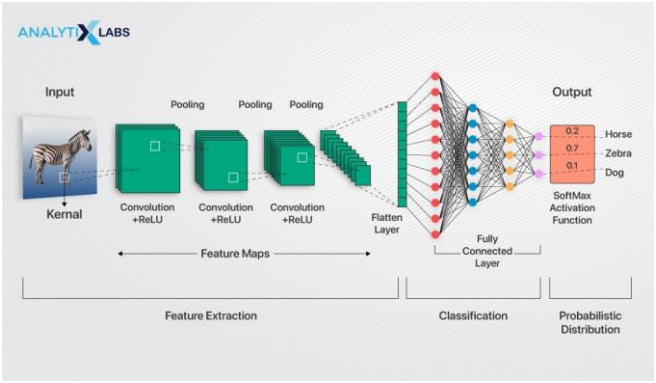
Metode *Deep Learning* merupakan pembelajaran dengan banyak tingkat representasi yang membentuk arsitektur jaringan saraf berlapis. Lapisan ini terdiri dari tiga bagian utama: *input layer, hidden layer,* dan *output layer* [22].

2.7 *Convolutional Neural network (CNN)*

Convolutional Neural network (CNN) adalah salah satu jenis jaringan saraf dalam *Deep Learning* yang dirancang khusus untuk memproses data dua dimensi, seperti gambar. *CNN* termasuk dalam jenis *Deep Neural network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra [23]. *CNN* merupakan pengembangan dari *Multilayer Perceptron* (MPL) yang memiliki arsitektur unik berupa lapisan seperti tersembunyi yang hanya terhubung ke subset neuron pada lapisan sebelumnya. Struktur ini memungkinkan *CNN* mempelajari fitur secara hierarkis:

1. **Lapisan awal** mendeteksi fitur dasar seperti tepi atau perubahan warna.
2. **Lapisan tengah** mengenali pola bentuk atau tekstur.
3. **Lapisan akhir** digunakan untuk mengenali objek secara keseluruhan.

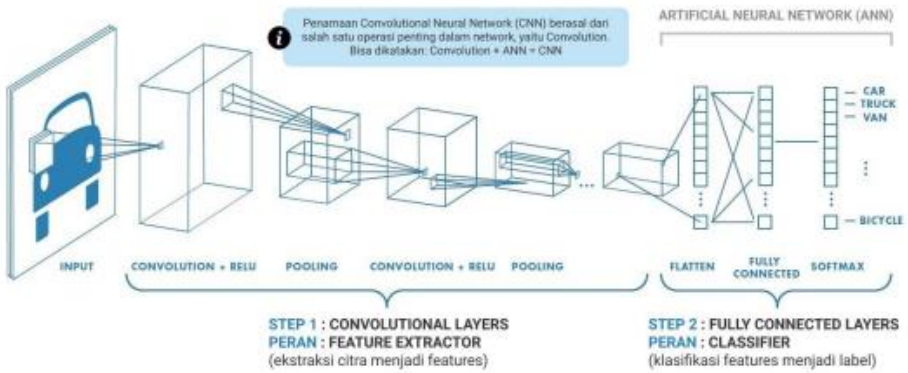
Arsitektur dari *Convolutional Neural network (CNN)* dibagi menjadi 2 bagian yaitu *feature Extraction Layer* dan *Fully connected Layer*.



Gambar 2. 9 *Convolutional Neural Network* [24]

2.8 Cara kerja *CNN*

Prinsip kerja arsitektur *CNN* , seperti yang terlihat pada Gambar 2.9, terdiri dari berbagai lapisan (*layers*), dimulai dengan *Input Layer*, *Convolutional Layer*, Fungsi Aktivasi ReLU, *Pooling Layer, Flatten Layer, Fully connected Layer (ANN),* Fungsi Aktivasi *Softmax*, dan *Dropout*. Setiap lapisan memiliki fungsinya masing-masing dalam proses klasifikasi. [21] dapat diuraikan sebagai berikut:



Gambar 2. 10 Arsitektur *CNN* [16]

2.8.1 Input Layer

Input layer pada *CNN* berfungsi untuk merepresentasikan gambar yang dimasukkan ke dalam model. Jika gambar yang diproses memiliki ukuran 224x224 piksel dan menggunakan format RGB (*Red, Green, Blue*), maka gambar *input*

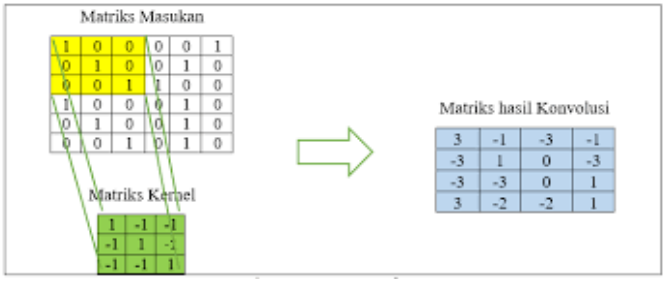
tersebut akan berbentuk *array* multidimensi dengan ukuran 224x224x3, di mana angka 3 menunjukkan jumlah channel warna RGB.

2.8.2 Convolutional Layer

Convolutional Layer merupakan komponen utama dalam *CNN* yang menggunakan *filter* atau kernel dengan bobot acak yang diperbarui saat pelatihan. *Filter* ini bergerak melintasi citra untuk mengekstraksi fitur seperti tepi, sudut, dan tekstur. Hasilnya adalah *feature map* yang menyimpan informasi penting dari citra input. [16].

