**IDENTIFIKASI TINGKAT KEMATANGAN BUAH KELAPA SAWIT BERDASARKAN *IMAGE***

***FEATURE* MENGGUNAKAN METODE**

***CONVOLUTIONAL* *NEURAL***

***NETWORK***

SKRIPSI



**MUKSIN ALFALAH**

**F1E117008**

**PROGRAM STUDI SISTEM INFORMASI**

**JURUSAN TEKNIK ELEKTRO DAN INFORMATIKA**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS JAMBI**

**2021**

# SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi ini benar-benar karya sendiri. Sepanjang pengetahuan saya tidak terdapat karya atau pendapat yang ditulis atau diterbitkan orang lain kecuali sebagai acuan atau kutipan dengan mengikuti tata penulisan karya ilmiah yang telah lazim.

Tanda tangan yang tertera dalam halaman pengesahan adalah asli. Jika tidak asli, saya siap menerima sanksi sesuai dengan peraturan yang berlaku.

Jambi

Yang menyatakan

MUKSIN ALFALAH

F1E117008

# RINGKASAN

Kelapa sawit merupakan salah satu tanaman perkebunan yang memiliki peran penting bagi perekonomian nasional, terutama sebagai penyedia lapangan kerja, sumber pendapatan dan devisa negara. Peningkatan akan kebutuhan bahan baku produk-produk minyak kelapa sawit dalam bentuk makanan maupun non makanan tidak hanya dirasakan oleh masyarakat Indonesia tetapi juga negara lain. Hal ini memicu para petani kelapa sawit atau pengelola untuk lebih memaksimalkan hasil produksi minyak kelapa sawit dalam bentuk *Crude Palm Oil* (CPO). Kendala yang sering dihadapi yaitu kurang berkualitasnya minyak kelapa sawit mentah. Berdasarkan wawancara yang dilakukan terhadap Manajer PT. Rigunas Agri Utama yang merupakan pabrik pengelola buah kelapa sawit di Kabupaten Tebo melakukan proses sortasi terhadap buah kelapa sawit yang akan diolah menjadi minyak mentah. Jumlah buah kelapa sawit yang di sortasi oleh pabrik, sekitar 5% buah kelapa sawit yang tidak siap untuk diolah selanjutnya. Proses dalam pemilihan buah kelapa sawit yang dilakukan berbagai perusahaan pengolah maupun para petani yang menanam kelapa sawit pada umumnya hanya dilakukan secara manual dan melibatkan manusia sebagai pengambil keputusan, proses pengidentifikasian seperti itu memiliki kelemahan. Oleh sebab itu, teknologi pengenalan citra dapat menjadi salah satu solusi yang bisa digunakan untuk mengatasi hal tersebut. Salah satu teknologi dalam pengenalan suatu gambar adalah menggunakan algoritma CNN Berdasarkan hasil penelitian didapatkan tingkat akurasi sebesar 98% pada proses *training* dan 76% pada proses *testing*. Kemudian penelitian ini menggunakan data baru untuk menguji model yang telah dibuat dan menghasilkan nilai akurasi yang baik dalam mengenali buah kelapa sawit. Sehingga dapat disimpulkan bahwa model dan Website yang telah mampu melakukan identifikasi buah kelapa sawit.

**Kata Kunci**: CNN, Kelapa Sawit, Identifikasi, Akurasi, Citra, Deep Learning

**IDENTIFIKASI TINGKAT KEMATANGAN BUAH KELAPA SAWIT BERDASARKAN *IMAGE***

***FEATURE* MENGGUNAKAN METODE**

***CONVOLUTIONAL* *NEURAL***

***NETWORK***

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh Gelar Sarjana pada Program Studi Sistem Informasi



**MUKSIN ALFALAH**

**F1E117008**

**PROGRAM STUDI SISTEM INFORMASI**

**JURUSAN TEKNIK ELEKTRO DAN INFORMATIKA**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS JAMBI**

**2021**

**PENGESAHAN**

Skripsi dengan Judul **IDENTIFIKASI TINGKAT KEMATANGAN BUAH KELAPA SAWIT BERDASARKAN *IMAGE FEATURE* MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK*** yang disusun oleh **MUKSIN ALFALAH, NIM: F1E117008** telah dipertahankan di depan tim penguji pada tanggal 21 Juni 2021 dan dinyatakan lulus.

Susunan Tim Penguji:

Ketua : Ir. Indra Weni, M.Kom.

Anggota : 1. Edi Saputra, S.T., M.Sc.

2. Ulfa Khaira, S.Komp., M.Kom.

Disetujui:

|  |  |
| --- | --- |
| Pembimbing Utama,  Dedy Setiawan, S.Kom.,M.IT.  NIP. 198007082005011003 | Pembimbing Pendamping,    Pradita Eko Prasetyo, S.Pd.,M.Cs.  NIP. 198710282019031010 |

Diketahui:

|  |  |
| --- | --- |
| Dekan Fakultas Sains dan Teknologi,  Prof. Drs. Damris M, M.Sc., Ph.D.  NIP. 196605191991121001 | Ketua Jurusan Teknik Elektro dan Informatika,    Nehru, S.Si., M.T.  NIP. 197602082001121002 |

# RIWAYAT HIDUP

Muksin Alfalah lahir di Rimbo Bujang, pada tanggal 26 Maret 1999. Penulis merupakan anak ketiga dari lima bersaudara dari pasangan Hardi dan Suryati. Jalur pendidikan formal yang ditempuh penulis adalah sebagai berikut:



1. SDN 170 Rimbo Bujang, 2005-2011
2. SMPN 23 Tebo, 2011-2014
3. SMKN 1 Merangin, 2014-2017

Saat ini Penulis melanjutkan pendidikannya di salah satu Perguruan Tinggi Negeri, yaitu Universitas Jambi. Penulis sedang menjalani semester 7 di program studi Sistem Informasi, Fakultas Sains dan Teknologi, Universitas Jambi. Selama menempuh pendidikan jenjang S1, penulis cukup aktif dalam bidang akademik dan non akademik.

# PRAKATA

Puji dan syukur penulis panjatkan atas berkat dan rahmat Allah SWT yang telah memberikan rahmat dan karunia-Nya, sehingga terselesaikannya skripsi yang berjudul “Identifikasi Tingkat Kematangan Buah Kelapa Sawit Berdasarkan Image Feature Menggunakan Algoritma Convolutional Neural Network”.

Penulis menyadari sangatlah sulit bagi penulis untuk menyelesaikan laporan ini tanpa bantuan dan bimbingan dari berbagai pihak, terlebih kepada kedua pembimbing saya, yaitu bapak Dedy Setiawan, S.Kom., M.IT selaku pembimbing utama saya dan bapak Pradita Eko Prasetyo, S.Pd., M.Cs. selaku pembimbing pendamping saya. Tidak lupa pula dengan segala rasa hormat penulis ucapkan terimakasih kepada:

1. Orang tua beserta keluarga besar saya atas kasih sayang, dukungan dan doanya sehingga dapat menyelesaikan Tugas Akhir ini dengan baik.
2. Seluruh dosen pengajar Program Studi Sistem Informasi Fakultas Sains dan Teknologi.
3. Keluarga besar HIMASI Unja (Himpunan Mahasiswa Sistem Informasi Unja), sebagai wadah yang membawahi mahasiswa Sistem Informasi Fakultas Sains dan Teknologi, terimakasih atas kebersamaan, kekeluargaan, kekompakan, keceriaan dan pelajaran berharga lainnya selama 4 tahun ini.
4. Teman-teman Sistem Informasi UNJA Angkatan 2017 yang bersama-sama menjadi pejuang gelar S.Kom dan Toga UNJA, terimakasih semangatnya.
5. Serta semua pihak lainya yang tidak bisa dituliskan penulis satu per satu yang telah membantu selama pembuatan Tugas Akhir ini.

Demikian Tugas Akhir ini, penulis mengucapkan terima kasih kepada semua pihak yang telah memberikan bantuan baik moril maupun materil sehingga tugas akhir ini dapat diselesaikan. Penulis menyadari bahwa tugas akhir ini masih jauh dari kata sempurna dan masih banyak kekurangan. Hal tersebut dikarenakan keterbatasan ilmu dan pengetahuan yang dimiliki penulis semata. Oleh karena itu penulis mengharapkan kritik dan saran dari pembaca untuk menyempurnakan penulisan tugas akhir ini. Semoga Tugas Akhir ini dapat memberikan manfaat bagi penulis khususnya dan umumnya bagi semua pihak yang membutuhkan. Akhir kata, semoga Allah SWT senantiasa melimpahkan rahmat serta hidayah-Nya kepada kita semua, Amin amin ya robbal ‘alamiin.

