

**SKRIPSI**

**IDENTIFIKASI JENIS SERANGAN HAMA PADA BUAH KAKAO  
MENGUNAKAN *IMAGE PROCESSING***

**Disusun dan diajukan oleh:**

**NURUL MUSFIRAH  
D421 16 521**



**PROGRAM STUDI SARJANA TEKNIK INFORMATIKA  
FAKULTAS TEKNIK  
UNIVERSITAS HASANUDDIN  
GOWA  
2023**

**LEMBAR PENGESAHAN SKRIPSI**

**IDENTIFIKASI JENIS SERANGAN HAMA PADA BUAH KAKAO  
MENGUNAKAN IMAGE PROCESSING**

**Disusun dan diajukan oleh**

**NURUL MUSFIRAH**

**D42116521**

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika Fakultas Teknik Universitas Hasanuddin pada tanggal 12 Juni 2023 dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama,

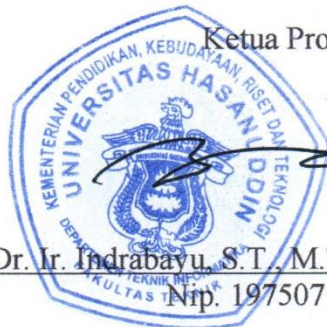
Pembimbing Pendamping,



Prof. Dr. Ir. Indrabayu, S.T., M.T.,  
M.Bus.Sys., IPM., ASEAN.Eng.  
Nip. 197507162002121004

Dr. Ir. Ingrid Nurtanio, MT  
Nip. 196108131988112001

Ketua Program Studi,



Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM., ASEAN.Eng.  
Nip. 19750716 200212 1 004

## PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini ;

Nama : Nurul Musfirah  
NIM : D42116521  
Program Studi : Teknik Informatika  
Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

Identifikasi Jenis Serangan Hama Pada Buah Kakao Menggunakan *Image Processing*

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain dan bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Semua informasi yang ditulis dalam skripsi yang berasal dari penulis lain telah diberi penghargaan, yakni dengan mengutip sumber dan tahun penerbitannya. Oleh karena itu semua tulisan dalam skripsi ini sepenuhnya menjadi tanggung jawab penulis. Apabila ada pihak manapun yang merasa ada kesamaan judul dan atau hasil temuan dalam skripsi ini, maka penulis siap untuk diklarifikasi dan mempertanggungjawabkan segala resiko.

Segala data dan informasi yang diperoleh selama proses pembuatan skripsi, yang akan dipublikasi oleh Penulis di masa depan harus mendapat persetujuan dari Dosen Pembimbing.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan isi skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Gowa, 14 Juni 2023

Yang Menyatakan



Nurul Musfirah

## ABSTRAK

**NURUL MUSFIRAH** Identifikasi Jenis Serangan Hama Pada Buah Kakao Menggunakan *Image Processing* (dibimbing oleh Prof. Dr. Ir. Indrabayu., ST, MT, M.Bus.Sys., IPM, ASEAN. Eng. dan Dr. Ir. Ingrid Nurtanio, M.T.)

Kakao (*Theobroma cacao L.*) merupakan salah satu komoditas unggulan strategis perkebunan yang memegang peranan penting dalam perekonomian Indonesia sebagai penghasil devisa negara. Terlepas dari potensinya, menurut *Intenational Cocoa Organization (ICCO)* produksi kakao mengalami penurunan dari tahun ke tahun. Salah satu penyebabnya yaitu adanya serangan hama, Tanaman yang terserang hama dapat menurunkan kualitas biji kakao dan yang terparah dapat merusak biji kakao. Pada tanaman kakao kebanyakan hama menyerang pada bagian buah. Buah kakao yang terkena serangan hama akan menampilkan gejala berupa bercak yang memiliki pola dan warna tertentu pada buah. Identifikasi jenis serangan hama pada buah kakao akan bermanfaat untuk mengidentifikasi gejala serangan pada buah kakao sehingga dapat segera dilakukan tindakan preventif pada tanaman kakao yang terserang hama. Terdapat beberapa jenis serangan hama yang akan diidentifikasi berdasarkan pola dan warnanya yaitu kepik buah kakao (*Helopeltis.Sp*), kutu putih dan tikus atau tupai. Pada penelitian ini akan dibuat sistem untuk mengidentifikasi jenis serangan hama dari citra buah kakao. Sistem ini menggunakan *deep learning* dengan metode *transfer learning* dengan algoritma *Convolutional Neural Network (CNN)* Arsitektur VGG16. Arsitektur VGG16 di modifikasi dan di *training* menggunakan 500 data citra dengan 4 label kelas. Adapun label pada tiap kelas yaitu *helopeltis*, kutu putih, tikus atau tupai, dan normal. Sistem kemudian di *training* menggunakan konfigurasi parameter dengan *learning rate*  $10^{-5}$ , *batchsize* 5, dan *epoch* 4. Model yang telah di *training*, kemudian diuji menggunakan 48 data *testing*, dan mendapatkan akurasi tertinggi 97.9%.

**Kata kunci** : Identifikasi serangan hama, buah kakao, *deep learning*, VGG16

## ABSTRACT

**NURUL MUSFIRAH** *Identification Types Of Pest On Cacao Pod Using Image Processing* (supervised by Prof. Dr. Ir. Indrabayu., ST, MT, M.Bus.Sys., IPM, ASEAN. Eng. and Dr. Ir. Ingrid Nurtanio, M.T)

Cocoa (*Theobroma cacao L.*) is one of the leading strategic plantation commodities which plays an important role in the Indonesian economy as a foreign exchange earner. Regardless of its potential, according to the International Cocoa Organization (ICCO) cocoa production has decreased from year to year. One of the causes is the existence of pest attacks. Plants that are attacked by pests can reduce the quality of cocoa beans and the worst can damage cocoa beans. In cocoa plants most of the pest attack on the pod. Cocoa pods that are affected by pests will show symptoms in the form of spots that have a certain pattern and color on the pod. Identification of the type of pest attack on cocoa pods will be useful for identifying symptoms of attack on cocoa pods so that preventive measures can be taken immediately on cocoa plants that are attacked by pests. There are several types of pest attacks that will be identified based on their patterns and colors, namely cocoa pod ladybugs (*Helopeltis.Sp*), mealybugs and mice or squirrels. In this study a system will be created to identify the type of pest attack from cocoa pod images. This system uses deep learning with the transfer learning method with VGG16 from Architecture Convolutional Neural Network (CNN) algorithm. VGG16 architecture was modified and trained using 500 image data with 4 class labels. The labels for each class are helopeltis, kutu putih (mealybugs), tikus (mice or squirrels), and normal. The system is then trained using a parameter configuration with a *learning rate* of  $10^{-5}$ , batch size of 5, and *epoch* of 4. The model that has been trained then tested using 48 data tests and obtains the highest accuracy of 97.9%.

Keywords: Pests Identification, Cacao Pod, Deep Learning, VGG16

## DAFTAR ISI

LEMBAR PENGESAHAN SKRIPSI .....	i
PERNYATAAN KEASLIAN .....	ii
ABSTRAK .....	iii
ABSTRACT .....	iv
DAFTAR ISI .....	v
DAFTAR GAMBAR .....	vi
DAFTAR TABEL .....	viii
DAFTAR SINGKATAN DAN ARTI SIMBOL .....	ix
DAFTAR LAMPIRAN .....	x
KATA PENGANTAR .....	xi
BAB I PENDAHULUAN .....	1
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah .....	2
1.3. Tujuan Penelitian .....	2
1.4. Manfaat Penelitian .....	3
1.5. Ruang Lingkup.....	3
BAB II TINJAUAN PUSTAKA .....	4
BAB III METODOLOGI PENELITIAN .....	24
3.1. Lokasi Penelitian .....	24
3.2. Tahapan Penelitian .....	25
3.3. Instrumen Penelitian .....	26
3.4. Teknik Pengambilan Data .....	26
3.5. Rancangan Sistem .....	27
3.6. Evaluasi Kinerja Sistem .....	40
BAB IV HASIL DAN PEMBAHASAN .....	42
4.1. Hasil Penelitian .....	42
4.2. Pembahasan .....	53
BAB V PENUTUP .....	58
5.1. Kesimpulan .....	58
5.2. Saran .....	58
DAFTAR PUSTAKA .....	59
LAMPIRAN .....	66