2.8.3 Fungsi Aktivasi ReLU

Fungsi aktivasi ReLU (*Rectified Linear Unit*) digunakan setelah operasi konvolusi dan sebelum *pooling*, serta pada neuron di lapisan tersembunyi pada fase *fully connected*. ReLU menghasilkan *output* yang sama dengan *input* jika *input* bernilai positif, dan outputnya menjadi nol jika *input* negatif. Dengan kata lain, ReLU mengubah nilai negatif menjadi nol dan membiarkan nilai positif tetap. [21].

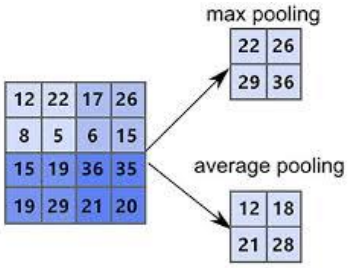


Gambar 2. 11 Ilustrasi Proses ReLu [25]

2.8.4 Pooling Layer

Pooling Layer menggunakan *filter* untuk mengurangi dimensi *feature map (downsampling)* guna mempercepat pelatihan dan mengurangi jumlah bobot. Dua jenis *pooling* yang umum adalah *Max Pooling* (mengambil nilai tertinggi) dan

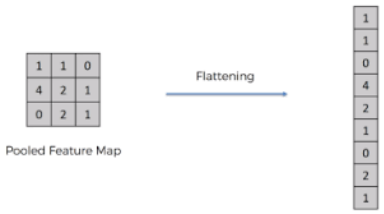
Average Pooling. Prosesnya membagi *feature map* menjadi *grid* kecil dan mengambil nilai maksimum atau rata-rata dari tiap *grid*, sambil tetap mempertahankan informasi penting [21]. *Pooling Layer* bekerja secara terpisah pada setiap kedalaman *input* dan mengurangi ukuran spasial menggunakan operasi maksimum. Umumnya, digunakan *filter* 2x2 dengan *stride* 2 untuk mengambil sampel. Pada *Max Pooling*, nilai maksimum diambil dari setiap area 2x2, sedangkan *Average Pooling* menghitung rata-ratanya..



Gambar 2. 12 Ilustrasi Proses *Average Pooling* dan *Max Pooling* [26]

2.8.5 Flatten Layer

Flatten Layer merupakan tahap sebelum *Fully Connected Layer*, di mana *feature map* yang masih berbentuk array multidimensi diubah menjadi vektor satu dimensi. Proses ini penting agar fitur dapat digunakan sebagai input untuk klasifikasi, dengan tetap mempertahankan informasi dari feature map dalam format yang sesuai [21].



Gambar 2. 13 Ilustrasi Proses *Flatten Layer* [5]

Gambar diatas menunjukkan hasil *pooling* yang telah diubah menjadi vektor satu dimensi. Proses flattening mengubah feature map menjadi vektor agar dapat digunakan sebagai input di *Fully Connected Layer* dengan format yang sesuai.

2.8.6 Fully connected Layer

Fully Connected Layer memiliki struktur yang sepenuhnya terhubung seperti pada jaringan saraf tiruan, di mana setiap node terhubung ke semua node di lapisan berikutnya. *Layer* ini menerima *input* dari proses sebelumnya untuk mengekstrak fitur penting dan mengolahnya dalam bentuk vektor satu dimensi hasil *flattening*. [21]. *Fully Connected Layer* berfungsi untuk memproses data guna melakukan klasifikasi, menghasilkan output berupa probabilitas tiap kategori, terutama saat menggunakan Softmax [21]. Persamaan *fully connected layer* sebagai berikut:

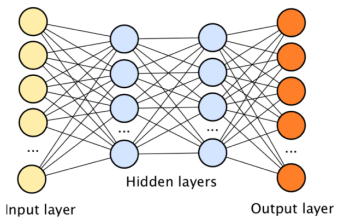
$$z = W \,.\, x + b$$

Keterangan:

z = hasil operasi linear (logit) sebelum aktivasi

W = matriks bobot

b = vektor *input*



Gambar 2. 14 Ilustrasi Proses *Fully connected Layer* [3]

2.8.7 Fungsi Aktivasi *Softmax*

Fungsi aktivasi *Softmax* diterapkan pada *output layer* untuk klasifikasi dengan lebih dari dua kelas. Fungsi ini menghitung probabilitas setiap kelas target dibandingkan dengan kelas lainnya, membantu menentukan kelas yang tepat untuk *input* citra. Keunggulan *Softmax* adalah outputnya berupa probabilitas antara 0 hingga 1, dengan total probabilitas dari semua kelas selalu berjumlah satu. [21].

$$Softmax\ (Zi) = \frac{e^{zi}}{\sum_{j=1}^{k} e^{zj}}$$

Keterangan:

z_i = logit neuron ke- i (nilai keluaran dari *fully connected layer* sebelum fungsi aktivasi diterapkan)

e^{zi}= nilai eksponensial dari logit ke- i , membuat semua nilai menjadi positif dan memperbesar perbedaan antar logit

$\sum_{j=1}^{k} a^{zj}$= jumlah seluruh nilai eksponensial dari semua neuron output (penyebut yang berfungsi sebagai faktor normalisasi)

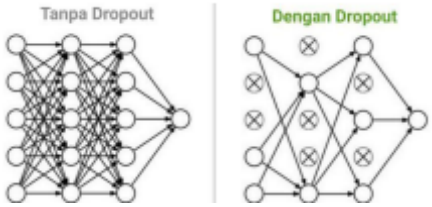
k = jumlah kelas

$softmax\ (z_i)$ = $\dot{y}i$: probabilitas prediksi model bahwa sampel termasuk kelas ke- i ; $\sum_{j=1}^{k} a^{zj}$ $\dot{y}i = 1$

2.8.8 *Dropout Regularization*

Dropout adalah teknik regularisasi untuk mencegah *overfitting* pada *neural network*. Dalam metode ini, neuron-neuron dipilih secara acak untuk dinonaktifkan selama pelatihan, sehingga beberapa fitur diabaikan untuk mengurangi variansi.