Jambi,

MUKSIN ALFALAH

F1E117008

# DAFTAR ISI

Halaman

[SURAT PERNYATAAN i](#_Toc76070140)

[RINGKASAN ii](#_Toc76070141)

[RIWAYAT HIDUP v](#_Toc76070142)

[PRAKATA vi](#_Toc76070143)

[DAFTAR ISI vii](#_Toc76070144)

[DAFTAR TABEL ix](#_Toc76070145)

[DAFTAR GAMBAR x](#_Toc76070146)

[DAFTAR LAMPIRAN xi](#_Toc76070147)

[I. PENDAHULUAN 1](#_Toc76070148)

[1.1 Latar Belakang 1](#_Toc76070149)

[1.2 Rumusan Masalah 3](#_Toc76070150)

[1.3 Batasan Masalah 4](#_Toc76070151)

[1.4 Tujuan Penelitian 4](#_Toc76070152)

[1.5 Manfaat Penelitian 4](#_Toc76070153)

[II. TINJAUAN PUSTAKA 5](#_Toc76070154)

[2.1 *Deep Learning* 5](#_Toc76070155)

[2.2 *Neural Network* 5](#_Toc76070156)

[2.3 Komponen *Neural Network* 6](#_Toc76070157)

[2.4 *Convolutional Neural Network* 7](#_Toc76070158)

[2.5 Kelapa Sawit 15](#_Toc76070159)

[2.6 Penelitian Terdahulu 17](#_Toc76070160)

[III. METODOLOGI PENELITIAN 20](#_Toc76070161)

[3.1 Tempat dan Waktu Penelitian 20](#_Toc76070162)

[3.2 Perangkat Penelitian 20](#_Toc76070163)

[3.3 Variabel Penelitian 21](#_Toc76070164)

[3.4 Metode Analisis Data 21](#_Toc76070165)

[3.5 Tahapan Penelitian 22](#_Toc76070166)

[3.6 Data Penelitian 22](#_Toc76070167)

[3.7 Preproses Data 22](#_Toc76070168)

[3.8 Rancangan *Convolutional Neural Network* (CNN) 23](#_Toc76070169)

[3.9 Rancangan Pengujian Model 25](#_Toc76070170)

[3.10 Rancangan Pengujian Gambar 25](#_Toc76070171)

[IV. HASIL DAN PEMBAHASAN 27](#_Toc76070172)

[4.1 Hasil Pengolahan Data Penelitian 27](#_Toc76070173)

[4.2 Hasil Praproses Data 28](#_Toc76070174)

[4.3 Hasil Rancangan *Convolutional Neural Network* 29](#_Toc76070175)

[4.4 Hasil Pengujian Model 36](#_Toc76070176)

[4.5 Hasil Klasifikasi Gambar 37](#_Toc76070177)

[V. KESIMPULAN DAN SARAN 41](#_Toc76070178)

[5.1 Kesimpulan 41](#_Toc76070179)

[5.2 Saran 41](#_Toc76070180)

[DAFTAR PUSTAKA 42](#_Toc76070181)

[LAMPIRAN 44](#_Toc76070182)

# DAFTAR TABEL

Tabel Halaman

**1.** Tabel Perbandingan Pustaka Metode CNN 17

**2.** Timeline Penelitian 20

**3.** Variabel Penelitian 21

**4.** Rancangan *Confusion Matriks* 25

**5.** Model CNN 30

**6.** Perhitungan *dot product* 33

**7.** *Confusion Matriks* 36

**8.** Hasil Pengujian 38

**9.** *Margin Error* Pengujian Gambar 40

# DAFTAR GAMBAR

Gambar Halaman

[**1.** Peta Sebaran Luas Areal Kelapa Sawit di Indonesia (BPS, 2020). 1](#_Toc76056085)

[**2.** Satu unit *neuron* pada JST (Kusumadewi, 2004) 6](#_Toc76056086)

[**3**. Arsitektur *Convolutional Neural Network* 8](#_Toc76056087)

[**4.** Operasi *Convolutional Layer* 8](#_Toc76056088)

[**5.** Fungsi aktivasi *ReLU* 9](#_Toc76056089)

[**6.** Fungsi aktivasi *sigmoid* 10](#_Toc76056090)

[**7.** Fungsi aktivasi *tanh* 10](#_Toc76056091)

[**8**. Operasi *Max Pooling* 11](#_Toc76056092)

[**9.** Operasi *Dropout* 11](#_Toc76056093)

[**10.** *Processing of a Fully-Connected Layer* 13](#_Toc76056094)

[**11.** Sawit Mentah 16](#_Toc76056095)

[**12.** Sawit Matang 16](#_Toc76056096)

[**13.** Sawit Busuk 17](#_Toc76056097)

[**14.** Diagram *fishbone* penelitian 22](#_Toc76056098)

[**15.** *Image Augmentation* 23](#_Toc76056099)

[**16.** Rancangan Arsitektur CNN Proses *Training* 24](#_Toc76056100)

[**17.** *Flowchart* Proses Training 25](#_Toc76056101)

[**18*.*** *Flowchat* proses *Testing*. 26](#_Toc76056102)

[**19.** Penghapusan background 27](#_Toc76056103)

[**20.** Kode pembagian Data 27](#_Toc76056104)

[**21.** Hasil proses Augmentasi 28](#_Toc76056105)

[**22.** Definisi Parameter 28](#_Toc76056106)

[**23.** Arsitektur Jaringan 29](#_Toc76056107)

[**24.** Proses *Convolution* 31](#_Toc76056108)

[**25.** Perhitungan Proses *Convolution* 32](#_Toc76056109)

[**26.** Posisi Kernel pada *Convolution* 32](#_Toc76056110)

[**27.** Proses *Pooling* 34](#_Toc76056111)

[**28.** Proses *Fully Connected* 34](#_Toc76056112)

[**29.** Grafik Hasil *Training* 35](#_Toc76056113)

[**30.** *Precision, Recall dan F1-score* 37](#_Toc76056114)

[**31.** Tampilan *Website* 38](#_Toc76056115)

# DAFTAR LAMPIRAN

Halaman

[**Lampiran 1** 44](#_Toc73295736)

[**Lampiran 2** 47](#_Toc73295737)

# I. PENDAHULUAN

## Latar Belakang

Kelapa sawit merupakan salah satu tanaman perkebunan yang memiliki peran penting bagi perekonomian nasional, terutama sebagai penyedia lapangan kerja, sumber pendapatan dan devisa negara. Tingkat perkembangan kelapa sawit saat ini sangat pesat dimana terjadi peningkatan baik luas areal maupun produksi kelapa sawit seiring dengan meningkatnya kebutuhan masyarakat. Pada Tahun 2020, pertumbuhan perkebunan kelapa sawit tercatat mencapai 14.824,60 hektar. Sedangkan untuk luas areal perkebunan kelapa sawit di Indonesia pada tahun 2020 seluas 16.381.000 hektar (Menteri Pertanian, 2020). Pada Gambar 1 merupakan peta sebaran produksi kelapa sawit di Indonesia. Perkebunan kelapa sawit juga menjadi penyedia lapangan kerja dan sumber pendapatan bagi sekitar 5,30 juta kepala keluarga petani, serta memberikan sumbangan devisa sebanyak US$ 20,2 miliar (Badan Pusat Statistik, 2020).



**Gambar 1.** Peta Sebaran Luas Areal Kelapa Sawit di Indonesia (BPS, 2020).

Kelapa sawit merupakan tanaman penghasil minyak sawit dan inti sawit. Kelapa sawit merupakan bahan baku utama dalam produksi *Crude Palm Oil* (CPO). Potensi hasil produksi *Crude Palm Oil* (CPO) di Indonesia sangat besar apabila digunakan sebagai bahan baku produk-produk minyak, baik untuk makanan maupun non makanan. Dengan terus meningkatnya kebutuhan masyarakat akan produksi yang berasal dari minyak kelapa sawit mengakibatkan kenaikan jumlah produksi kelapa sawit di Indonesia dari tahun ke tahun. Sejak tahun 1980, perkembangan produksi kelapa sawit dalam bentuk *Crude Palm Oil* (CPO) di Indonesia terus mengalami peningkatan dengan rata-rata pertumbuhan sebesar 11,48% per tahun (Direktorat Jendral Perkebunan, 2019).

Banyak hal yang mempengaruhi kualitas dari minyak kelapa sawit, diantarnya, kandugan air, kandungan kotoran, kandungan asam lemak bebas, dan salah satu yang paling penting yaitu tingkat kematangan dari buah kelapa sawit. Karena pada dasarnya tingkat kematangan dari buah kelapa sawit merupakan faktor yang sangat penting dalam menentukan kualitas minyak mentah (Rangkuti dan Syahputra, 2019). Tetapi dalam praktiknya untuk menentukan tingkat kematangan buah kelapa sawit pada pabrik-pabrik sawit terdapat banyak kendala. Salah satu kendalanya adalah tingkat kematangan dari buah kelapa sawit tersebut, sehingga perlu dilakukan sortasi untuk mendapatkan buah kelapa sawit yang berkualitas dengan tingkat kematangan yang sesuai. Berdasarkan hasil wawancara dengan Manajer PT. Rigunas Agri Utama (2020) yang mana PT. Rigunas Agri Utama merupakan salah satu pabrik pengelola buah kelapa sawit di Kabupaten Tebo, PT. Rigunas Agri Utama selalu melakukan proses sortasi terhadap buah kelapa sawit yang akan diolah menjadi minyak mentah supaya mendapat hasil minyak berkualitas. Untuk jumlah buah kelapa sawit yang di sortasi oleh pabrik, sekitar 5% buah kelapa sawit yang tidak siap untuk diolah selanjutnya.