**DAFTAR GAMBAR**

Gambar 1 Buah kakao yang terserang hama <i>helopeltis</i> .....	4
Gambar 2 Buah kakao yang terserang hama kutu putih .....	5
Gambar 3 Buah kakao yang terserang hama tikus atau tupai .....	6
Gambar 4 Citra <i>grayscale</i> .....	8
Gambar 5 Citra warna .....	8
Gambar 6 <i>Artificial Neural Network</i> (ANN) .....	9
Gambar 7 <i>Deep Neural Network</i> (DNN) .....	12
Gambar 8 Ilustrasi arsitektur CNN .....	12
Gambar 9 Proses <i>convolution</i> .....	13
Gambar 10 Ilustrasi <i>weights</i> dan <i>bias</i> pada <i>layer</i> .....	14
Gambar 11 Ilustrasi penggunaa <i>learning rate</i> .....	16
Gambar 12 Grafik fungsi aktivasi ReLU .....	17
Gambar 13 Proses perhitungan menggunakan fungsi aktivasi <i>softmax</i> .....	18
Gambar 14 Operasi <i>max pooling</i> .....	19
Gambar 15 Proses <i>fully connected layer</i> .....	20
Gambar 16 Arsitektur VGG16 .....	21
Gambar 17 Visualisasi Grad-CAM .....	22
Gambar 18 Lokasi peneltian .....	24
Gambar 19 Lokasi pengambilan data .....	24
Gambar 20 Tahap penelitian .....	25
Gambar 21 Teknik pengambilan data .....	27
Gambar 22 Rancangan sistem .....	27

Gambar 23 Data citra yang akan diklasifikasi .....	28
Gambar 24 Data citra yang telah dilakukan <i>backgroud removal</i> .....	29
Gambar 25 Citra yang telah di <i>resize</i> .....	29
Gambar 26 Ilustrasi augmentasi .....	30
Gambar 27 Potongan <i>source code</i> augmentation .....	31
Gambar 28 Arsitektur VGG16 .....	32
Gambar 29 Konvolusi menggunakan <i>zero padding</i> .....	34
Gambar 30 Proses perpindahan stride 1 .....	34
Gambar 31 Visualisasi <i>feature layer</i> yang dipelajari VGG16 .....	35
Gambar 32 Hasil <i>feature extraction convolution layer</i> .....	35
Gambar 33 <i>Source Code</i> parameter proses <i>training</i> .....	36
Gambar 34 Grafik proses <i>training</i> .....	37
Gambar 35 Visualisasi Grad-CAM .....	39
Gambar 36 <i>Confusion matrix</i> dari data <i>testing</i> .....	40
Gambar 37 <i>Confusion matrix</i> kelas <i>helopeltis</i> .....	41
Gambar 38 Proses konvolusi .....	47
Gambar 39 Perubahan nilai matriks jika ditambahkan ReLU .....	48
Gambar 40 Proses <i>max pooling</i> .....	48
Gambar 41 Proses perhitungan pada fungsi aktivasi <i>softmax</i> .....	50
Gambar 42 <i>Confusion matrix</i> data <i>test</i> .....	51



**DAFTAR TABEL**

Tabel 1 Skenario pembagian data .....	28
Tabel 2 Akurasi rasio pembagian data .....	42
Tabel 3 Penggunaan variasi <i>input size</i> pada rasio pembagian data <i>training</i> 70% .....	43
Tabel 4 Penggunaan variasi <i>input size</i> pada rasio pembagian data <i>training</i> 75% .....	44
Tabel 5 Penggunaan variasi <i>input size</i> pada rasio pembagian data <i>training</i> 80% .....	44
Tabel 6 Penggunaan variasi <i>input size</i> pada rasio pembagian data <i>training</i> 85% .....	44
Tabel 7 Akurasi VGG16 tanpa augmentasi .....	45
Tabel 8 Akurasi VGG16 dengan augmentasi .....	46
Tabel 9 Perbandingan <i>training time</i> dan akurasi pada <i>batch size</i> .....	49
Tabel 10 Perbandingan akurasi berdasarkan <i>learning rate</i> .....	50
Tabel 11 Nilai <i>score</i> pada data <i>test</i> .....	51
Tabel 12 Perbandingan akurasi beberapa arsitektur CNN .....	53
Tabel 13 Hasil kesalahan klasifikasi .....	53
Tabel 14 Hasil prediksi benar .....	54

**DAFTAR SINGKATAN DAN ARTI SIMBOL**

Lambang/Singkatan	Arti dan Keterangan
Rgb	Red Green Blue
e atau exp	Nilai ekponensial
$\sigma$	sigma
w	weight
P	Probabilitas

## DAFTAR LAMPIRAN

Lampiran 1 Nilai score masing-masing kelas pada data <i>test</i> .....	66
Lampiran 2 Hasil deteksi serangan hama kakao .....	68
Lampiran 3 Hasil <i>training</i> dan <i>testing</i> .....	73

## KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya sehingga skripsi yang berjudul “**Identifikasi Jenis Serangan Hama Pada Buah Kakao Menggunakan *Image Processing***” ini dapat terselesaikan dengan baik sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Penulis menyadari bahwa banyak kendala yang dihadapi dalam menyelesaikan skripsi ini, namun berkat dorongan, dukungan, bimbingan, serta motivasi yang diberikan sehingga penyusunan skripsi ini dapat terselesaikan dengan baik. Ucapan terima kasih serta penghargaan yang setinggi-tingginya penulis sampaikan kepada:

1. Tuhan Yang Maha Esa yang melalui berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan tugas akhir ini.
2. Kedua Orang tua penulis, Bapak Abdul Hafid, S.T dan Ibu drg. Murniati, serta saudara dan saudari penulis, Nurhafni Hiroso, S.Si, Apt, drg, dewi evandari, ahmad triadi, S.T, yang selalu bersabar serta memberikan dukungan, doa, semangat, dan fasilitas kepada penulis, .
3. Bapak Prof. Dr. Ir. Indrabayu., ST, MT, M.Bus.Sys., IPM, ASEAN. Eng. selaku pembimbing I dan Ibu Dr. Ingrid Nurtanio, M.T., selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan memberikan bimbingan dalam penyusunan skripsi ini.

4. Bapak Prof. Dr. Ir. Indrabayu., ST, MT, M.Bus.Sys., IPM, ASEAN. Eng. selaku ketua Prodi Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas ilmu, bimbingan dan arahnya selama masa perkuliahan penulis.
5. Segenap Dosen dan Staff Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah banyak membantu penulis selama masa perkuliahan.
6. Teman-teman Laboratorium *AIMP Research Group* FT-UH yang telah memberikan bantuan selama penelitian, dan diskusi mengenai penyusunan skripsi.
7. Teman-teman Prodi Teknik Informatika angkatan 2016 atas dukungan dan semangat yang telah diberikan.
8. Tuti amalia, S.T yang senantiasa mendampingi dan membantu penulis dalam menyelesaikan penelitian ini, memotivasi sekaligus tempat curhat penulis. Ayu lestari ramadhani, S.T, yang telah membantu memotivasi, dan terus mendorong penulis untuk tetap lanjut mengerjakan. Serta Andi amelia ramadanti, Cici purnamasari, S.T, Putri anggriani, S.T, Asri oktianawati, S.T, Dhinda fitri wiludjeng, S.T. Diki Siswanto S.T, Muh. Raedi radifan S.T, yang menyemangati dan menjadi teman seperjuangan penulis sepanjang penyelesaian penulisan ini.
9. Teman-teman Diksar 21 Perbakin Unhas, yang telah memberikan semangat dan motivasi kepada penulis.

10. Syahril gunawan bitu, S.H yang telah membantu dan menyemangati penulis untuk memulai penelitian ini, dan Andi sengngeng khairunnisa, S.Pi yang menyempatkan waktu menemani penulis.
11. Keluarga besar della hatta, Dr. Eng. Mukhsan Putra Hatta, ST., MT yang selalu berusaha mensupport dan memotivasi penulis, Hasan pacirinnai, S.H paman penulis yang senantiasa membantu pengambilan data penulis dan Annisa fadillah darlang, S.Ak yang selalu menemani dan menyemangati penulis agar tidak menyerah.
12. Serta berbagai pihak atas segala dukungan dan bantuannya yang tidak dapat penulis tuliskan satu persatu.

Akhir kata, penulis berharap semoga Allah SWT. membalas segala kebaikan dari semua pihak yang telah membantu penulis dalam penyusunan skripsi ini, dan semoga skripsi ini dapat memberikan hal yang bermanfaat serta menambah wawasan ilmu untuk pembaca dan juga bagi penulis sendiri.

Makassar, 12 April 2023

# BAB I PENDAHULUAN

## 1.1 Latar Belakang

Subsektor perkebunan mempunyai peranan yang cukup penting dalam kegiatan perekonomian di Indonesia. Berdasarkan data dari Badan Pusat Statistik (BPS) menunjukkan kontribusi perkebunan 97,4 persen terhadap volume ekspor sektor pertanian dan berkontribusi 96,9 persen terhadap nilai ekspor sektor pertanian (Kementerian Pertanian). Kakao (*Theobroma cacao L.*) merupakan salah satu komoditas unggulan strategis perkebunan yang memegang peranan penting dalam perekonomian Indonesia yakni sebagai penghasil devisa negara, sumber pendapatan petani, penciptaan lapangan pekerjaan, mendorong agribisnis dan agroindustri dalam negeri, pelestarian lingkungan serta pengembangan wilayah (DirJen Perkebunan, 2020).