Neuron yang dinonaktifkan tidak memberikan kontribusi pada proses *forward pass* dan bobotnya tidak diperbarui selama *backpropagation*. Selain mengurangi *overfitting*, dropout juga mempercepat pelatihan model. [21].



Gambar 2. 15 Ilustrasi Penggunaan *Dropout* [16]

2.9 *Training and Testing di CNN*

Proses pelatihan *CNN* melibatkan feed forward, perhitungan error, backpropagation, dan update bobot dengan optimizer hingga model akurat, sedangkan saat pengujian hanya dilakukan *feed forward* dan perhitungan error untuk menghasilkan prediksi yang dievaluasi menggunakan bobot hasil pelatihan, dengan *categorical cross entropy* sebagai *loss function* untuk klasifikasi multi-kelas.

2.10 *Underfitting, overfitting & hyperparameter*

Underfitting dan *Overfitting* model adalah hal yang bisa terjadi ketika membuat model *CNN* . Model yang *Underfitting* atau *overfitting* tidak akan melakukan prediksi dengan benar, berikut penjelasan:

2.10.1 Underfitting

Underfitting Terjadi ketika model tidak bisa melihat logika dibelakang data, hingga tidak bisa melakukan prediksi dengan tepat, baik untuk dataset *training*

maupun dataset *testing*. *Underfitting* model akan memiliki *loss* tinggi dan akurasi rendah.

2.10.2 Overfitting

Overfitting terjadi saat model terlalu menyesuaikan diri dengan data latih hingga sulit memprediksi data uji secara akurat karena menangkap noise. Model yang baik seharusnya mengenali pola tanpa terganggu noise dan tetap akurat pada data baru. Dalam *CNN* , *overfitting* umum terjadi, namun dapat diatasi dengan *tuning hyperparameter*.

2.10.3 Hyperparameter

Hyperparameter adalah parameter yang ditentukan oleh perancang model *CNN* , untuk mengontrol proses pelatihan. Penyesuaian *Hyperparameter* secara eksperimental diperlukan agar memperoleh akurasi yang optimal. Berikut beberapa *hypermparameter* yang bisa disesuaikan dalam model *CNN* :

1. *Input shape*

   Model *CNN* mungkin tidak dapat mengenali fitur dalam citra dengan baik jika ukurannya terlalu kecil. Oleh karena itu, ukuran citra sebagai *input* harus diperhatikan. Beberapa arsitektur *CNN* populer merekomendasikan ukuran 224x224 piksel, dimana 3 mewakili channel RGB.

2. *Batch size*

   Pelatihan data (*training*) keseluruhan sekaligus memerlukan komputasi yang barat, terutama jika dataset berjumlah ribuan dan berupa citra. *Batch size* berfungsi untuk membagi dataset menjadi beberapa bagian kecil agar lebih efisien. Misalnya, jika dasaset berisi 3200 data dengan batch size 64, maka

diperlukan 50 iterasi per epoch (50x60=3200). Selain itu, batch size membantu *optimizer* menghindari jebakan local optimum karena bobot diperbarui setiap iterasi, bukan setiap epoch. Nilai *batch size* yang umum digunakan adalah 16, 32, 64, 128, atau 256.

3. *Epoch*

   Satu *epoch* berarti satu putaran penuh pelatihan terhadap seluruh dataset. Tidak ada keterangan yang pasti terkait berapa jumlah *epoch* yang optimal, karena akan berbeda untuk kumpulan data yang berbeda. Sebagai acuan, hentikan *epoch* saat nilai *loss* sudah *converge* (tidak turun lagi) atau akurasi sudah tidak mengalami peningkatan yang signifikan.

4. Jumlah filter

   Jumlah filter pada *Convolutional Layer* bisa disesuaikan dan tidak ada keterangan yang pasti terkait berapa jumlah filter yang optimal. Namun secara umum, banyak penelitian yang menggunakan nilai pangkat dua (2^n) seperti 8/16/32/64/128/256.

5. *Filtersize*

   Ukuran filter pada *Convolutional Layer* dan *Pooling Layer* bisa disesuaikan, tetapi umumnya filter konvolusi berukuran 3x3 atau 5x5 dengan *stride* 1 atau 2, sedangkan untuk *Pooling* yaitu 2x2 dengan *stride* 2 yang berfungsi untuk mengurangi ukuran citra hingga setengah dari ukuran aslinya.

6. *Stride*

   Stride adalah parameter yang menentukan berapa jumlah pergeseran filter. Jika nilai stride=1, maka filter akan bergeser sebanyak 1 piksel secara horizontal lalu

vertikal hingga seluruh area terkena efek filter.

7. *Padding*

   Padding adalah Parameter yang menentukan jumlah piksel bernilai 0 yang akan ditambahkan di setiap sisi dari *input* untuk memanipulasi dimensi *output* dari proses *convolutional* atau *pooling*. Dalam keras, *padding*=”sama” menjaga dimensi *output* tetap sama dengan *input*, sementara *padding* =”*valid*” mengurangi dimensi tanpa penambahan piksel.

8. *Optimizer*

   Optimize adalah algoritma yang digunakan untuk memperbarui bobot dalam jaringan saraf guna menurunkan nilai *loss* dan meningkatkan akurasi. Salah satu *optimizer* terbaik dan paling sering digunakan dikalangan para peneliti adalah *Adam Optimizer*, karena kemampuannya yang cepat dalam mencapai nilai *loss* minimum (*converge*).