Proses dalam pemilihan buah kelapa sawit yang dilakukan berbagai perusahaan pengolah maupun para petani yang menanam kelapa sawit pada umumnya hanya dilakukan secara manual dan melibatkan manusia sebagai pengambil keputusan. Proses pengidentifikasian seperti itu memiliki beberapa kelemahan, di antaranya adalah waktu yang diperlukan relatif lama, manusia juga cenderung mudah merasa lelah dan jenuh jika melakukan aktivitas yang monoton, adanya perbedaan persepsi tentang mutu buah, beragam hasil produk juga didapatkan karena adanya keterbatasan visual manusia, serta sangat dipengaruhi oleh kondisi psikis pengamatnya. Hal tersebut juga bisa mengakibatkan tidak konsisten dalam proses pemilihannya. Cara manual yang dilakukan terlalu banyak memakan waktu, sehingga jika diterapkan pada skala industri besar diperlukan bantuan mesin pada proses tersebut. Oleh karena itu untuk mengenali tingkat kematangan buah kelapa sawit untuk diolah menjadi minyak mentah, hingga pada akhirnya diolah oleh produsen pengelola minyak mentah sampai dengan dikonsumsi atau digunakan untuk keperluan sehari-hari, perlu adanya suatu sistem yang dapat mengenali tingkat kematangan buah kelapa sawit sesuai tiga kategori yaitu mentah, matang dan busuk, lalu untuk melakukan pemilihannya secara otomatis dengan memanfaatkan citra buah kelapa sawit.

Oleh karena itu, dalam penelitian ini diharapkan mampu memberikan kontribusi dan menjadi alternatif bagi pengelola buah kelapa sawit atau petani dalam membantu proses pemilihan atau sortasi. Dalam hal ini pengolahan citra merupakan solusi untuk mengatasi masalah tersebut. Cara ini memiliki kemampuan yang lebih peka karena dilengkapi sensor elektro-optik yang bisa dipastikan akan lebih tepat dan objektif dibandingkan dengan cara visual yang bersifat subjektif dan dipengaruhi oleh kondisi psikis pengamatnya. Dengan adanya teknologi ini, kematangan buah kelapa sawit dapat ditentukan dengan cepat, murah, dan dengan tingkat ketelitian yang dapat dipercaya.

Dari persoalan tersebut, dibutuhkan teknologi yang dapat mengidentifikasi kematangan buah kelapa sawit dengan cepat dan otomatis. Beberapa penelitian terdahulu yang telah diselesaikan, misalnya Implementasi Jaringan Syaraf Tiruan (Jst) dan Pengolahan Citra untuk Klasifikasi Kematangan Tbs Kelapa Sawit (Minarni et al., 2018) dan Klasifikasi Tingkat Kematangan Tandan Buah Segar Kelapa Sawit Menggunakan *Probe Optik* (Sari et al., 2019). Namun dari penelitian sebelumnya yang telah dilakukan, belum ada penelitian serupa yang menggunakan algoritma *Convolutional Neural Network*. Oleh sebab itu, membuat peneliti berkeinginan melakukan penelitian serupa yang menggunakan algoritmaCNN.

Metode *Deep Learning* yang saat ini memiliki hasil paling signifikan dalam pengenalan citra adalah *Convolutional Neural Network*. Karena metode *Convolutional Neural Network* berusaha meniru sistem pengenalan citra pada *visual cortex* manusia sehingga memiliki kemampuan mengolah informasi citra selayaknya manusia (Maulana & Rochmawati, 2019). Tantangan menggunakan metode CNN adalah bagaimana menghasilkan akurasi yang tinggi dengan biaya komputasi yang relatif rendah. Tentu banyak hal yang dapat mempengaruhi hal tersebut, beberapa diantaranya adalah jumlah *training* dataset, dimensi dari data, serta pemilihan *training rate* yang tepat (Sahrul et al., 2018). Cara lain yaitu dengan menggunakan arsitektur *transfer learning* yang mampu meniru dan menduplikasi dari model yang sebelumnya telah di inisiasi oleh mesin (Prathivi et al., 2019).

Berdasarkan latar belakang diatas penelitian ini menerapkan implementasi dari metode *deep learning* menggunakan CNN untuk membantu mengidentifikasi tingkat kematangan buah kelapa sawit. Penelitian ini diharapkan mampu menghasilkan akurasi yang terbaik dan juga mampu untuk mengklasifikasikan kematangan buah kelapa sawit.

## Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dipaparkan sebelumnya, maka rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana tingkat akurasi yang didapatkan dari hasil klasifikasi tingkat kematangan buah kelapa sawit menggunakan algoritma *Convolutional Neural Network*?
2. Bagaimana penggunaan metode *Deep Learning* menggunakan *Convolutional Neural Network* untuk mengidentifikasi tingkat kematangan buah kelapa sawit?

## Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Dataset terdiri dari tiga kategoriyaitu: buah kelapa sawit mentah, buah kelapa sawit matang dan buah kelapa sawit busuk*.*
2. Dataset berupa gambar buah kelapa sawit pertandan.
3. Data *input* dan data uji merupakan citra buah kelapa sawit yang bersumber dari internet dengan berbagai sumber dan foto lapangan.
4. Penelitian ini hanya sampai pada proses identifikasi pada perangkat lokal.
5. Proses pengumpulan data hanya dilakukan di Provinsi Jambi.

## Tujuan Penelitian

Tujuan penelitian ini adalah sebagai berikut:

1. Mengetahui tingkat akurasi model data yang di latih dan di uji dengan metode *Convolutional Neural Network*.
2. Penggunaan metode *Deep Learning* menggunakan *Convolutional Neural Network* untuk mengidentifikasi gambar kematangan buah kelapa sawit.

## Manfaat Penelitian

Manfaat yang didapat dari penelitian ini adalah sebagai berikut:

1. Memberikan pengetahuan mengenai penggunaan metode *Deep Learning* menggunakan *Convolutional Neural Network* untuk klasifikasi citra kematangan buah kelapa sawit.
2. Mampu mengidentifikasi tingkat kematangan buah kelapa sawit secara cepat dan tepat.
3. Membantu pengelola buah kelapa sawit atau petani dalam pemilihan buah kelapa sawit sesuai kategori yang telah di tentukan.

# II. TINJAUAN PUSTAKA

## *Deep Learning*

*Deep Learning* merupakan salah satu bidang dari *Machine Learning* yang memanfaatkan jaringan syaraf tiruan untuk implementasi permasalahan dengan dataset yang besar. Teknik *Deep Learning* memberikan arsitektur yang sangat kuat untuk *Supervised Learning*. Dengan menambahkan lebih banyak lapisan maka model pembelajaran tersebut bisa mewakili data citra berlabel dengan lebih baik. Pada *Machine Learning* terdapat teknik untuk menggunakan ekstraksi fitur dari data pelatihan dan algoritma pembelajaran khusus untuk mengklasifikasi citra maupun untuk mengenali suara. Namun, metode ini masih memiliki beberapa kekurangan baik dalam hal kecepatan dan akurasi (Deng dan Yu, 2014).

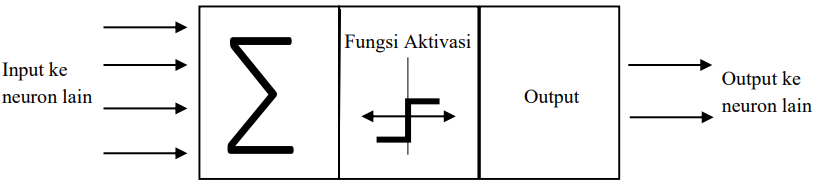
Algoritma yang digunakan pada *Feature Engineering* dapat menemukan pola umum yang penting untuk membedakan antara kelas Dalam *Deep Learning*, metode CNN atau *Convolutional Neural Network* sangatlah bagus dalam menemukan fitur yang baik pada citra ke lapisan berikutnya untuk membentuk hipotesis nonlinier yang dapat meningkatkan kekompleksitasan sebuah model. Model yang kompleks tentunya akan membutuhkan waktu pelatihan yang lama sehingga di dunia *Deep Learning* pengunaan GPU sudah sangatlah umum (Danukusumo, 2017).

## *Neural Network*

*Neural Network* adalah salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut (Saputra et al., 2017). Otak manusia terdiri dari ratusan juta sel syaraf yang disebut *neuron*. Otak digambarkan sebagai sebuah mesin yang mengubungkan *neuron* satu dengan lainnya dalam bentuk impuls saraf sehingga dapat mengkoordinasikan berbagai fungsi tubuh.