Terlepas dari potensinya produksi kakao beberapa tahun terakhir mengalami penurunan. Menurut data International Cocoa Organization (ICCO), pada tahun 2015 produksi kakao Indonesia bisa mencapai 320.000 ton, ditahun berikutnya produksi kakao semakin menurun hingga hanya mencapai 200.000 ton di tahun 2019. Salah satu faktor utama yang menyebabkan rendahnya produksi kakao yaitu faktor serangan hama ataupun patogen yang menyerang tanaman kakao.

Tanaman yang terserang hama dapat menurunkan kualitas biji yang dihasilkan, dampak yang terparah biji dapat mengalami kerusakan. Pada tanaman kakao, kebanyakan hama menyerang pada bagian buah. Selain karena buah berperan sebagai sumber makanan, buah kakao juga dapat dijadikan tempat hama menyimpan telurnya. Buah yang terserang hama akan menampakkan gejala berupa bercak yang memiliki pola dan warna tertentu pada bagian permukaan buah. Ada beberapa jenis serangan hama yang dapat dikenali dari pola dan warnanya yaitu kepik buah kakao (*Helopeltis*), kutu putih dan tikus atau tupai.

Pada penelitian sebelumnya sudah ada penelitian yang dilakukan untuk mendeteksi buah kakao seperti penelitian oleh Yuliana Ekawati pada tahun 2019 dalam tesisnya berjudul “deteksi dan hitung otomatis buah cokelat pada pohon secara *realtime* menggunakan *deep learning*” yang berhasil mendeteksi dan menghitung buah kakao menggunakan algoritma *deep learning*, pada tahun berikutnya dilanjutkan oleh Muhammad Raedi Radifan pada 2022 dalam skripsi berjudul “deteksi dan hitung buah pada buah cokelat matang dan non-matang pada pohonnya secara *realtime*” yang berhasil

mendeteksi kematangan buah kakao, selanjutnya dari penelitian ini akan dikembangkan sistem yang dapat mengidentifikasi jenis serangan hama yang menyerang buah kakao. Penelitian ini akan menggunakan metode *deep learning* dengan arsitektur VGG16 pada algoritma CNN untuk mengidentifikasi jenis serangan hama pada buah kakao dengan menggunakan *image processing*. *Deep learning* dikenal menghasilkan akurasi yang tinggi dan dapat mengenali fitur yang lebih kompleks.

Dengan adanya sistem pengidentifikasian serangan hama berbasis *image processing* ini diharapkan akan bermanfaat bagi petani untuk mengidentifikasi gejala awal dan mengidentifikasi jenis serangan hama yang menyerang kakao, sehingga dapat dilakukan tindakan preventif sebelum serangan hama menjadi lebih parah. Sistem ini nantinya akan mengidentifikasi jenis serangan hama yang menyerang buah kakao. Selain manfaat di atas sistem ini bisa digunakan dalam pengembangan pengidentifikasian serangan hama yang lebih lanjut, seperti otomatisasi pengidentifikasian serangan hama yang dominan menyerang tanaman kakao.

## **1.2 Rumusan Masalah**

Berdasarkan latar belakang, maka rumusan masalah adalah sebagai berikut:

1. Bagaimana cara mengidentifikasi jenis serangan hama pada buah kakao dengan metode *deep learning VGG16*?
2. Bagaimana akurasi sistem dalam pengidentifikasian jenis serangan hama pada buah kakao dengan menggunakan metode *deep learning VGG16*?

## **1.3 Tujuan Penelitian**

Berdasarkan rumusan masalah, tujuan penelitian dari penelitian ini sebagai berikut:

1. Untuk mengetahui cara mengidentifikasi jenis serangan hama pada buah kakao dengan metode *deep learning VGG16*.
2. Untuk mengukur tingkat akurasi dalam pengklasifikasian jenis serangan hama pada buah kakao dengan menggunakan metode *deep learning VGG16*.



## 1.4 Manfaat Penelitian

Dengan dilakukannya penelitian ini, diharapkan dapat memberi manfaat sebagai berikut:

1. Membantu petani mengidentifikasi jenis hama yang menyerang tanaman kakao.
2. Meningkatkan peran teknologi dalam mengidentifikasi jenis serangan hama yang ada pada komoditas perkebunan.
3. Menjadi referensi untuk penelitian selanjutnya tentang peran teknologi pada sektor perkebunan khususnya perkebunan kakao.

## 1.5 Ruang Lingkup

Yang menjadi batasan masalah dari penelitian ini adalah sebagai berikut:

1. Objek penelitian difokuskan pada buah kakao.
2. Serangan hama yang akan diidentifikasi yaitu serangan hama kutu putih, *helopeltis*, tikus/tupai.
3. Data sampel di ambil dari 6 sisi yaitu sisi depan, sisi samping kiri, sisi samping kanan, sisi belakang, sisi atas, dan sisi bawah.
4. Sistem klasifikasi menggunakan metode *deep learning* CNN dengan arsitektur VGG16.
5. Sistem hanya dapat mengidentifikasian satu jenis serangan hama yang dominan pada buah kakao

## BAB II TINJAUAN PUSTAKA

### 2.1 Serangan Hama Kakao

Hama merupakan salah satu jenis organisme yang dapat merusak tanaman. Serangan dari hama pada buah kakao dapat merusak biji kakao dan menurunkan produksi kakao. Kakao yang terserang hama akan menampilkan gejala berupa bercak yang memiliki pola dan warna tertentu pada beberapa bagian buah. Berikut beberapa jenis serangan hama yang akan diteliti.

#### a. Hama Kepik Penghisap Buah (*Helopeltis*)

*Helopeltis spp.* (Hemiptera: Miridae) atau kepik penghisap buah merupakan salah satu hama utama pada tanaman kakao, dengan potensi kerugian dapat menurunkan produksi buah kakao 50-60%. Gejala buah yang terserang ditandai dengan bercak-bercak berwarna coklat kehitaman. Serangan pada buah muda dapat menyebabkan layu pentil dan rontok, atau apabila pertumbuhan buah terus berlanjut maka kulit buah akan mengeras dan retak-retak, sehingga menghambat perkembangan biji di dalamnya (Indriati et al., 2014).



Gambar 1 Buah kakao yang terserang hama *helopeltis*

b. Kutu Putih

Kutu Putih (*Pseudococcus*) merupakan Hama yang menimbulkan kerusakan secara langsung dengan mengisap cairan pada tanaman, akibatnya pada tingkat kerusakan berat dapat menyebabkan terhambatnya pertumbuhan tanaman serta menimbulkan kerontokan buah muda. Kutu putih berbentuk oval dan pada bagian punggung terdapat garis-garis yang diselubungi lapisan lilin (DirJen Perkebunan, 2021 a).



Gambar 2 Buah kakao yang terserang hama kutu putih

c. Tikus atau Tupai

Tikus atau tupai merupakan binatang pengerat yang banyak menyerang tanaman terutama pada bagian buah sebagai sumber makanan. Akibat serangan hama tersebut buah kakao menjadi berlubang. Lubang akibat serangan tikus dan tupai mudah dibedakan, yaitu bekas gigitan tikus biasanya rapi atau teratur, sedangkan bekas gigitan tupai tidak beraturan. Tikus maupun tupai biasanya menyerang pada malam hari (DirJen Perkebunan, 2021b).



Gambar 3 Buah kakao yang terserang tikus atau tupai

## 2.2 Visi Komputer

Vision secara bahasa dapat diartikan sebagai penglihatan atau proses pengamatan apa yang ada pada dunia nyata melalui panca indera penglihatan manusia. Adapun *Computer Vision* adalah suatu pembelajaran menganalisis gambar dan video untuk memperoleh hasil sebagaimana yang bisa dilakukan manusia. Pada hakikatnya, *Computer Vision* mencoba meniru cara kerja sistem visual manusia (*Human Vision*) (Umam & Negara, 2016).

*Computer Vision* menerjemahkan citra menjadi deskripsi atau informasi yang merepresentasikan citra dan bertujuan untuk mengenali objek pada suatu gambar. Jadi, masukannya berupa citra, sedangkan luarannya berupa informasi (T. Sutoyo et al., 2009). Adapun visi komputer terdiri dari tahapan-tahapan seperti perolehan citra, *preprocessing*, pengekstraksian fitur, penyimpanan objek secara asosiatif, pengaksesan suatu basis pengetahuan, dan pengenalan (B. Fisher, 2013).