9. *Learning rate*

   Learning rate adalah Parameter yang mengontrol seberapa cepat atau lambat model mempelajari masalah pada saat pelatihan. Jika terlalu kecil, model memerlukan waktu yang lama untuk mencapai bobot yang optimal. Jika terlalu besar, model bisa melewati bobot optimal tanpa mencapainya. Beberapa nilai *learning rate* yang umum digunakan, yaitu 0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3.

10. Jumlah *hidden layer*

    Jumlah *hidden Layer* di tahap *Fully connected* dapat disesuaikan sesuai kebutuhan, tanpa aturan pasti mengenai jumlah optimalnya.sebagai acuan

umum, bisanya digunakan minimal 2 *hidden layer* untuk memastikan model dapat menangkap pola yang lebih kompleks.

11. Jumlah neuron

    Jumlah neuron di setiap *Hidden Layer* dapat disesuaikan, tetapi umumnya menggunakan nilai dalam bentuk 2^n (misalnya: 8/16/32/64/128/256). Semakin banyak jumlah neuron yang digunakan, semakin besar pula ukuran model *CNN* yang dihasilkan (bahkan bisa sampai ratusan MB, seperti VGG).

2.11 *K-Fold Cross Validation*

K-Fold Cross Validation adalah metode statistik untuk mengevaluasi model dengan membagi data latih menjadi dua subset: *training* dan *validation*. Model dilatih menggunakan data *training* dan divalidasi menggunakan data *validation* untuk mengukur akurasi model. Teknik ini membagi data menjadi K subset dan mengulangi proses pelatihan sebanyak K kali, menghasilkan K model dari tahap *training* dan *validation*. Umumnya, nilai K yang digunakan adalah 5 atau 10, meskipun tidak ada aturan baku.

2.12 *VGG16*

Arsitektur *Convolutional Neural Network (CNN)* yang diperkenalkan pada tahun 2014 memiliki keunggulan dalam desainnya yang sederhana serta penggunaan kernel berukuran kecil, yaitu 3×3. Ukuran kernel yang kecil ini membantu mengurangi jumlah parameter, sehingga proses pelatihan dan pengujian menjadi lebih efisien dan ringan. Selain itu, penggunaan kernel kecil memungkinkan jaringan memiliki kedalaman yang lebih tinggi dan meningkatkan

kemampuan dalam mendeteksi fitur [27]. *VGG16* adalah salah satu contoh arsitektur dari metode *CNN* .

Secara umum, perbedaan antara *CNN* dan *VGG16* adalah sebagai berikut:

1. *CNN* merupakan sebuah metode dalam bidang *deep learning*, sedangkan *VGG16* adalah arsitektur spesifik yang digunakan dalam metode tersebut.
2. *VGG16* merupakan salah satu implementasi arsitektur dari metode *Convolutional Neural Network* (*CNN*).

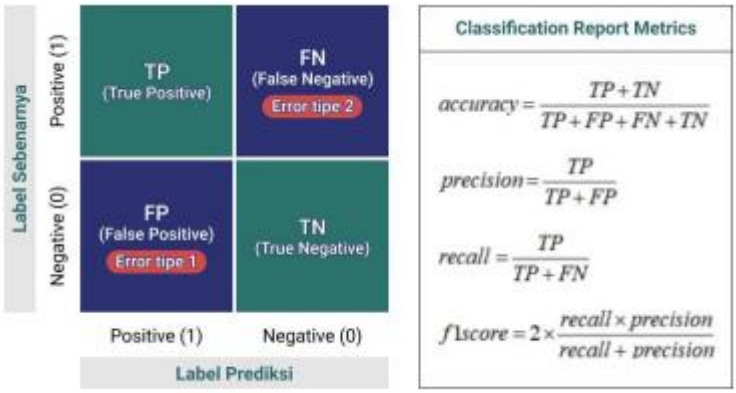
Gambar 2.16 Arsitektur *VGG16* [27]

Pada Gambar 2.13, arsitektur ini terdiri dari 16 lapisan yang mencakup lapisan *Convolutional* dan *Fully Connected*. Citra yang digunakan sebagai input memiliki resolusi 224 × 224 piksel dengan nilai *learning rate* sebesar 0.0001 [27]. Arsitektur ini menggunakan kernel berukuran 3 × 3 dengan kedalaman 64, yang disesuaikan dengan format citra berwarna yang memiliki tiga saluran, yaitu RGB (*Red, Green, Blue*). Selain itu, terdapat proses *Max Pooling* yang berfungsi untuk mengekstraksi nilai maksimum dari area kecil, misalnya berukuran 2 × 2, guna merepresentasikan fitur paling penting dari citra. Setelah data melalui 13 lapisan konvolusi, proses

dilanjutkan ke dua lapisan *Fully Connected*, seperti ditunjukkan dalam Gambar 2.16, dan akhirnya diakhiri dengan penerapan fungsi aktivasi Softmax.

2.13 *Confusion matrix* and *classification report*

Dalam *ML* /*DL* , klasifikasi merupakan bagian dari *supervised learning*. Langkah penting dalam *life cycle* model *ML* /*DL* adalah evaluasi performanya. Terdapat dua teknik yang dapat digunakan untuk mengevaluasi model klasifikasi yaitu *Confusion matrix* dan *Classification Report*. Berikut penjelasan lebih detailnya:



Gambar 2.17 : (kiri) *Confusion Matrix*, (kanan) Rumus metriks di *Classification Report*

2.13.1 Confusion Matrix

Confusion matrix adalah tabel berukuran N x N (N = jumlah kelas) yang berisi yang menunjukkan jumlah prediksi yang benar dan salah paham model klasifikasi. Matrix ini digunakan untuk membandingkan nilai aktual dengan nilai prediksi. Baris mewakili kelas aktual dan kolom mewakili kelas prediksi. *Confusion matrix* membantu dalam menganalisis performa model dengan lebih detail,

terutama dalam melihat kesalahan klasifikasi antar kelas. Nilai yang dikembalikan oleh *Confusion Matrix* dibagi kedalam 4 kategori:

1. *True Positive* (TP): Prediksi positif & nilai sebenarnya positif.
2. *True Negative* (TN): Prediksi negatif & nilai sebenarnya negatif.
3. *False Positive* (FP): Prediksi positif & nilai sebenarnya negatif.
4. *False Negative* (FN): Prediksi negatif & nilai sebenarnya positif.