Cara kerja *neuron* tersebut ditiru oleh *Neural Network* sebagai langkah membuat machine yang pintar. Dalam hal ini, *Neural Network* tidak diprogram untuk menghasilkan keluaran tertentu. Semua keluaran yang ditarik oleh jaringan didasarkan pada pengalamannya selama mengikuti proses pembelajaran. Pada proses pembelajaran, pola-pola *input* dimasukkan ke *Neural Network* untuk dan kemudian jaringan diajari untuk menentukan jawaban yang bisa diterima. Pembelajaran dilakukan dengan menentukan bobot masing-masing node dalam jaringan (Kusumadewi, 2004).

## Komponen *Neural Network*

*Neural Network* memiliki beberapa tipe yang berbeda-beda, akan tetapi hampir semua komponen yang dimiliki sama. Seperti halnya jaringan syaraf pada otak manusia, *Neural Network* juga terdir dair beberapa *neuron* unit yang saling behubungan. Masing-masing dari *neuron* tersebut akan melakukan transformasi informasi yang diterima melalui sambungan keduanya menuju *neuron* lain. Hubungan ini biasanya disebut dengan sebutan bobot (*Weight*). Informasi tersebut disimpan pada suatu nilai tertentu pada bobot tertentu. Berikut adalah struktur *Neuron* pada *Neural Network*:

**Gambar 2.** Satu unit neuron pada JST (Kusumadewi, 2004)

Berdasarkan Gambar 2. diatas menunjukkan struktur *neuron* pada *Neural Network*. Komponen yang dimiliki struktur tersebut yaitu sebagai berikut:

1. *Input* terdiri dari variabel independet (X1, X2, X3....... Xn,) yang merupakan sebuah sinyal yang masuk ke sel syaraf.
2. Bobot (*Weigth*) terdiri dari beberapa bobot (W1, W2, W3...... W3,) yang berhubungan dengan masing-masing node.
3. *Threshod* merupakan nilai ambang batas internal dari node. Besar nilai ini mepengaruhi aktivasi dari *output* node y.
4. *Activation Function* (Fungsi Aktivasi) merupakan operasi matematika yang dikenal pada sinyal *output* y.

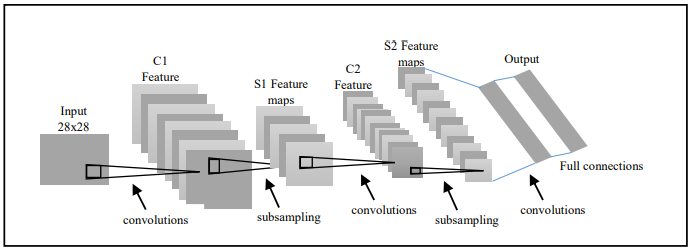
Informasi masukan bagi *neuron*, atau *input*, dikirim ke *neuron* dengan bobot kedatangan tertentu. *Input* ini diproses dengan suatu fungsi yang menjumlahkan nilai berbobot dari semua *input* yang datang. Hasil penjumlahan kemudian dikenakan fungsi aktivasi untuk menentukan apakah *neuron* tersebut akan diaktifkan atau tidak. Biasanya hal ini dilakukan dengan membandingkan dengan *threshold* atau ambang nilai tertentu. Apabila *neuron* tersebut diaktifkan, maka *neuron* tersebut mengirimkan *output* melalui bobot-bobotnya ke semua *neuron* selanjutnya yang berhubungan. Pada *neuron* layer, penempatan *neuron*-*neuron* akan dikumpulkan dalam *neuron* layer (lapisan-lapisan). Kemudian *neuron*-*neuron* pada satu lapisan akan di hubungkan dengan lapsan-lapisan sebelum dan sesudahnya, kecuali lapisan *input* dan *output*. Informasi yang di bawa dari langkah *input* awal akan dirambatkan dari lapisan ke lapisan dari lapisan *input* sampai lapisan *output*. Lapisan ini sering disebut dengan istilah hidden layer (lapisan tersembunyi). Pada umumnya setiap *neuron* terletak pada lapisan yang sama akan memiliki keadaan yang sama. Sehingga pada setiap lapisan sama, setiap *neuron* akan memiliki fungsi aktifasi yang sama. Koneksi antar lapisan dengan *neuron* harus selalu berhubungan. Faktor terpenting dalam menentukan kelakuan suatu *neuron* adalah terletak pada pola bobot dan fungsi aktivasinya (Kukreja et al., 2016).

## *Convolutional Neural Network*

*Convolutional Neural Network* atau dikenal juga dengan sebutan *ConvNets* adalah sebuah metode untuk memproses data dalam bentuk beberapa array, contohnya yaitu gambar berwarna yang terdiri dari tiga array 2D yang mengandung intensitas piksel dalam tiga jenis warna. *Convolutional Neural Network*s (*ConvNets*) merupakan penerapan dari *Artificial Neural Networks* (ANN) yang lebih istimewa dan saat ini diklaim sebagai model terbaik untuk memecahkan masalah pengenalan objek (Goodfellow et al., 2016).

Penelitian awal yang mendasari penemuan *Convolutional Neural Network* pertama kali dilakukan oleh Hubel dan Wiesel (Hubel & Wiesel, T, 1968) mengenai *virual cortex* pada indera penglihatan kucing. Secara teknis, *Convolutional Neural Network* memiliki arsitektur yang dapat dilatih dan terdiri dari beberapa tahap. Masukan dan keluaran dari masing-masing tahap adalah beberapa array yang disebut *feature map* atau peta fitur. Contohnya untuk citra *greyscale*, *input* atau masukan adalah berupa *matriks* dua dimensi. *Output* dari masing-masing tahap adalah *feature map* hasil pengolahan dari semua lokasi pada citra masukan. Masing-masing tahap terdiri dari tiga lapisan yaitu *convolution*, aktivasi dan *pooling*.

Secara umum, arsitektur dari sebuah *convolution Neural Network* ditunjukkan pada Gambar 3. *Input* dari CNN adalah berupa citra dengan ukuran tertentu. Tahap pertama dalam CNN adalah tahap *convolution*. *Convolution* dilakukan dengan menggunakan kernel dengan ukuran tertentu. Jumlah kernel yang dipakai tergantung dari jumlah fitur yang dihasilkan. *Output* dari tahap ini kemudian dikenakan fungsi aktivasi, yang bisa berupa fungsi *tanh* atau *Rectifier Linear Unit* (*ReLU*). *Output* dari fungsi aktivasi kemudian melalui proses *sampling* atau *pooling*. *Output* dari proses *pooling* adalah citra yang telah berkurang ukurannya, tergantung dari *pooling* mask yang dipakai. Proses ini diulang beberapa kali sampai didapatkan peta fitur yang cukup untuk dilanjutkan ke *fully connected neural network*, dan dari *fully connected network* adalah *output class* (LeCun et al., 2010).



**Gambar 3**. Arsitektur Convolutional Neural Network (LeCun et al., 2010)

**Convolutions Layer**

Operasi *convolution* merupakan operasi dua fungsi argumen yang bernilai nyata (Goodfellow et al., 2016). Operasi ini menerapkan fungsi keluaran (*output*) sebagai *Feature Maps* dari masukan (*input*) citra. *Input* dan *output* ini dapat dilihat sebagai dua argumen yang mempunyai nilai riil. Operasi untuk *convolution* ke dalam *input* lebih dari satu dimensi dapat ditujukan pada persamaan (3.1).

(3.1)

Pada persamaan 3.1 ditunjukkan perhitungan dasar dalam operasi *convolution* dimana i dan j adalah piksel dari citra. Perhitungannya bersifat kumulatif dan muncul saat K sebagai *kernel*, I sebagai *input* dan *kernel* yang dapat dibalik relatif terhadap *input*. Sebagai alternatif, operasi *convolution* dapat dilihat sebagai perkalian *matriks* antara citra masukan dan *kernel* yang dimana keluarannya dapat dihitung dengan *dot product*. Dalam pengolahan citra, *convolution* berarti mengaplikasikan sebuah *kernel* (kotak kuning) pada citra di semua *offset* yang memungkinkan seperti padaGambar 4 sebagai berikut.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 4 | 3 | 4 |
| 2 | 4 | 3 |
| 2 |  |  |

***Convolved Feature Image***

**Gambar 4.** Operasi Convolutional Layer

Kotak hijau secara keseluruhan adalah citra yang akan dilakukan *convolution*. *Kernel* bergerak dari sudut kiri atas ke kanan bawah. Sehingga hasil *convolution* dari citra tersebut dapat dilihat dari gambar yang berwarna biru. Tujuan dilakukannya *convolution* pada data citra adalah untuk mengekstraksi fitur dari citra *input*. *Convolution* akan menghasilkan transformasi *linear* dari data *input* sesuai informasi spasial pada data. Bobot pada layer tersebut menspesifisikasikan kernel *convolution* yang digunakan, sehingga kernel *convolution* dapat dilatih berdasarkan *input* pada CNN(Putra et al., 2016)