## 2.3 Pengolahan Citra Digital

Pengolahan citra digital merupakan metode yang digunakan untuk mengolah citra pada komputer sehingga dapat menghasilkan gambar sesuai dengan yang dibutuhkan. Misalnya terdapat sebuah citra digital berwarna dengan ukuran 1280x720 piksel. Dengan pengolahan citra digital, gambar tersebut dapat diubah ukurannya menjadi 640x360 piksel tanpa mengurangi kualitas gambar. Citra merupakan fungsi kontinu dari intensitas cahaya dalam bidang dua dimensi. Secara matematis, fungsi ini dapat dilambangkan dengan  $f(x,y)$  dimana  $(x,y)$  merupakan koordinat pada bidang dua dimensi tersebut, dan  $f(x,y)$  merupakan intensitas cahaya pada titik  $(x,y)$ . Ukuran terkecil dalam citra digital biasa disebut *picture element* atau *pixel* (Syawaluddin, 2016).

### 2.3.1 Citra digital

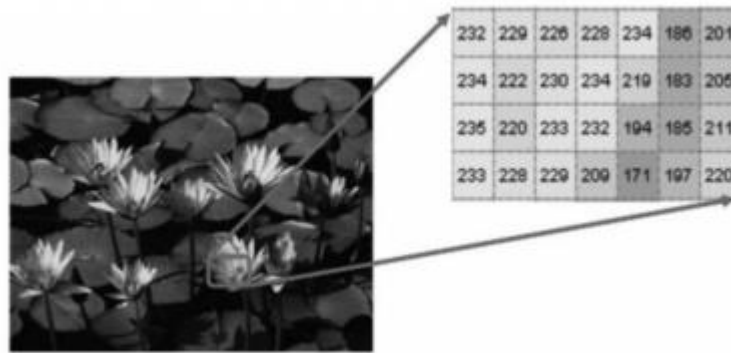
Citra adalah kombinasi antara titik, garis, bidang, dan warna yang mewakili suatu objek atau benda. Sedangkan citra digital merupakan keluaran yang dihasilkan melalui perangkat pencitraan digital seperti kamera dan dapat disimpan dan diolah komputer.

Citra perlu diterjemahkan menjadi angka-angka yang dapat dipahami oleh komputer, maka citra dibagi menjadi bagian-bagian kecil yang disebut dengan piksel. Dalam setiap piksel, perangkat pencitraan merekam sebuah angka ataupun sekumpulan angka yang merepresentasikan beberapa properti pada piksel yang dituju, seperti intensitas cahaya maupun warna, angka-angka ini diatur dalam sebuah baris-baris dan kolom-kolom array yang merujuk pada posisi vertikal dan horizontal dari piksel-piksel yang ada dalam citra. Representasi citra jika dibuat dalam bentuk matriks, dapat dilihat pada persamaan berikut: (Syawaluddin, 2016)

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & \dots & \dots & f(1,M-1) \\ \dots & \dots & \dots & \dots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix} \quad (2.1)$$

#### a. Citra *grayscale*

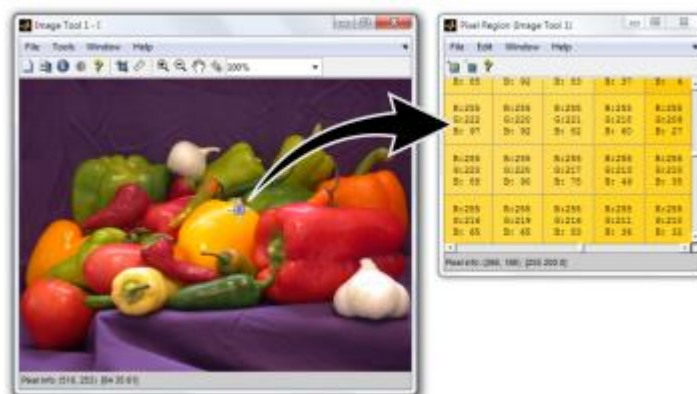
Sesuai dengan namanya, jenis citra ini memiliki gradasi warna hitam dan putih, yang menghasilkan efek warna abu-abu. Intensitas berkisar antara 0 sampai dengan 255. Nilai 0 menyatakan hitam dan 255 menyatakan putih. Citra RGB dengan matriks penyusun citra yang sebelumnya 3 matriks akan berubah menjadi 1 matriks saja. Citra *grayscale* merupakan citra yang nilai intensitas pikselnya didasarkan pada derajat keabuan. Pada citra *grayscale* 8-bit, derajat warna hitam sampai dengan putih dibagi ke dalam 256 derajat keabuan dimana warna hitam sempurna direpresentasikan dengan nilai 0 dan putih sempurna dengan nilai 255. Citra RGB dapat dikonversi menjadi citra *grayscale* sehingga dihasilkan hanya satu kanal warna (Andono et al., 2017).

Gambar 4 Citra *grayscale*

## b. Citra warna

Dalam citra warna, setiap piksel yang terdapat di dalamnya mewakili warna yang merupakan kombinasi dari tiga warna dasar yaitu merah (*red*), hijau (*green*), dan biru (*blue*). Biasanya warna-warna dasar ini disebut sebagai *channel*. Setiap warna dasar menggunakan penyimpanan  $8 \text{ bit} = 1 \text{ byte}$ , yang berarti setiap warna mempunyai gradasi 255 warna. Berarti setiap piksel mempunyai kombinasi warna sebanyak  $2^8 \times 2^8 \times 2^8 = 2^{24} = 16 \text{ juta}$  warna lebih. Itu sebabnya format ini dinamakan *true color* karena mempunyai jumlah warna yang cukup besar sehingga bisa dikatakan hampir mencakup semua warna di alam.

Penyimpanan *true color* di dalam memori berbeda dengan citra *grayscale*. Setiap piksel dari citra *grayscale* yang terdiri atas 256 gradasi warna hanya diwakili oleh 1 *byte*. Sedangkan 1 piksel citra *true color* diwakili oleh 3 *byte*, di mana masing-masing *byte* merepresentasikan tiap R, G, dan B (Andono et al., 2017).



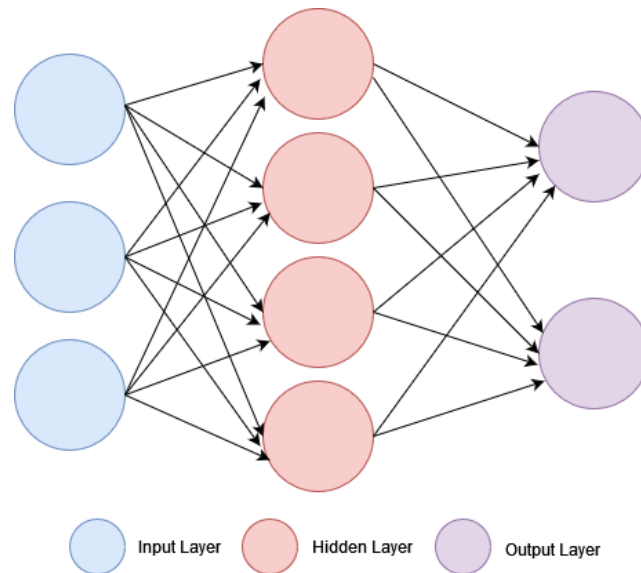
Gambar 5 Citra warna

## 2.4 Artificial Neural Network

*Artificial Neural Network* (ANN) atau Jaringan saraf tiruan adalah model pembelajaran yang terinspirasi dari sistem biologis pembelajaran dan klasifikasi. Secara sederhana, ANN bekerja pada banyak lapisan yang terhubung melalui neuron.

- **Input Layer:** Dimana data diberikan sebagai *input*.
- **Hidden Layer:** Data *input* kemudian melewati neuron berdasarkan fungsi aktivasi.
- **Output Layer:** *Output* dihasilkan di sini.

Secara visual kita dapat menggambarkan ANN seperti pada gambar



Gambar 6 Artificial Neural Network(ANN)

Jika kita membahas mengenai ANN, terdapat dua fungsi penting yaitu *basis function* dan *activation function*. *Basis function* adalah fungsi yang memproses *input* yang masuk ke *hidden layer*. Maka untuk *shallow ANN*, *output* dihasilkan dengan fungsi berikut:

$$y(x, w) = f\left(\sum_{j=1}^M w_j \Phi_j(x)\right) \quad (2.2)$$

Di sini,  $f$  diganti dengan fungsi aktivasi nonlinier, dan  $\Phi_j(x)$  adalah fungsi basis yang disesuaikan dengan koefisien  $w_j$  yang diberikan melalui vector ( $w$ ). Fungsi aktivasi adalah fungsi yang mengaktifkan neuron untuk melanjutkan *input* ke layer selanjutnya. Secara umum, fungsi aktivasi ini dapat berupa fungsi *non-linear sigmoidal*.