2.13.2 Classification Report

Meskipun *Confusion Matrix* memberikan informasi detail, memahami kinerja model secara keseluruhan tetap sulit. Oleh karena itu, metrik evaluasi seperti *Classification Report* digunakan untuk menilai performa model berdasarkan data dari *Confusion Matrix*. Berikut matrix yang digunakan:

1. *Accuracy*: presentasi prediksi yang benar dari seluruh data.
2. *Precision*: kemampuan model dalam menghindari *false positive*.
3. *Recall*: kemampuan model dalam mendeteksi kelas yang benar (menghindari *false negative*).
4. *F1-Score*: Menggambarkan perbandingan rata-rata *Precision* dan *Recall* yang memberikan keseimbangan antara keduanya.

Dalam kebanyakan kasus klasifikasi dunia nyata, data sering tidak seimbang, sehingga *F1-Score* lebih direkomendasikan untuk mengevaluasi performa model.

2.14 *Phyton*

Python adalah bahasa pemrograman yang sangat populer dan banyak digunakan oleh berbagai kalangan, mulai dari akademisi hingga masyarakat umum,

untuk berbagai kebutuhan. Penggunaannya meliputi pembuatan aplikasi *desktop, website, game*, analisis data, hingga pengembangan proyek kecerdasan buatan, termasuk implementasi metode *CNN* [16]. Berikut ini adalah beberapa *library Python* yang sering digunakan dalam *Convolutional Neural network* (*CNN*).

2.14.1 TensorFlow

TensorFlow adalah *library* yang dikembangkan oleh *Google* untuk memenuhi kebutuhan *machine learning* (*ML*) dan *Deep Learning* (*DL*). *Library* ini mendukung berbagai aplikasi *ML* /*DL* , seperti regresi dan klasifikasi, dan dapat dijalankan di CPU, GPU, bahkan pada sistem operasi *mobile*. *TensorFlow* memiliki tingkat fleksibilitas yang tinggi sehingga memungkinkan banyak penyesuaian sesuai kebutuhan, tetapi penggunaannya cukup kompleks dan kurang ideal bagi pemula [24].

2.14.2 Keras

Keras adalah API jaringan saraf tingkat tinggi yang memudahkan penerapan *Deep Learning* dan *artificial intelligence*. Ditulis dalam bahasa *Python*, *keras* dapat berjalan di atas *TensorFlow*, CNTK, atau Theano dan dikenal sebagai *library neural network* yang ramah pengguna. Dengan kemajuan dalam *deep learning*, *keras* menyediakan berbagai *pretrained models*, yaitu model yang telah dilatih sebelumnya, yang dapat digunakan untuk menangani berbagai tugas, seperti klasifikasi gambar. Dalam *keras*, *pretrained models* ini dikelompokkan dalam kelas *Applications* [28].

2.14.3 cv2 (*OpenCV*)

cv2 (*OpenCV*) adalah *library* yang digunakan dalam pengolahan citra digital dan computer vision. Dalam konteks *Convolutional Neural network* (*CNN*), cv2 berperan dalam berbagai proses, seperti membaca *file* citra (misalnya jpg, png), mengkonversi citra menjadi array untuk pemrosesan lebih lanjut, mengubah format warna citra (contohnya dari RGB ke *Grayscale*), dan mengubah ukuran citra (contohnya dari 500x500 piksel ke 224x224 piksel) [24].

2.14.4 Sklearn

Library ini digunakan sebagai alat untuk berbagai model *machine learning* dan statistik, seperti klasifikasi, regresi, *clustering*, serta pengurangan dimensi. Dalam konteks *Convolutional Neural network* (*CNN*), *library* ini mendukung penerapan teknik evaluasi seperti *K-Fold Cross Validation* untuk memvalidasi dataset, serta pembuatan *Confusion Matrix* dan *Classification Report*, yang berguna dalam menilai performa model, termasuk tingkat akurasi, presisi, dan *recall* pada tiap kelas [24].

2.14.5 Numpy

Library ini digunakan untuk memproses *array* dan *matriks*, seperti merubah dimensi citra, menghitung rata-rata serta standar deviasi akurasi model *CNN* , dan pengolahan lain berbasis *array* [24].

2.14.6 Pandas

Library ini digunakan untuk manipulasi dan analisis data. Dalam konteks *CNN* , fungsinya mencakup penyimpanan riwayat pelatihan, seperti akurasi dan *loss*, ke dalam tabel berformat CSV [24].

2.14.7 Matplotlib

Library ini berfungsi untuk visualisasi data. Dalam konteks *CNN* , *library* ini digunakan untuk memetakan dataset beserta labelnya serta menampilkan hasil pelatihan dan pengujian model *CNN* dalam bentuk grafik [24].

2.14.8 Seaborn

Library ini digunakan untuk membuat visualisasi data yang lebih menarik dan informatif dengan dasar *Matplotlib*. Dalam konteks *CNN* , contohnya adalah pembuatan heatmap untuk *Confusion Matriks* guna mempermudah dan memperjelas analisis [24].