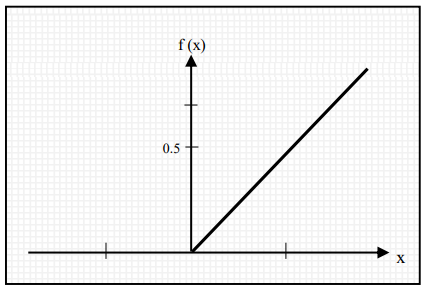
**Activation Layer**

Fungsi aktivasi merupakan operasi matematik yang dikenakan pada sinyal *output*. Fungsi aktivasi berfungsi menentukan apakah suatu *neuron* aktif atau tidak berdasarkan *weighter sum* dari suatu *input*. Beberapa jenis fungsiaktivasi yang sering digunakan pada *Deep Learning* adalah *sigmoid*, *Tanh*, algebraic *sigmoid*, *ReLU*, noisy *ReLU*, Leakly *ReLU*/P*ReLU*, *Randomized Leakly ReLU*, dan *Eksponential Linear Unit* (Khan et al., 2018). Secara garis besar fungsi aktivasi akan di jelaskan sebagai berikut:

**Fungsi *ReLU* (*Rectified Linier Unit*)**

Fungsi yang digunakan untuk aktivasi pada *ReLU* adalah sebagaimana ditunjukkan pada Gambar 5 secara umum fungsi *ReLU* dinyatakan dalam persamaaan (3.2). Dari persamaan tersebut, maka nilai *output* dari *neuron* bisa dinyatakan sebagai 0 jika *input*nya adalah negatif. Jika nilai *input* dari fungsi aktivasi adalah positif, maka *output* dari *neuron* adalah nilai *input* aktivasi itu sendiri (Krizhevsky et al., 2012).

(3.2)

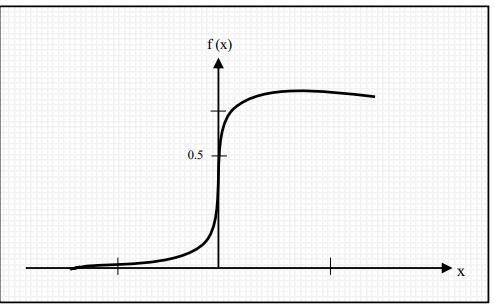


**Gambar 5.** Fungsi aktivasi ReLU

**Fungsi *Sigmoid***

Fungsi *sigmoid* adalah fungsi non-*linear* yang mempunyai persamaan matematika sebagaimana ditunjukkan pada Persamaan (3.3). Masukan untuk fungsi aktivasi tersebut adalah nilai real dan keluaran dari fungsi tersebut adalah nilai antara 0 dan 1. Jika masukannya sangat negatif, maka keluaran yang didapatkan adalah 0, sedangkan jika masukan sangat positif maka nilai keluaran yang didapatkan adalah 1. Nilai masukan dan keluaran dari fungsi *sigmoid* dapat dinyatakan dalam grafik pada Gambar 6.

(3.3)



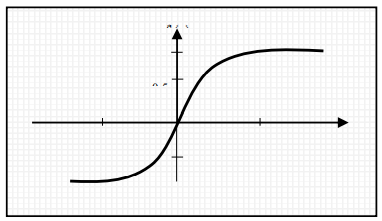
**Gambar 6.** Fungsi aktivasi sigmoid

Fungsi *sigmoid* mempunyai dua kekurangan utama yaitu pertama,bahwa fungsi ini mempunyai gradien mendekati nilai nol jika nilai *input* sangat negatif atau sangat positif. Hal ini tidak diharapkan karena nilai gradien digunakan dalam proses pelatihan. Kelemahan kedua adalah bahwa fungsi ini tidak terpusat pada nilai 0 (*zero centered*) (Khan et al., 2018).

**Fungsi *Tanh***

Fungsi *tanh* mengubah masukan yang bernilai *real* menjadi nilai antara -1 dan 1 sebagaimana ditunjukkan pada Gambar 7. Pada gambar tersebut ditunjukkan bahwa nilai yang sangat negatif dirubah menjadi nilai -1 dan nilai yang sangat positif dirubah menjadi nilai 1. Fungsi ini merupakan fungsi yang nilainya terpusat pada nilai 0 sehingga lebih dipilih dibandingkan dengan fungsi *sigmoid*. Persamaan untuk fungsi *tanh* ditunjukkan pada Persamaan (3.4) dimana nilai fungsi *tanh* merupakan dua kali nilai *sigmoid* dikurangi satu (Khan et al., 2018).

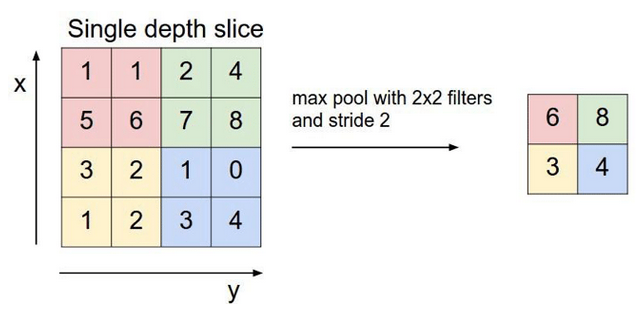
(3.4)



**Gambar 7.** Fungsi aktivasi tanh

***Pooling* Layer**

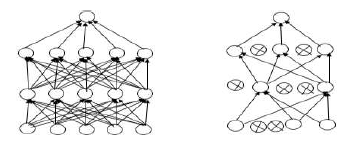
*Pooling layer* terletak setelah *convolution layer*. *Pooling layer* terdiri atas sebuah *filter* dengan ukuran dan *stride* tertentu yang secara bergantian bergeser pada seluruh area *feature map*.Jenis *poolinglayer*yang biasa digunakan yaitu *average pooling* dan *max pooling*. *Max Pooling* adalah proses untuk meningkatkan invariansi posisi dari fitur menggunakan operasi *Max*. *Max Pooling* membagi *output* dari *Convolutional Layer* menjadi beberapa grid kecil lalu mengambil nilai maksimal dari setiap grid untuk menyusun *matriks* citra yang telah direduksi yang ditunjukkan Gambar 8. Grid yang berwarna merah, hijau, kuning dan biru merupakan kelompok grid yang akan dipilih nilai maksimumnya. Sehingga hasil proses tersebut dapat dilihat pada kumpulan grid disebelah kanannya. Proses tersebut memastikan fitur yang didapatkan akan sama meskipun objek citra mengalami translasi (pergeseran)(Putra et al., 2016).



**Gambar 8**. Operasi Max Pooling

**Dropout Layer**

*Dropout* yaitu merupakan teknik regularisasi jaringan syaraf dimana beberapa akan dipilih secara *random* dan tidak dipakai selama data latih. Teknik regularisasi adalah teknik yang digunakan untuk melakukan modifikasi pada model neural network yang bertujuan untuk mengurangi generalization error, bukan mengurangi *training* error seperti peran loss function. Kemudian *neuron-neuron* ini dibuang juga secara *random*. Hal ini, artinya bahwa kontribusi *neuron* yang dibuang akan dihentikan sementara jaringan dan bobot baru juga tidak diterapkan pada *neuron* pada saat melakukan *backpropagation* (Srivastava et al., 2014).



1. *Neural Network* Biasa (b) Setelah dilakukan dropout

**Gambar 9.** Operasi Dropout (Srivastava et al., 2014)

Gambar 9merupakan jaringan syaraf biasa (a) dengan 2 lapisan yang tersembunyi. Sedangkan, untuk bagian (b) yaitu jaringan syaraf yang sudah diaplikasikan teknik regularisasi *dropout* yang terdapat beberapa *neuron* aktivasi yang tidak terpakai lagi. Teknik ini sangat mudah diterapkan pada model CNN sehingga akan berdampak pada performa model dalam melatih serta mengurangi *overfitting* (Srivastava et al., 2014).

Pada jaringan syaraf tiruan biasa, dimisalkan 𝑦l adalah nilai keluaran dari suatu lapisan 𝑙 dan 𝑧l adalah nilai masukan pada layer 𝑙 dengan 𝑊l dan 𝑏𝑙 adalah bobot dan bias dari lapisan 𝑙, dengan unit ke 𝑖 maka perhitungan proses *feedforward* menggunakan fungsi aktivasi 𝑓 dapat dilakukan dengan persamaan (3.5).