Adapun fungsi aktivasi secara umum yaitu *logistic sigmoid* dengan persamaan :

$$\sigma(a) = \frac{1}{1+e^{-a}} \quad (2.3)$$

Adapun untuk masalah *multi-class*, yang digunakan adalah fungsi *softmax activation* dengan persamaan :

$$\text{softmax}(x)_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (2.4)$$

Adapun fungsi aktivasi lain seperti *hyperbolic tan function*:

$$f(x) = \tanh(x) = \frac{2}{1+e^{-a}} - 1 \quad (2.5)$$

*Rectified Linear Units* atau ReLU:

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \quad (2.6)$$

*Exponential Linear Units* atau ELU:

$$f(x) = \begin{cases} a(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \quad (2.7)$$

## 2.5 Deep Learning

*Deep Learning* adalah salah satu cabang dari ilmu pembelajaran mesin (*Machine Learning*) yang terdiri algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linier yang ditata berlapis-lapis dan mendalam. *Deep Learning* disebut sebagai *Deep* (dalam) karena struktur dan jumlah jaringan saraf pada algoritmanya sangat banyak bisa mencapai hingga ratusan lapisan. *Deep Learning* adalah salah satu jenis algoritma jaringan saraf tiruan yang menggunakan metadata sebagai input dan mengolahnya menggunakan sejumlah lapisan tersembunyi (*hidden layer*) transformasi non linier dari data masukan untuk menghitung nilai *output*. Algoritma pada *Deep Learning* memiliki fitur yang unik yaitu sebuah fitur yang mampu mengekstraksi secara otomatis. Hal ini berarti algoritma yang dimilikinya secara otomatis dapat menangkap fitur yang relevan sebagai keperluan dalam pemecahan suatu masalah. Algoritma semacam ini sangat penting dalam sebuah kecerdasan buatan karena mampu mengurangi beban pemrograman dalam memilih fitur yang eksplisit. Algoritma ini dapat digunakan untuk memecahkan permasalahan yang perlu pengawasan (*supervised*), tanpa pengawasan (*unsupervised*), dan semi terawasi (*semi supervised*). Teknik dan algoritma *deep learning* telah digunakan dalam beberapa bidang seperti klasifikasi gambar, klasifikasi video, deteksi objek, pengenalan pola, *text-to-speech*, *natural language processing*, robotika, klasifikasi teks, dan lain sebagainya.



Jaringan saraf yang dimiliki oleh *Deep Learning* terbentuk dari hirarki sederhana dengan beberapa lapisan hingga tingkat tinggi atau banyak lapisan (*multi layer*). Berdasarkan hal itulah *Deep Learning* dapat digunakan untuk memecahkan masalah kompleks yang lebih rumit dan terdiri dari sejumlah besar lapisan transformasi non-linier. *Deep Learning* bekerja berdasarkan jaringan dan prosedural optimal yang digunakan pada arsitektur. Setiap *output* dari lapisan per lapisan yang tersembunyi dapat dipantau dengan menggunakan grafik khusus yang dirancang untuk setiap *output* neuron. Kombinasi dan rekombinasi dari setiap neuron yang saling terhubung dari semua unit lapisan tersembunyi dilakukan menggunakan gabungan dari fungsi aktivasi. Prosedur-prosedur tersebut dikenal sebagai Transformasi Non-Linier yang digunakan untuk prosedur optimal untuk menghasilkan bobot optimal pada setiap unit lapisan guna mendapatkan nilai target yang dibutuhkan.

Dalam proses perancangan, apabila jumlah saraf yang ditambahkan sangat banyak, hal tersebut tidak akan pernah cocok untuk menyelesaikan setiap masalah. Persoalan terpenting dalam *Deep Learning* adalah jaringan sarafnya dilatih dengan cara penurunan gradien secara sederhana. Pada saat kita menambahkan lapisan jaringan yang semakin banyak, maka sebaliknya penurunan dari gradien semakin berkurang sehingga dapat mempengaruhi nilai *output* yang dihasilkan. Jaringan Saraf Tiruan (*Artificial Neural Network*) adalah jaringan saraf yang biasanya menggunakan jaringan seperti umpan maju (*feed forward*) atau *recurrent network* yang hanya memiliki 1 atau 2 lapisan tersembunyi. Tetapi jika lapisan jaringan sarafnya lebih dari 2 *layer* atau bahkan mencapai ratusan lapisan itulah yang disebut sebagai *Deep Learning*. Pada Jaringan Saraf Tiruan, arsitektur jaringan yang dimilikinya kurang kompleks dan membutuhkan lebih banyak informasi tentang data input sehingga dapat menentukan algoritma mana yang dapat digunakan.

Dalam Jaringan Saraf Tiruan terdiri dari beberapa algoritma yaitu *Model Hebb*, *Perceptron*, *Adaline*, *Forward Propagation*, dll. Sedangkan pada algoritma jaringan saraf *Deep Learning* tidak memerlukan informasi apapun terhadap data yang akan dipelajarinya, dan algoritmanya dapat secara mandiri melakukan *tuning* (penyetelan) dan pemilihan model yang paling optimal (Dadang, 2018).



Tahap pertama dalam arsitektur CNN adalah tahap konvolusi. Kemudian dilanjutkan fungsi aktivasi menggunakan fungsi aktivasi ReLu (*Rectifier Linear Unit*), selanjutnya dilanjutkan dengan proses *pooling*. Proses ini diulang terus menerus sampai didapatkan *feature map* yang cukup untuk dilanjutkan ke *fully connected neural network*, sehingga dapat dihasilkan *output class*. Penjelasan secara detail dari tahap arsitektur CNN tersebut dijelaskan sebagai berikut:

### 2.6.1 Convolutional Layer

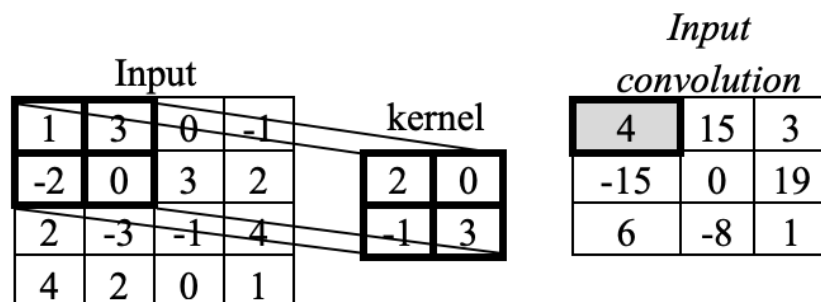
*Convolution layer* merupakan proses utama yang mendasari jaringan arsitektur CNN dan terdiri atas kernel. Kernel-kernel pada lapisan ini disebut filter konvolusi. Kernel berfungsi mempelajari fitur-fitur lokal pada *feature map*. Tahap *convolutional layer* melakukan operasi konvolusi pada *output* dari *layer* sebelumnya. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Persamaan konvolusi merupakan persamaan pada dua fungsi bernilai riil. Operasi konvolusi  $s(t)$  dapat ditunjukkan pada persamaan berikut:

$$s(t) = \sum_a I(a) \cdot K(t - a) \quad (2.8)$$

Dimana  $I(a)$  adalah input dan  $K(a)$  adalah kernel. *Input convolution layer* merupakan gambar yang direpresentasikan menjadi sebuah matriks. Operasi konvolusi menghasilkan nilai tinggi dan rendah pada posisi tertentu pada *feature map*. Posisi tertentu dari konvolusi kernel, merupakan perkalian untuk setiap nilai pada sel kernel dan nilai piksel gambar yang tumpang tindih dengan sel kernel (Khan et al., 2018). Adapun operasi perkaliannya menggunakan persamaan sebagai berikut:

$$h_{i,j} = \sum_{k=1}^m \sum_{l=1}^m W_{k,l} X_{i+k-1,j+l-1} \quad (2.9)$$

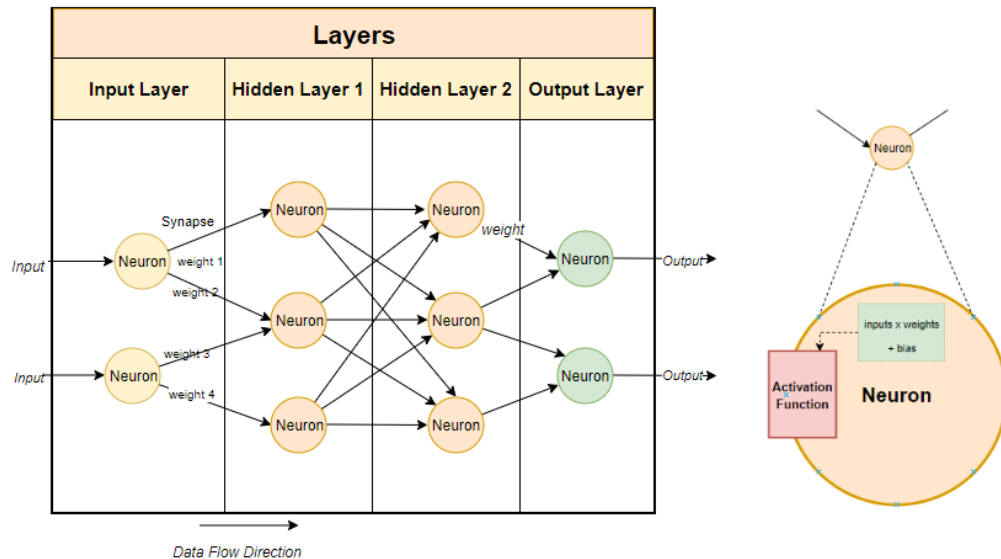
Dimana  $m$  adalah lebar dan tinggi kernel,  $h$  adalah hasil konvolusi atau input konvolusi selanjutnya,  $x$  adalah input matriks, dan  $w$  adalah kernel konvolusi. Sebagai contoh, ilustrasi pada proses *convolution layer* dengan nilai stride 2 dapat dilihat pada gambar 9.