2.15 *Jupyter Notebook*

Jupyter Notebook adalah aplikasi *web* sumber terbuka yang memungkinkan pengguna membuat dan berbagi dokumen interaktif yang menggabungkan kode, visualisasi, teks naratif, dan elemen lainnya. Aplikasi ini sangat populer di kalangan data *scientist* dan pengembang karena kemampuannya untuk menggabungkan berbagai elemen dalam satu dokumen yang dapat dijalankan dan diakses secara interaktif. Mendukung berbagai bahasa pemrograman seperti *Python, R,* dan *Julia*, *Jupyter Notebook* memungkinkan eksekusi kode dalam sel terpisah, yang memungkinkan pengguna menjalankan dan menguji kode serta melihat hasilnya langsung di bawah sel. Selain itu, *Jupyter Notebook* juga memungkinkan pembuatan dan pengolahan visualisasi interaktif. Keunggulannya terletak pada fleksibilitas dalam eksplorasi, analisis, dan berbagi hasil eksperimen atau proyek data. Dokumen *Jupyter* dapat diekspor ke berbagai format seperti HT*ML* , PDF, atau slide presentasi untuk memudahkan berbagi penemuan secara interaktif [29].

2.16 *Flask*

Flask adalah *framework web* bersifat minimalis yang dibuat menggunakan bahasa pemrograman *Python*. *Framework* ini dirancang untuk mempermudah *developer* dalam membangun aplikasi *web*. Walaupun strukturnya tergolong sederhana, *flask* tetap mampu memberikan performa yang baik dan dapat diandalkan, baik untuk pengembangan aplikasi *web* berskala kecil maupun besar [30].

2.17 Penelitian Terkait

Berikut ini adalah beberapa ulasan mengenai penelitian terdahulu yang berkaitan dengan data dan metode yang digunakan, seperti yang ditampilkan pada Tabel 2.1.

Tabel 2. 1 Penelitian Terdahulu

No	Nama dan Tahun	Judul	Metode	Hasil
1.	(Yani Parti Astuti, 2023)	Implementasi Algoritma *Convolutional Neural network (CNN)* untuk Klasifikasi Jenis Tanah Berbasis *Android*	*Convolutional Neural network* (*CNN*)	Penelitian ini berhasil mengimplementasikan algoritma *Convolutional Neural network (CNN)* untuk klasifikasi jenis tanah berbasis *android*. Model *CNN* yang dibangun menggunakan *input shape* 150 x 150 *piksel*, *learning rate* 0.001, ukuran filter 3 x 3, dan 100 *epoch* dengan 704 data *training* dan 88 data validasi menghasilkan akurasi *training* sebesar 97% dan akurasi *testing* sebesar 95%. Pengujian aplikasi secara manual menggunakan 10 gambar dan hasil observasi menunjukkan 8 gambar diklasifikasikan dengan benar dan 2 gambar salah klasifikasi, menghasilkan akurasi 80%.

2.	(Fathoni Dwiatmoko, 2024)	Klasifikasi Citra Sampah Organik dan Non Organik Menggunakan Algoritma *CNN* (*Convolutional Neural network*)	*Convolutional Neural network* (*CNN*)	Penelitian ini menggunakan algoritma *Convolutional Neural network (CNN)* untuk klasifikasi sampah organik dan anorganik. Dataset terdiri dari 2800 gambar dengan 1400 gambar untuk setiap kelas, dan parameter yang digunakan adalah *epoch* 50 dan batch_size 32. Pengujian dengan 200 gambar menghasilkan akurasi sebesar 99% dan nilai *loss* sebesar 0.005, menunjukkan model mampu mengenali jenis sampah dengan sangat baik [31].
3.	(Susi Yuliany, Aradea, Andi Nur Rachman 2022)	Implementasi *Deep Learning* pada Sistem Klasifikasi Hama Tanaman Padi Menggunakan Metode *Convolutional Neural network* (*CNN*)	*Convolutional Neural network* (*CNN*)	Penelitian ini menggunakan metode *Convolutional Neural network (CNN)* untuk klasifikasi hama pada tanaman padi dengan mengatasi masalah *Overfitting* melalui pembagian data dan pengaturan parameter. Pembagian data terbaik adalah 90%:10% untuk *training* dan *testing*, dengan arsitektur yang mencapai akurasi *training* 83,02%, 78,30%, dan 81,13%. Hasil akurasi pengujian dari model-model tersebut adalah 69,33%, 77,33%, dan 76% [32].
4.	(Budi Yanto, Luth Fimawahib, Asep Supriyanto, B. Herawan Hayadi, Rinanda Rizki Pratama, 2021)	Klasifikasi Tekstur Kematangan Buah Jeruk Manis Berdasarkan Tingkat Kecerahan Warna dengan Metode *Deep Learning Convolutional Neural network*	*Convolutional Neural network* (*CNN*)	Penelitian ini menggunakan algoritma *Convolutional Neural network (CNN)* untuk klasifikasi kualitas jeruk manis dengan akurasi 97.5184% pada *training* dan 92% pada *testing*, menunjukkan hasil yang sangat baik. Pengujian menggunakan 10 citra jeruk (5 jeruk bagus dan 5 jeruk busuk) menghasilkan akurasi 96% untuk *training*. Hasil ini menegaskan kemampuan model dalam mengklasifikasikan kelayakan jeruk manis secara efektif [22].
5.	(Muhammad Alfin Ji*ML* y Asshiddiqie, 2020)	Deteksi Tanaman Tebu Pada Lahan Pertanian Menggunakan Metode *Convolutional Neural network*	*Convolutional Neural network (CNN)* dan YOLO (*You Only Look Once*)	Penelitian ini mengembangkan sistem deteksi jenis tanaman berbasis *drone* dengan algoritma *CNN* dan YOLO, berhasil mendeteksi tebu dengan rata-rata *confidence* 95%. Pengujian menunjukkan skor *precision* 1.00, *recall* 0.95, dan *accuracy* 0.95 pada tebu, menjadikannya efektif untuk identifikasi tanaman dari udara [25].