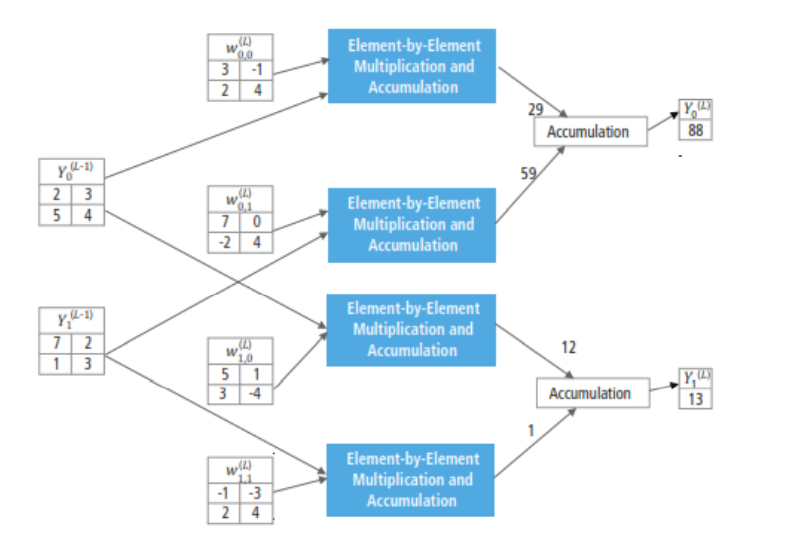
3.5

3.6

Sementara pada jaringan yang mengimplementasikan teknik Dropout, variable 𝑟l melambangkan vector sepanjang 𝑗 yang menyimpan nilai yang diperoleh dari distribusi Bernoulli. Proses feedforward dilakukan dengan persamaan (3.6).

Keluaran *feature maps* dari *convolution* akhir atau *layer pooling* biasanya menggunakan *flatten* seperti diubah menjadi satu dimensi angka/vektor array yang dihubungkan ke satu/lebih layer yang terhubung sempurna atau di sebut juga *Dense Layer*, dimana setiap masukan yang terhubung ke setiap luaran memiliki bobot (Labach et al., 2019).

**Fully-Conected Layer**

*Fully-Conected Layer* adalah sebuah lapisan dimana semua *neuron* aktivasi dari lapisan sebelumnya terhubung semua dengan *neuron* di lapisan selanjutnya sama seperti halnya dengan *Neural Network* biasa. 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 antara lapisan *Fully-Connected* dan lapisan *convolution* biasa adalah *neuron* di lapisan *convolution* terhubung hanya ke daerah tertentu pada *input*, sementara lapisan *Fully-Connected* memiliki *neuron* yang secara keseluruhan terhubung. Namun, kedua lapisan tersebut masih mengoperasikan produk dot, sehingga fungsinya tidak begitu berbeda (Lau et al., 2015). Proses *fully-connected* dapat ditunjukkan pada Gambar berikut ini:

**Gambar 10.** Processing of a Fully-Connected Layer

**Optimizer**

*Optimizer* merupakan salah satu parameter yang dibutuhkan untuk membangun model dalam *package Keras. Optimizer* memliki peranan penting untuk meningkatkan akurasi sebuah model. Beberapa *optimizer* yang terdapat dalam package Keras yaitu Adam, Adamax, dan RMSprop. Dalam penelitian ini hanya menggunakan metode Adam. Adam (*Adaptive Moment Estimation)* adalah metode yang menghitung *adaptive learning rate* untuk setiap parameter. Nilai parameter yang direkomendasikan adalah β1 = 0.9, β2 =0.999, dan ϵ = 10-8 dengan β1 = β2 = tingkat penurunan eksponensial dan ϵ = nilai epsilon untuk update parameter (Rawat & Wang., 2017).

**Softmax Layer**

Fungsi *softmax* digunakan untuk metode klasifikasi dengan jumlah kelas yang banyak, seperti regresi *logistik multinomial*, analisis diskriminan *linear multiclass*, *Naive Bayes Classifier*, dan *Artificial Neural Network* (ANN). *Softmax* adalah sebuah fungsi yang mengubah K-dimensi vektor 'x' yang berupa nilai sebenarnya menjadi vektor dengan bentuk yang sama namun dengan nilai dalam rentang 0-1, yang jumlahnya 1. Fungsi softmax digunakan dalam layer yang terdapat pada *Neural Network* dan biasanya terdapat pada layer terakhir untuk mendapatkan *output*. Tidak jauh berbeda dengan *neuron* pada umumnya softmax *neuron* menerima *input* lalu melakukan pembobotan dan penambahan bias. Tetapi setelah itu *neuron* pada softmax layer tidak menerapkan fungsi aktivasi melainkan menggunakan fungsi softmax. Jika diketahui p adalah *input* berbobot yang diterima oleh *neuron* pada softmax layer maka aktivasi yi untuk *neuron* ke-i adalah:

(3.7)

Di mana bagian penyebut pada Persamaan (3.7) merupakan total nilai masing-masing *neuron* pada *output* layer. Sehingga bisa dikatakan pada softmax layer, *output* merupakan distribusi probabilitas untuk setiap kelas. Penyebutnya memastikan bahwa *output* ke-i berjumlah mendekati 1. Dengan menggunakan *softmax* kita bisa menafsirkan *output* jaringan sebagai perkiraan (Dzulqarnain et al., 2019).

***Cross Entropy Loss Function***

*Loss Function* atau *Cost Function* merupakan fungsi yang menggambarkan kerugian yang terkait dengan semua kemungkinan yang dihasilkan oleh model. *Loss Function* bekerja ketika model pembelajaran memberikan kesalahan yang harus diperhatikan. *Loss Function* yang baik adalah fungsi yang menghasilkan error yang diharapkan 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, dan selama pelatihan banyak algoritma yang dapat menyesuaikan parameter sehingga perbedaan ini diminimalkan (Zhu et al., (2020). *Crossentropy* adalah pilihan yang masuk akal. Gambaran umum algoritma ini adalah meminimalkan kemungkinan log negatif dari dataset, yang merupakan ukuran langsung dari performa prediksi model. Fungsi yang dilakukan pada proses *Cross Entropy Loss Function* dapat ditunjukkan pada persamaan (3.8) sebagai berikut:

(3.8)

**Confusion *Matriks***

Penentuan baik atau tidaknya performa suatu model klasifikasi dapat dilihat dari parameter pengukuran performanya, yaitu tingkat akurasi, recall, dan presisi. Untuk menghitung faktor-faktor tersebut diperlukan sebuah matrik yang biasa disebut confusion *matriks*. Persamaan-persamaan *matriks* yang digunakan yaitu *Recall, Precision* dan *Accuracy/F1-Score*.

**Recall**

3.9

Recall atau sensitifitas sistem terhadap suatu kelas juga dapat dilihat. Recall dapat dihitung dengan menggunakan persamaan (3.9).

**Precision**

Sementara untuk menghitung tingkat presisi prediksi kejadian dapat digunakan persamaan (3.10).

3.10

**Accuracy**

Presisi menggambarkan seberapa tepat suatu model memprediksi kejadian positif dalam serangkaian kegiatan prediksi. Perhitungan presisi biasanya bermanfaat pada pengembangan model prediksi hujan di suatu daerah. Selain presisi dan akurasi, untuk dapat melihat lebih detail lagi kinerja suatau sistem,

3.11

Akurasi digunakan sebagai parameter sebagaimana akurat suatu model melakukan klasifikasi (Fawcett, 2006).

## Kelapa Sawit

Kelapa Sawit (*Elaeis*) adalah tumbuhan industri sebagai bahan baku penghasil minyak yang termasuk dalam keluarga Arecaceae yang terdiri dari dua spesies yaitu kelapa sawit Afrika (*Elaeis guineensis*) dan kelapa sawit Amerika (*Elaeis oleifera*). Kelapa sawit merupakan salah satu komoditas hasil perkebunan yang mempunyai peran cukup penting dalam kegiatan perekonomian di Indonesia karena kemampuannya menghasilkan minyak nabati yang banyak dibutuhkan oleh sektor industri. Sifatnya yang tahan oksidasi dengan tekanan tinggi dan kemampuannya melarutkan bahan kimia yang tidak larut oleh bahan pelarut lainnya, serta daya melapis yang tinggi membuat minyak kelapa sawit dapat digunakan untuk beragam peruntukan, diantaranya yaitu untuk minyak masak, minyak industri, maupun bahan bakar (biodiesel) (BPS, 2020). Buah kelapa sawit memiliki varietas atau jenis terbaik dalam produksi buahnya, seperti varietas Dy x P Sungai Pancur 1 (dumpy) merupakan varietas kelapa sawit terbaik untuk saat ini yang mampu tumbuh hingga 30 tahun, lebih lama dari varietas lain, juga pada kasus dimana pohon kelapa sawit di tanam pada ketinggian tertentu, memang berpengaruh terhadap kualitas dari minyak sawit mentah, itu karena pada ketinggian 800 mdpl kualitas buah kelapa sawit akan lebih baik karena kandungan karoten dari setiap buah itu lebih tinggi dari pada buah yang di tanam dibawah ketinggian tersebut, akan tetapi karakteristik morfologi dalam menentukan tingkat kematangan kelapa sawit itu sama saja (Pusat Penelitian Kelapa Sawit, 2020). Suatu indikator/alat ukur kematangan tandan buah segar yang diukur melalui jumlah buah yang membrondol dan warna dari buah tersebut. Pengetahuan akan fraksi buah kelapa sawit berfungsi untuk memperkirakan hasil panen secara kuantitas dan kualitas yang akan diperoleh. Idealnya, proses panen TBS dilakukan pada fraksi 3 atau buah matang (Pahan, 2015). Berikut ini merupakan karakteristik kelapa sawit tersebut:

1. Sawit Mentah

Sawit mentah merupakan kategori buah kelapa sawit yang belum siap diolah. Karakter morfologi dari buah tersebut adalah warna kulit masih berwarna hijau kehitaman dan tidak ada buah yang memberondol. Berdasarkan deskripsi tersebut bentuk gambar dapat dilihat pada Gambar 11 Sebagai berikut:



**Gambar 11.** Sawit Mentah

1. Sawit Matang

Sawit matang merupakan kategori buah kelapa sawit yang siap untuk di olah. Karakter morfologi dari buah tersebut adalah Buah kelapa sawit yang matang mempunyai warna cerah yang didominasi oleh warna kuning kemerahan dan 12-25% buah memberondol. Berdasarkan deskripsi tersebut bentuk gambar dapat dilihat pada Gambar 12 Sebagai berikut:



**Gambar 12.** Sawit Matang

1. Sawit Busuk

Sawit busuk merupakan kategori buah kelapa sawit yang siap diolah atau sebaliknya. Karakter morfologi dari buah tersebut adalah warna kulit coklat kehitaman (Supriadi, 2019). Berdasarkan deskripsi tersebut bentuk gambar dapat dilihat pada Gambar 13 Sebagai berikut:



**Gambar 13.** Sawit Busuk

## Penelitian Terdahulu

Berdasarkan penelitian yang akan dilakukan, acuan dari beberapa penelitian terdahulu menjadi sangat penting dalam melakukan sebuah penelitian dengan tujuan untuk mengetahui hubungan antara penelitian yang akan dilakukan dengan penelitian terdahulu, sehingga dengan menambahkan acuan tersebut dapat menghindari adanya suatu duplikasi dalam penelitian yang akan dilakukan.

Banyak pengembangan sistem yang menjadikan *Computer Vision* sebagai acuan dalam pembuatannya, seperti *image recognition*, *face detection* maupun pengenalan pola tertentu lainnya. Pengembangan sistem tersebut menjadi sebuah fungsionalitas yang dapat membantu dan mempermudah perkerjaan diberbagai bidang. Pengembangan dari *Deep Learning* tentunya menjadi alternatif yang tepat dan efektif digunakan untuk menyelesaikan permasalahan tersebut. Hal ini tidak lepas dengan adanya riset atau penelitian di bidang tersebut. Penelitian terdahulu mengenai *Deep Learning* menggunakan *Convolutional Neural Network* sudah banyak dilakukan oleh para peneliti pada berbagai macam object yang digunakan. Berikut merupakan perbandingan diantara penelitian-penelitian terdahulu:

**Tabel 1.** Tabel Perbandingan Pustaka Metode CNN

|  |  |  |  |
| --- | --- | --- | --- |
| No | Peneliti | Topik | Keterangan |
| 1. | Rismiyati dan Azhari (2016). | Implementasi *Convolutional Neural Network* untuk Sortasi Mutu Salak Ekspor Berbasis Citra Digital. | Dengan menggunakan learning rate 0.0001, Akurasi yang diperoleh adalah 81,5%. Model empat kelas mendapat akurasi 70,7% dengan dua lapisan *convolution*. |
| 2. | Astuti et al., (2019). | *Real-Time Classification of Facial expressionsusing a Principal Component Analysis and Convolutional Neural Network.* | Pada penelitian tersebut menggabungkan konsep Principal Component Analysis dengan metode *Convolutional Neural Network*. Jadi PCA akan di inisiasi terlebih dahulu sebelum pembuatan model *Convolutional Neural Network*. Kelas yang di gunakan berjumlah enam kelas yaitu angry, disgust, sad, surprised, netral dan happy. Kombinasi antara Principal Component Analysis dengan metode *Convolutional Neural Network* memperoleh nilai akurasi sebesar 95% dan lebih tinggi dari penelitian sebelumnya. |
| 3. | Fadhilah et al., (2019). | *Non-Halal Ingredients Detection of Food Packaging Image Using Convolutional Neural Networks.* | Memperoleh hasil akurasi sangat baik, yang diperoleh hingga 98,08%. Namun, proses penggabungan karakter dan mencocokkan daftar bahan baku membuat akurasi hanya 50%. Kendala tersebut karena spasi antara huruf dan huruf kecil ukuran dan permukaan yang tidak rata. |
| 4. | Tibor Trnovszky dkk, (2018). | Implementasi *Convolutional Neural Network* (CNN) Pada Pengenalan Hewan Dengan Membandingkan Beberapa Metode Klasifikasi. | *Convolutional Neural Network* menjadi metode dengan nilai akurasi paling baik, yaitu sebesar 98 %. |
| 5. | Razak et al, (2018). | Klasifikasi Kualitas Dari Kematangan Buah Mangga Menggunakan Metode *Fuzy Inference Engine.* | Menghasilkan tingkat akurasi sebesar 80%. Serta memiliki kelemahan kurang akurat dalam memprediksi buah |

\*Sumber: Penelitian Terdahulu

# III. METODOLOGI PENELITIAN

## Tempat dan Waktu Penelitian

Penelitian ini dilaksanakan di Universitas Jambi. Waktu yang digunakan untuk penelitian ini dilaksanakan pada bulan Oktober 2020 sampai dengan Mei 2021. Pada Tabel 2 merupakan *timeline* penelitian berikut:

**Tabel 2.** Timeline Penelitian

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Kegiatan | Okt | Nov | Des | Jan | Feb | Mar | Mei |
| Literatur review |  |  |  |  |  |  |  |
| Penulisan Skripsi |  |  |  |  |  |  |  |
| Collecting Data |  |  |  |  |  |  |  |
| Praproses data |  |  |  |  |  |  |  |
| Rancangan model |  |  |  |  |  |  |  |
| Pengujian model |  |  |  |  |  |  |  |
| Pengujian Gambar |  |  |  |  |  |  |  |

## Perangkat Penelitian

Perangkat keras yang digunakan dalam penelitian ini adalah komputer personal dengan spesifikasi sebagai berikut:

* Prosesor : AMD Ryzen 3 Vega 3
* RAM : 4 GB
* Memori : 1 TB

Perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut:

* Sistem operasi *Windows* 10 (64-bit)
* *Jupyter*, Bahasa pemrograman *Python* 3.8, *Package Keras*, sklearn, streamlit, tkinter dan *Package* *Tensorflow* untuk membuat model klasifikasi CNN.

## Variabel Penelitian

Berikut ini merupakan variabel penelitian beserta penjelasan dari masing-masing variabel, variabel tersebut dapat dilihat pada Tabel 3 sebagai berikut:

**Tabel 3.** Variabel Penelitian

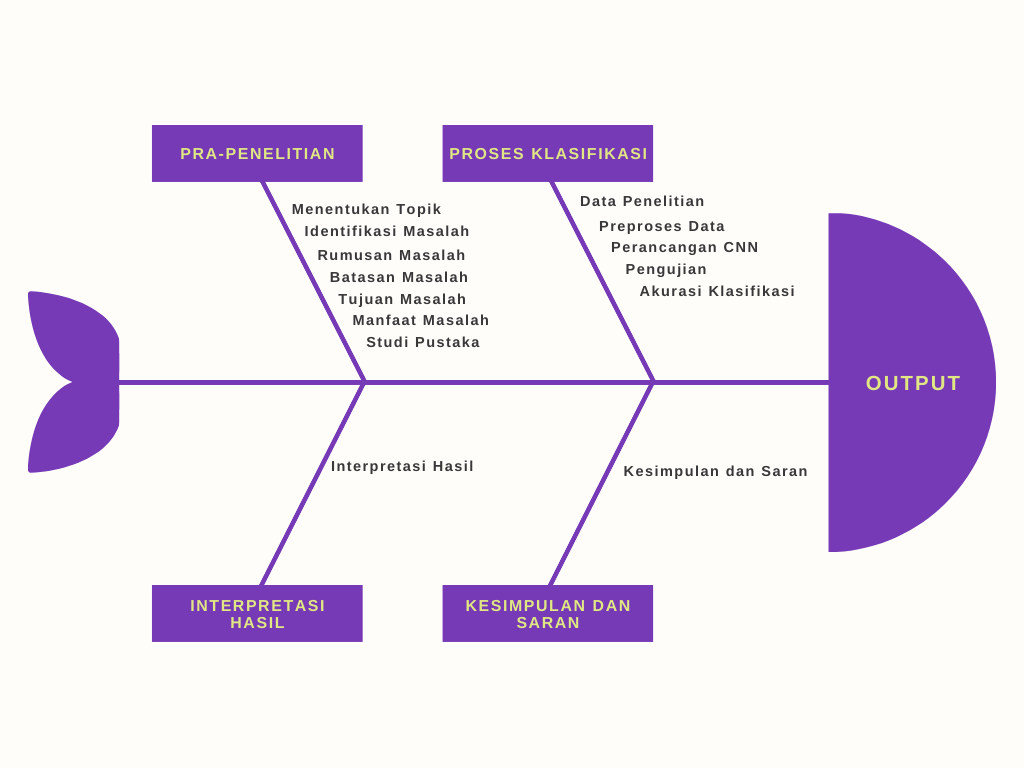
|  |  |  |
| --- | --- | --- |
| **Citra Buah Sawit** | **Variabel** | **Definisi Variabel** |
|  | Sawit Mentah | Sawit mentah merupakan kategori buah kelapa sawit yang belum siap diolah. Ciri dari buah tersebut adalah warna kulit masih berwarna hitam. |
|  | Sawit Matang | Sawit matang merupakan kategori buah kelapa sawit yang siap untuk di olah. Ciri dari buah tersebut adalah warna kulit berwarma merah atau orange. |
|  | Sawit Busuk | Sawit busuk merupakan kategori buah kelapa sawit yang terlewat masak. Ciri-ciri nya yaitu warna kulit coklat kehitaman. |