Gambar 9 Proses *convolution* (Khan et al., 2018).

### 2.6.2 Weights dan Bias

Ketika *input* ditransmisikan antar neuron, *weights* diterapkan dan diteruskan ke fungsi aktivasi bersama dengan bias. Untuk lebih jelasnya perhatikan ilustrasi *weights* dan bias pada gambar 10.



Gambar 10 Ilustrasi *weights* dan bias pada *layer*

<https://medium.com/fintechexplained/neural-networks-bias-and-weights-10b53e6285da>

*Weights* mengontrol sinyal (atau kekuatan koneksi) antara dua neuron. Dengan kata lain, bobot menentukan seberapa besar pengaruh input terhadap *output*. Saat jaringan neural dilatih pada set pelatihan, jaringan akan di inialisasi dengan satu set *weights*. *Weights* kemudian dioptimasi selama periode pelatihan untuk menghasilkan *weights* yang optimal.

Bias adalah nilai konstanta yang ditambahkan ke layer input dengan *weights*. Bias digunakan untuk mengimbangi hasil dan untuk menggeser hasil fungsi aktivasi ke nilai positif atau negatif.

Dalam proses perhitungannya, neuron akan menghitung *weighted sum* dari input dengan persamaan:

$$Y = \sum(\text{weight} * \text{input}) + \text{bias} \quad (2.10)$$

Dimana input merupakan :

$$x_1, x_2, \dots, x_n \quad (2.11)$$

Dan *weight* adalah :

$$w_1, w_2, \dots, w_n \quad (2.12)$$

Maka *weighted sum* dapat dihitung dengan persamaan :

$$x_1 w_1 + x_2 w_2 + \dots + x_n w_n + bias \quad (2.13)$$

### 2.6.3 Backpropagation

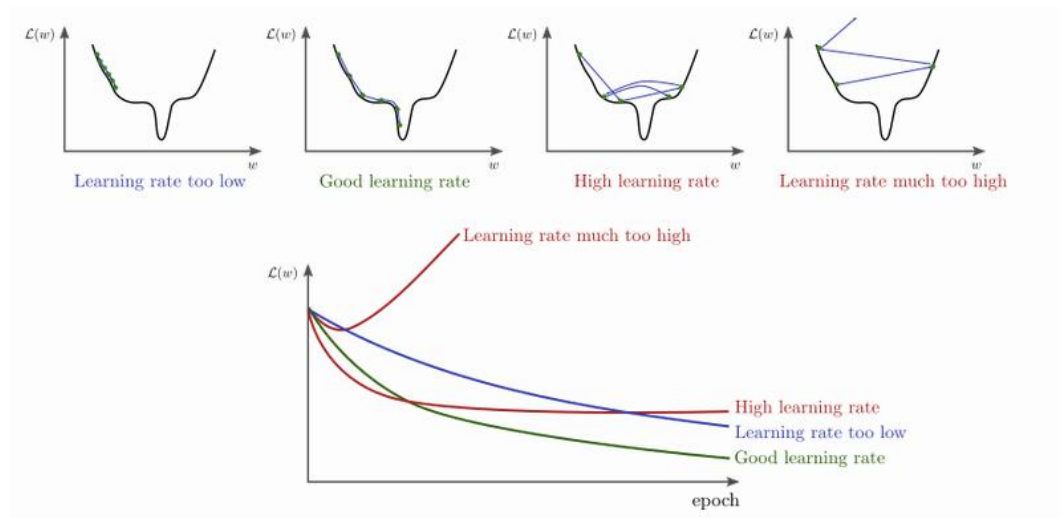
*Backpropagation* merupakan salah satu metode pada jaringan syaraf tiruan yang secara struktural biasanya terbagi menjadi tiga bagian, yaitu lapisan masukan (*input layer*), lapisan tengah (*hidden layer*), dan lapisan keluaran (*output layer*) dan antara tiap layer terhubung melalui bobot (*weight*). Nilai masukan pada *input layer* menggunakan parameter suatu objek (parameter ini dapat berupa warna, luas, berat dan sebagainya). Nilai *weight* yang tepat didapatkan melalui proses *training* berdasarkan data yang sudah ada dan nilai *output* didapatkan setelah nilai *input* dikalkulasikan dengan nilai *weight*.

*Backpropagation* memiliki metode pembelajaran *supervised learning* dan lapisan masukan akan menerima pola masukan dan melakukan proses komputasi berdasarkan bobot awal yang diperoleh secara acak (random). Jika keluaran dari jaringan berbeda dengan target yang diharapkan maka jaringan melakukan penyesuaian terhadap bobot yang ada. Proses itu akan terus berlanjut hingga keluaran dari jaringan dan target yang diharapkan menjadi sama. Proses pembelajaran membutuhkan waktu lama hingga mencapai nilai tersebut. Oleh karena itu, proses pembelajaran dibatasi dan akan berhenti jika perbedaan *output* dan target sudah mencapai nilai yang lebih kecil dari nilai toleransi (*error rate*). Besarnya penyesuaian bobot pada setiap siklus pembelajaran ditentukan oleh parameter yang disebut *learning rate* (Santika et al., 2007).

### 2.6.4 Learning rate

*Learning rate* merupakan sebuah parameter yang menghitung seberapa besar bobot (*weight*) dari model yang di update pada tiap iterasi dari algoritma optimasi (Retnowardhani & Ramdani, 2019). Besar kecilnya *learning rate* akan memengaruhi hasil dari *training* maka perlu penentuan *learning rate* yang tepat. Apabila *learning rate* yang di tetapkan terlalu tinggi waktu untuk mencapai target akan semakin cepat tetapi akurasi akan berkurang di akibatkan algoritma optimasi dapat melewati titik optimal dan menyebabkan mode divergen sehingga gagal mencapai target. Tetapi *learning rate* yang

terlalu kecil maka akan membutuhkan waktu yang lebih lama untuk mencapai nilai target yang optimum. Hal ini disebabkan pada setiap iterasi hanya mengubah sedikit nilai bobot dan bias pada model (Goodfellow et al., 2016). Berikut ilustrasi proses yang terjadi jika *learning rate* yang digunakan kurang tepat.



Gambar 11 Ilustrasi penggunaan *learning rate*

<http://www.bdhammel.com/learning-rates/>

### 2.6.5 Fungsi Aktivasi

Fungsi aktivasi merupakan operasi matematik yang dikenakan pada sinyal *output*  $y$ . Fungsi aktivasi berfungsi menentukan apakah suatu neuron aktif atau tidak berdasarkan *weighter sum* dari suatu input. Beberapa jenis fungsi aktivasi yang sering digunakan pada *deep learning* adalah *sigmoid*, *Tanh*, *algebraic sigmoid*, *ReLU*, *noisy ReLU*, *Leakly ReLU/PReLU*, *Randomized Leakly ReLU*, dan *Eksponential Linear Unit*. Fungsi aktivasi yang digunakan bergantung pada jenis data dan *output* yang ingin dihasilkan. Penelitian ini menggunakan dua fungsi aktivasi yaitu fungsi aktivasi ReLU dan fungsi aktivasi *softmax*. Fungsi aktivasi ReLU akan digunakan pada *hidden layer* dan fungsi aktivasi *softmax* akan digunakan pada layer *output*. Penjelasan dari kedua fungsi tersebut adalah sebagai berikut:

#### a. ReLU

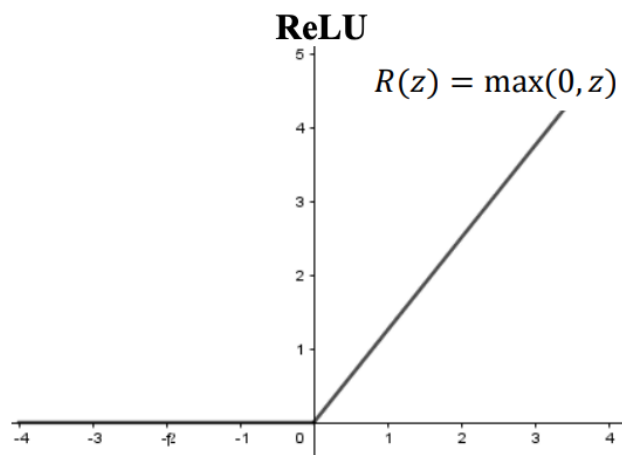
Fungsi aktivasi *Rectifier Linear Unit*(ReLU) merupakan fungsi aktivasi sederhana yang memiliki kepentingan praktis khusus karena perhitungannya yang cepat. Kelebihan fungsi aktivasi ReLU dibandingkan dengan fungsi aktivasi lain adalah sebagai berikut.