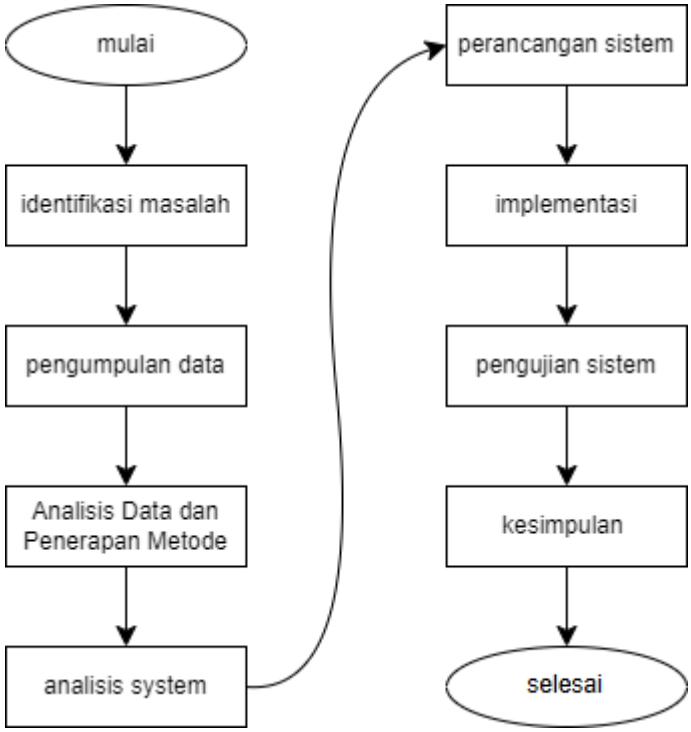
6.	(Susi Yuliany, Aradea, Andi Nur Rachman, 2022)	Implementasi *Deep Learning* untuk Image *Classification* Menggunakan Algoritma *Convolutional Neural network (CNN)* pada Citra Kebun dan Sawah	*Convolutional Neural network* (*CNN*)	Penelitian ini tentang klasifikasi citra kebun dan sawah menggunakan metode *CNN* dengan dataset 100 citra mengalami *Overfitting*. Pembagian data 80% ; 20% menghasilkan akurasi pelatihan 96,25% dan akurasi validasi 75%, serta mencapai akurasi 75% pada 20 data uji untuk dua kelas [33].
7.	(Muhammad Athoillah, Rani Kurnia Putri, 2023)	Identifikasi Jenis Kendaraan Bermotor dengan Algoritma *Convolutional Neural networks*	*Convolutional Neural network* (*CNN*)	Hasil penelitian mengungkapkan bahwa deteksi jenis kendaraan bermotor menggunakan *CNN* memberikan kinerja optimal dalam mendukung manajemen lalu lintas, penegakan hukum, serta keamanan pada sistem transportasi pintar. Model ini efektif dalam mengidentifikasi pola dan fitur pada citra kendaraan, dengan hasil pengujian mencapai presisi rata-rata sebesar 97,00%, recall 97,60%, spesifisitas 97,59%, dan akurasi 97,30% [34].
8.	(Mohammad Amin Triwinanto, Aang Alim Murtopo, Syefudin, Gunawan Gunawan, 2024)	Penerapan Algoritma *Convolutional Neural network (CNN)* untuk Identifikasi Lahan Kosong di Kota Tegal Berdasarkan Citra *Google Earth*	*Convolutional Neural network* (*CNN*)	Penelitian ini berhasil mengklasifikasikan jenis lahan kosong tambak, rawa, pemukiman, dan sawah menggunakan *CNN* . Dengan dataset 120 gambar yang dibagi 70:30 untuk data latih dan uji, model *CNN* terdiri dari empat *convolutional layer* (filter: 32, 32, 64, 64) dan satu *hidden layer* dengan 512 neuron. Pengujian menunjukkan akurasi 80,5% pada data uji.
9.	(Ummi Sri Rahmadhani , Noveri Lysbetti Marpaung, 2023)	Klasifikasi Jamur Berdasarkan Genus dengan Menggunakan Metode *CNN*	*Convolutional Neural network* (*CNN*)	Model yang dikembangkan terdiri dari tiga *convolutional layer*, tiga *Max Pooling layer*, dan dua *dropout layer* untuk mengurangi *overfitting*. Dataset yang digunakan mencakup 1.200 gambar, dengan pembagian 70:30 untuk data latih dan uji, menghasilkan 840 gambar untuk pelatihan dan 360 gambar untuk pengujian. Model ini mencapai akurasi terbaik sebesar 89% pada data latih dan 82% pada data validasi, menunjukkan kemampuan yang efektif dalam

				mengklasifikasikan jamur berdasarkan Genus [35].
10.	(M. Ridha, 2022)	Pendeteksi *Fake Masker* Menggunakan Metode *Convolutional Neural network (CNN)* dengan Arsitektur *Xception* Skripsi	*Convolutional Neural network* (*CNN*)	Penelitian ini menggunakan arsitektur *Xception* untuk mendeteksi tiga kategori objek: wajah bermasker, wajah tanpa masker, dan wajah dengan masker palsu. Berdasarkan analisis yang dilakukan, akurasi mencapai 99,53% dengan 10 *epoch* pelatihan, dan 99,30% dengan 5 epoch. Hasil ini menunjukkan bahwa arsitektur *Xception* sangat andal untuk klasifikasi ini, dengan akurasi yang meningkat seiring jumlah *epoch* yang bertambah.