1. Fungsi aktivasi ReLU merupakan fungsi aktivasi default ketika mengembangkan *multilayer perceptron* dan *convolutional neural network*.
2. Fungsi aktivasi ReLU mengatasi masalah *gradient descent* yang hilang, yang memungkinkan model belajar lebih cepat dan berkinerja lebih baik.
3. Menemukan cara melatih jaringan dengan lebih cepat, sehingga mengurangi kemungkinan terjadinya *overfitting*.

Fungsi aktivasi ReLU memetakan input ke 0 jika negatif dan mempertahankan nilainya jika positif. Representasi fungsi ReLU adalah sebagai berikut:

$$f_{ReLU}(x) = \max(0, x) \quad (2.14)$$

Adapun grafiknya dapat dilihat pada gambar 11.



Gambar 12 Grafik Fungsi Aktivasi ReLU (Khan et al., 2018).

#### b. Fungsi Aktivasi *Softmax*

Fungsi aktivasi *softmax* merupakan fungsi input vektor dari bilangan real  $K$ , yang kemudian dinormalkan menjadi distribusi probabilitas yang terdiri atas probabilitas  $K$  yang proporsional ke eksponensial input. Komponen vektor pada *softmax* memiliki interval  $(0,1)$ . Fungsi *softmax* merupakan lapisan yang menghubungkan antara *fully connected layer* dengan *dense connection*. *Softmax* berfungsi untuk menghitung probabilitas pada setiap kelas target yang memungkinkan dan akan membantu menentukan kelas target pada input yang diberikan. Nilai *softmax* berada pada rentang probabilitas *output* dari 0 hingga 1 dan

jumlah semua probabilitas sama dengan satu. Definisi fungsi *softmax* dituliskan sebagai persamaan berikut:

$$P(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (2.15)$$

dimana  $i = 1, \dots, K$  dan  $z = (z_1, \dots, z_K) \in \mathbb{R}^K$ . Adapun contoh ilustrasi fungsi aktivasi *softmax* dapat dilihat pada gambar 13 sebagai berikut.

<i>Logit Score</i>	<i>Softmax</i>	<i>Probabilitas</i>
$z = \begin{cases} 2.0 \\ 1.0 \\ 0.1 \end{cases}$	$\longrightarrow P(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$	$\longrightarrow \begin{matrix} 0.7 \\ 0.2 \\ 0.1 \end{matrix}$

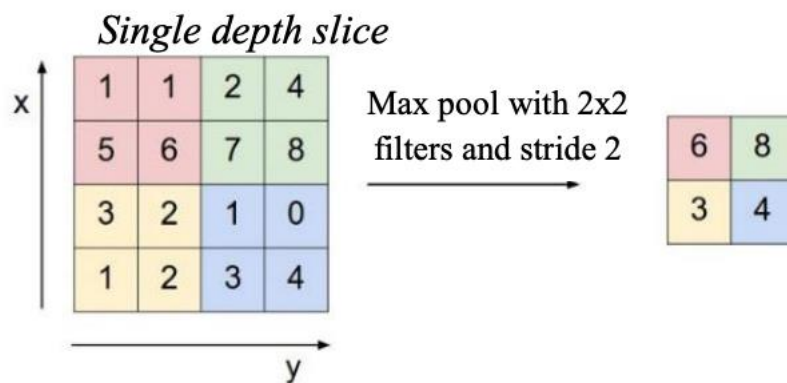
Gambar 13 proses perhitungan menggunakan fungsi aktivasi *softmax*

*Logits score* menunjukkan lapisan neuron terakhir sebagai *output* mentah pada lapisan terakhir neural network sebelum proses aktivasi berlangsung. Setelah *output* diproses dengan *softmax* akan menghasilkan nilai probabilitas dengan jumlah 1 (Khan et al., 2018).

#### 2.6.6 Pooling Layer

*Pooling* merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling*. *Pooling Layer* biasanya berada setelah *conv layer*. Pada dasarnya *pooling layer* terdiri dari sebuah filter dengan ukuran dan *stride* tertentu yang akan secara bergantian bergeser pada seluruh area *feature map*. Dalam *pooling layer* terdapat dua macam *pooling* yang biasa digunakan yaitu *average pooling* dan *max-pooling*. Nilai yang diambil pada *average pooling* adalah nilai rata-rata, sedangkan pada *max-pooling* adalah nilai maksimal. Lapisan *pooling* yang dimasukkan di antara lapisan konvolusi secara berturut-turut dalam arsitektur CNN dapat secara progresif mengurangi ukuran volume *output* pada *feature map*, sehingga mengurangi jumlah parameter pada jaringan untuk mengendalikan *overfitting*. Lapisan *pooling* bekerja pada setiap tumpukan *feature map* dan melakukan pengurangan pada ukurannya. Bentuk lapisan *pooling* umumnya dengan menggunakan filter dengan ukuran 2x2 piksel yang diaplikasikan dengan langkah (*stride*) sebanyak dua dan beroperasi pada setiap irisan dari inputnya. Berikut ini adalah contoh gambar operasi *max-pooling*:



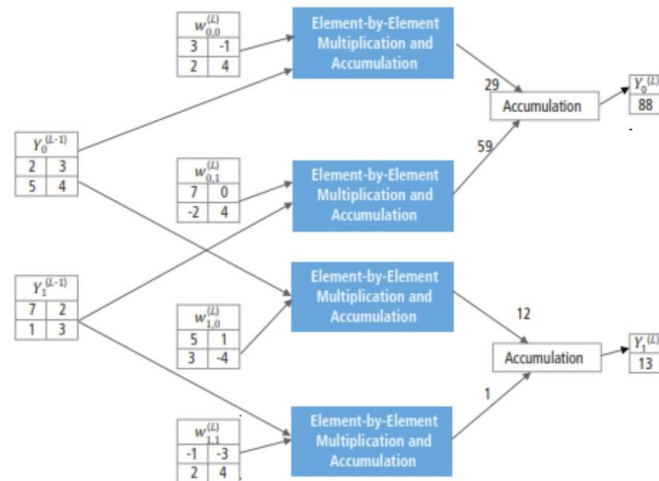


Gambar 14 Operasi *max pooling* (Khan et al., 2018)

*Output* dari proses *pooling* adalah sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan citra awal. Lapisan *pooling* akan beroperasi pada setiap irisan kedalaman volume input secara bergantian. Operasi *max-pooling* pada gambar di atas menggunakan ukuran filter 2x2. Masukan pada proses tersebut berukuran 4x4. Dari masing-masing 4 angka pada *input* operasi tersebut diambil nilai maksimalnya kemudian dilanjutkan membuat ukuran *output* baru menjadi ukuran 2x2.

#### 2.6.7 Fully Connected Layer

*Fully connected layer* merupakan sebuah lapisan dimana semua neuron aktivasi dari lapisan sebelumnya terhubung dengan neuron lapisan selanjutnya. Pada dasarnya lapisan ini biasanya digunakan pada MLP (*Multi Layer Perceptron*) yang mempunyai tujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear. Perbedaan *fully connected layer* dengan konvolusi biasa adalah neuron pada lapisan konvolusi terhubung hanya ke daerah tertentu sedangkan *fully connected* memiliki neuron yang semuanya terhubung (Khan et al., 2018). Namun keduanya masih menggunakan operasi dot, sehingga fungsinya tidak begitu berbeda. Berikut adalah proses fully-connected:



Gambar 15 Proses *fully connected layer*

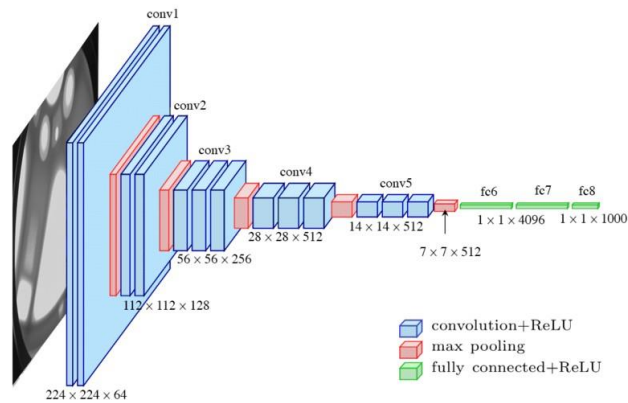
## 2.7 Transfer Learning

*Transfer learning* adalah suatu teknik atau metode yang memanfaatkan model yang sudah dilatih terhadap suatu *dataset* untuk menyelesaikan permasalahan lain yang serupa dengan cara menggunakannya sebagai *starting point*, memodifikasi dan meng-*update* parameternya sehingga sesuai dengan dataset yang baru (Sena, 2018).

## 2.8 VGG16

VGG adalah singkatan dari *Visual Geometric Group* dari *Oxford University* dan VGG16 adalah jaringan dengan 16 lapisan yang diusulkan oleh *Visual Geometric Group*. 16 lapisan ini berisi parameter yang dapat dilatih dan ada lapisan lain yang juga seperti *max pool layer*, tetapi lapisan tersebut tidak berisi parameter yang dapat dilatih. Arsitektur ini merupakan *runner up* pertama dari *Visual Recognition Challenge 2014* yaitu *ILSVRC-2014* dan dikembangkan oleh *Simonyan* dan *Zisserman*.

Kelompok riset VGG merilis serangkaian model jaringan konvolusi mulai dari VGG11 hingga VGG19. Variasi VGG yang berbeda persis sama di tiga lapisan terakhir yang terhubung sepenuhnya. Struktur keseluruhan mencakup 5 set lapisan konvolusional, diikuti oleh *max pool*. Tetapi perbedaannya adalah bahwa saat kedalaman meningkat yaitu saat berpindah dari VGG11 ke VGG19, semakin banyak lapisan konvolusional bertingkat ditambahkan dalam lima set lapisan konvolusional (Khandelwal, 2020).



Gambar 16 Arsitektur VGG16 (Loukadakis dkk, 2018)

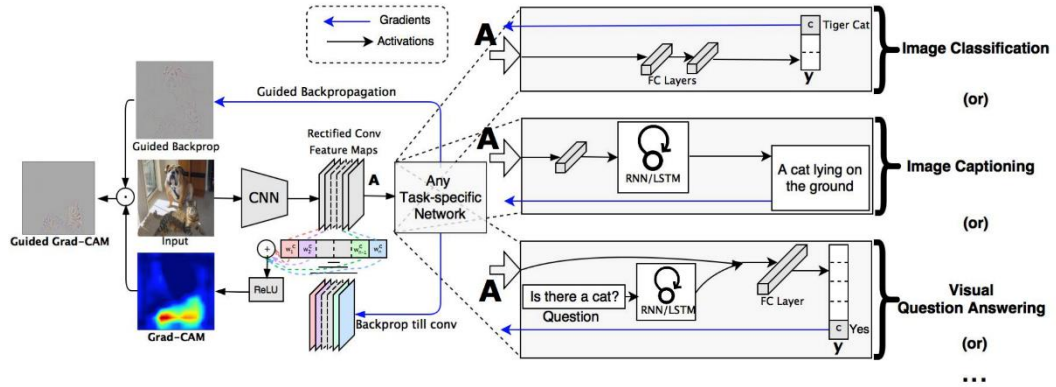
Pada Gambar 16 di atas, semua persegi panjang biru mewakili lapisan konvolusi bersama dengan fungsi aktivasi non-linier yang merupakan *rectified linear unit* (*ReLU*). Seperti yang terlihat pada gambar bahwa terdapat 13 buah persegi panjang biru dan 5 buah merah yaitu 13 buah *convolution layer* dan 5 buah *max-pooling layer*. Pada bagian akhir, ada 3 persegi panjang hijau yang mewakili 3 lapisan yang terhubung sepenuhnya. Jadi, jumlah total lapisan yang memiliki parameter adalah 16 di antaranya 13 untuk lapisan konvolusi dan 3 untuk lapisan *fully-connected*, sehingga nama tersebut diberikan sebagai VGG16. Pada *output*, arsitektur ini memiliki lapisan *softmax* yang memiliki 1000 output per kategori gambar dalam dataset *imagenet*.

## 2.9 Cross Entropy Lost Function

*Loss Function* merupakan salah satu komponen penting dalam *neural network*. *Loss* menggambarkan kemungkinan kesalahan yang dihasilkan oleh model. *Loss Function* yang baik adalah fungsi yang diharapkan menghasilkan *error* yang paling rendah. Ketika suatu model memiliki kelas yang cukup banyak, perlu adanya cara untuk mengukur perbedaan antara probabilitas hasil hipotesis dan probabilitas kebenaran yang asli. *Categorical Cross Entropy* merupakan salah satu pilihan terbaik untuk menghitung nilai *loss* pada permasalahan *multi-class classification*. *Categorical Cross Entropy* (CE) biasa juga disebut *Softmax Loss* yang merupakan gabungan dari *softmax activation* dan *cross-entropy loss*. Berikut merupakan rumus *Categorical Cross Entropy* (Gomez, 2018).

$$CE = -\sum_i^C t_i \log(s_i) \quad (2.16)$$

## 2.10 Grad-CAM



Gambar 17 Visualisasi *Grad-CAM* (Selvaraju et al., 2019)

*Grad-CAM* menggunakan informasi gradien yang berada pada lapisan konvolusional terakhir CNN untuk menetapkan nilai penting ke seluruh neuron untuk setiap keputusan tertentu. Meskipun teknik ini cukup umum karena dapat digunakan untuk menjelaskan *layer* aktivasi manapun dari dalam *network*, *Grad-CAM*, berfokus pada *layer output* saja. Seperti yang ditunjukkan pada gambar 17 untuk mendapatkan kelas diskriminatif peta lokalisasi *Grad-CAM*  $L_{\text{Grad-CAM}}^c \in \mathbb{R}^{u \times v}$  dari lebar  $u$  dan tinggi  $v$  untuk setiap kelas  $c$ , pertama kita menghitung gradien skor untuk kelas  $c$ ,  $y^c$  (sebelum *softmax*), sehubungan dengan aktivasi *feature map*  $A^k$  dari *layer* konvolusi seperti  $\frac{\partial y^c}{\partial A^k}$ . Gradien yang mengalir kembali ini adalah *global average pooled* pada dimensi lebar dan tinggi (masing-masing di indeks oleh  $i$  dan  $j$ ) untuk mendapatkan bobot neuron  $\alpha_k^c$  yang dapat dituliskan pada persamaan berikut:

$$\alpha_k^c = \underbrace{\frac{1}{Z} \sum_i \sum_j}_{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}} \quad (2.17)$$

Selama proses komputasi  $\alpha_k^c$ , ketika gradien *backpropagating* sehubungan dengan aktivasi, jumlah komputasi yang tepat untuk produk maktriks berturut-turut dari matriks bobot dan gradien sehubungan dengan fungsi aktivasi hingga akhir konvolusi dimana gradien disebarkan. Oleh karena itu, bobot  $\alpha_k^c$  ini merepresentasikan linierisasi parsial *deep network* di hilir dari  $A$ , dan menangkap 'pentingnya' *feature map*  $k$  untuk kelas target  $c$ . *Grad-CAM* melakukan kombinasi *forward activation map*, dan dilanjutkan dengan ReLU untuk mendapatkan :

$$L_{\text{Grad-CAM}}^c = \text{ReLU} \left( \underbrace{\left( \sum_k \alpha_k^c A^k \right)}_{\text{linear combination}} \right) \quad (2.18)$$

ReLU diterapkan ke kombinasi linier *maps* karena Grad-CAM hanya tertarik pada fitur yang memiliki pengaruh positif pada *class of interest*, misalnya piksel yang intensitasnya harus ditingkatkan untuk meningkatkan  $y^c$ . Piksel negatif cenderung termasuk kategori lain dalam gambar. Seperti yang diduga, tanpa ReLU, *localization maps* terkadang menyorot lebih dari sekadar kelas yang diinginkan dan berperforma lebih buruk dalam *localization*. Secara umum,  $y^c$  tidak perlu berupa skor kelas yang dihasilkan oleh klasifikasi gambar CNN. (Selvaraju et al., 2019).

### 2.11 Precision, Recall, Accuracy, Error

Ada beberapa jenis metode yang digunakan untuk mengukur kinerja sistem diantaranya *precision*, *recall*, akurasi, dan error. Akurasi merupakan perbandingan antara data yang terklasifikasi benar dengan jumlah keseluruhan data. Nilai presisi atau *precision* merupakan jumlah data kategori positif dibagi dengan total data yang diklasifikasi positif. Recall menunjukkan berapa persen data kategori positif yang terklasifikasi dengan benar oleh sistem. Nilai *error* merupakan perbandingan antara kesalahan klasifikasi data dengan jumlah keseluruhan data (Satria et al., 2020).

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.19)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2.20)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (2.21)$$

$$\text{Error} = \frac{FP+FN}{TP+TN+FP+FN} \quad (2.22)$$

atau

$$\text{Error} = 1 - \text{Accuracy} \quad (2.23)$$