BAB 3

METODOLOGI PENELITIAN

Penelitian ini dilakukan dengan melaksanakan tahapan demi tahapan yang berhubungan. Tahapan-tahapan tersebut dijabarkan dengan metode penelitian. Metode penelitian diuraikan ke dalam bentuk skema yang jelas, teratur, dan sistematis. Berikut tahapan-tahapan penelitian dapat dilihat pada Gambar 3.1 dibawah:



Gambar 3. 1 Tahapan Metode Penelitian

Tahapan-tahapan penelitian pada gambar 3.1 dapat dilihat pada penjelasan berikut:

3.1 Mulai

Tahap ini merupakan langkah awal dalam proses penelitian, di mana peneliti menetapkan latar belakang masalah, tujuan penelitian, serta signifikansi penelitian. Tahapan ini bertujuan untuk memberikan arah yang jelas pada proses penelitian sekaligus menentukan fokus utama yang ingin dicapai.

3.2 Identifikasi Masalah

Pada tahap ini, peneliti mengidentifikasi permasalahan utama yang akan dipecahkan, yaitu bagaimana menerapkan metode *CNN* untuk mengklasifikasikan jenis tanah gambut. Tantangan utama dalam klasifikasi ini meliputi ketepatan identifikasi dan keakuratan model. Analisis terhadap masalah ini menjadi landasan untuk memilih solusi yang tepat.

3.3 Pengumpulan Data

Pada tahap ini peneliti mengumpulkan dataset berupa citra tanah gambut yang diperoleh dari pemotretan langsung, kecuali citra bukan tanah gambut. Total dataset yang digunakan berjumlah 800 citra resolusi 224x224 piksel dengan kategori tanah gambut saprik, hemik, dan fibrik serta satu kelas bukan gambut.

3.4 Analisis Data dan Penerapan Metode

Peneliti menganalisis data untuk memahami karakteristik masing-masing jenis tanah gambut sebelum menerapkan metode *Convolutional Neural network* (*CNN*). Tahapan ini mencakup:

1. *Processing* data: mencakup normalisasi piksel ke rentang [0,1], augmentasi data untuk memperkaya variasi dataset, serta pengurangan noise.

2. Pembuatan model *CNN* : mencakup arsitektur jaringan lapisan *convolurion, pooling, dan fully connected*

3.5 Analisis sistem

Analisis sistem bertujuan untuk memahami alur kerja dalam klasifikasi citra tanah gambut menggunakan metode *Convolutional Neural Network* (*CNN*). Sistem ini mengklasifikasikan citra gambut melalui tahapan *preprocessing*, pelatihan model, dan klasifikasi. Citra diproses melalui *convolutional layer* untuk ekstraksi fitur dan *pooling layer* untuk mereduksi dimensi, lalu dilanjutkan ke *fully connected layer* hingga menghasilkan *output* klasifikasi. Pemilihan parameter seperti ukuran *input*, *epoch, batch size*, dan *learning rate* juga diperhatikan agar sesuai dengan karakteristik dataset dan mendukung akurasi model secara optimal.

3.6 Perancangan sistem

Perancangan sistem dalam penelitian ini bertujuan untuk membangun model klasifikasi citra tanah gambut menggunakan metode *convolutional neural network* (*CNN*). Sistem ini terdiri dari beberapa komponen utama, yaitu:

1. **Pengumpulan Data**: Dataset yang digunakan mencakup citra tanah gambut dan bukan tanah gambut yang telah diklasifikasikan ke dalam data pelatihan dan data validasi.
2. **Pra-pemprosesan Data**: Meliputi proses normalisasi, augmentasi citra, serta pembagian dataset ke dalam data pelatihan dan validasi.
3. **Pembangunan Model *CNN*** : Arsitektur *CNN* dikembangkan untuk mengenali dan mengklasifikasikan citra berdasarkan fitur visual yang diekstraksi.

4. **Pelatihan dan Evaluasi Model**: Model dilatih menggunakan dataset yang telah diproses, kemudian dievaluasi dengan menggunakan matrik evaluasi performa.
5. **Implementasi Antarmuka *Web* dengan *Flask***: Model yang telah dilatih diintegrasikan ke dalam aplikasi berbasis web menggunakan *framework Flask*, sehingga pengguna dapat melakukan klasifikasi citra secara praktis dan interaktif.

3.7 Implementasi Sistem

Pada tahap implementasi, model *CNN* dikembangkan dan diuji menggunakan:

1. **Perangkat keras (Hardware):**

   Processor : Intel (R) Core (TM) i5-6200U Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz (4 CPUs), ~2.4GHz

   Memory : 16 GB

   Operating system : Windows 11 Pro 64-bit (10.0, Build 22621)

   Hard Disk : 238,46 GB

2. **Perangkat Lunak:**

   Sistem Operasi : *Windows* 11

   Tools : *Jupyter Notebook, Visual Studio Code,* dan *Framework Flask*

3.8 Pengujian Sistem

Pengujian dilakukan untuk memastikan model dalam kondisi beragam. Data uji dipisahkan dari dataset dengan rasio 80% *training* dan 20% *validation*. Pengujian ini mengevaluasi akurasi model terhadap data yang belum pernah dilatih.

3.9 Kesimpulan

Tahap akhir adalah merumuskan kesimpulan berdasarkan hasil yang diperoleh. Kesimpulan ini mencakup efektivitas penerapan *CNN* dalam klasifikasi tanah gambut serta rekomendasi untuk pengembangan lebih lanjut dalam bidang ini.