## Metode Analisis Data

Pada penelitian ini digunakan metode untuk mengidentifikasi gambar buah sawit dengan algoritma *Convolutional Neutral Network* (CNN) yang merupakan salah satu metode dalam *Deep Learning* yang terkenal mampu mengklasifikasi data gambar dengan baik. Proses untuk mengolah algoritma CNN dibantu oleh *software Jupyter Notebook* dengan bahasa pemograman *Python 3.6* dan *package* *Tensorflow*. Adapun cara kerja yang dilakukan adalah dengan mengenali objek atau citra sebagai *input* dan *output* yang diharapkan adalah tingkat akurasi pengenalan objek tersebut.

## Tahapan Penelitian

Berikut merupakan tahapan penelitian yang dilakukan dalam penelitian ini:



**Gambar 14.** Diagram fishbone penelitian

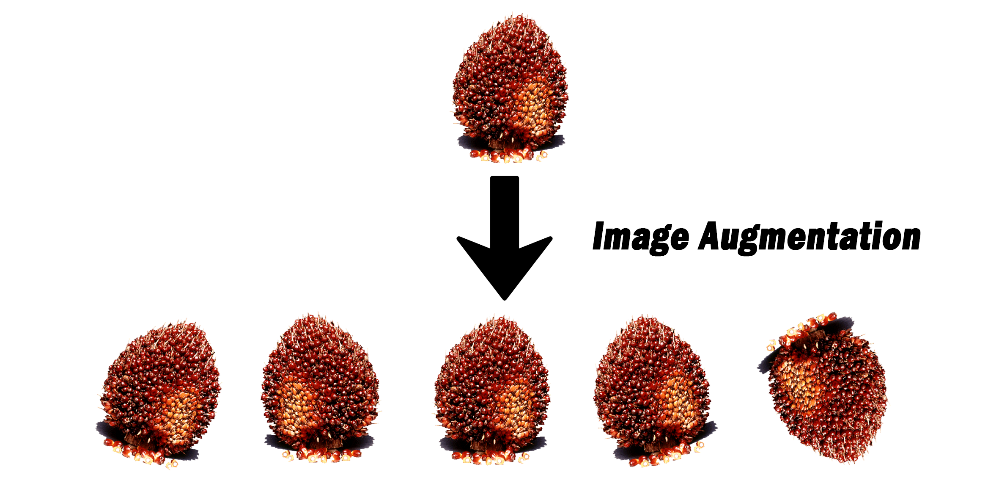
## Data Penelitian

Dalam melakukan penelitian ini, langkah pertama yaitu pengumpulan dataset berupa citra buah kelapa sawit yang dibagi menjadi tiga jenis kelas, yaitu gambar buah kelapa sawit mentah, matang dan busuk. Dataset diambil dari berbagai sumber internet dan juga di foto secara langsung pertandan menggunakan kamera handphone sesuai kelas yang telah ditentukan. Dataset yang akan digunakan berupa buah kelapa sawit pertandan, walaupun struktur warna buah kelapa sawit tidak sama rata akan tetapi buah kelapa sawit yang sesuai kategori diatas memiliki warna dominan, misalnya buah kelapa sawit matang warna dominannya merah dan komputer dapat mengenali hal tersebut dengan baik. Dataset berjumlah 400 gambar yang di bagi menjadi 2 bagian, proses pembagian menggunakan teknik *Train Test Split* yang disediakan oleh *package Sklearn,* selanjutnya data dibagi menjadi 80% data *training* dan 20% data testing. Data *training* merupakan kumpulan data yang akan digunakan untuk membuat model yang akan dibangun, data *testing* digunakan untuk melakukan pengujian terhadap model yang telah di buat. Kemudian dilakukan desain arsitektur dimulai dari menentukan kedalaman jaringan, susunan layer, dan pemilihan jenis layer yang akan digunakan untuk mendapatkan model berdasarkan *input* dataset.

## Preproses Data

Sebelumnya dataset akan di bagi menjadi beberapa bagian, yaitu 80% data latih dan 20% data *testing*. Kemudian setiap gambar buah kelapa sawit akan dirubah dan disimpan berukuran 128x128 pixel dengan *channel* berukuran 3, *input shape* berukuran 128x128x3, *batch size* 16 dan *epoch* 10 untuk kemudian dijadikan sebagai dataset. Gambar tersebut kemudian di augmentasi terlebih dahulu sebelum masuk jaringan.

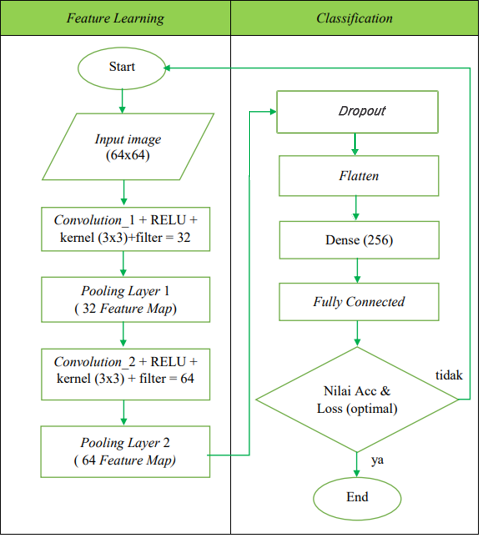
Menurut Wang & Perez (2017) *Augmentasi* merupakan proses mengubah atau memodifikasi gambar sedemikian rupa sehingga komputer akan mendeteksi bahwa gambar yang diubah adalah gambar yang berbeda, namun manusia masih dapat mengetahui bahwa gambar yang diubah tersebut adalah gambar yang sama. Pada penelitian ini dilakukan proses augmentasi dengan cara *rescaling* menjadi 1/255, *shearing* dengan skala 0.2, *zooming image* dengan *range* 0.2 dan melakukan *horizontal flip,* berikut merupakan hasil gambar dari proses augmentasi.



**Gambar 15.** Image Augmentation

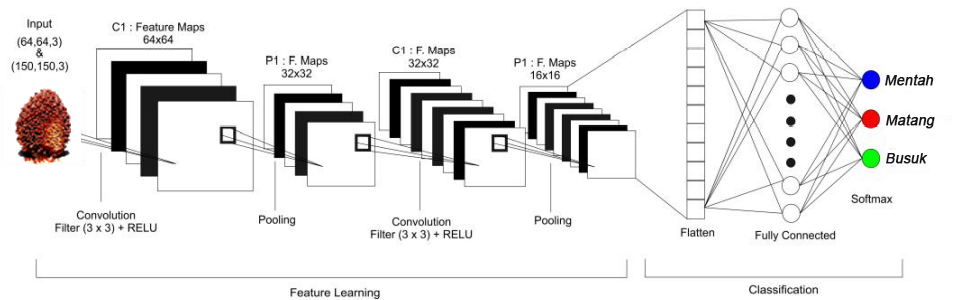
## Rancangan *Convolutional Neural Network* (CNN)

Setelah dilakukan pembuatan data, langkah selanjutnya adalah melakukan pelatihan model CNN. Umumnya dalam CNN memiliki 2 tahapan, yaitu tahap *feature learning* dan *classification*. *Input* gambar pada model CNN menggunakan citra yang berukuran 128x128x3. Angka tiga yang dimaksud adalah sebuah citra yang memiliki 3 channel yaitu *Red, Green*, dan *Blue* (RGB) Citra masukan kemudian akan diproses terlebih dahulu melalui proses *convolution* dan proses *pooling* pada tahapan *feature learning*. Jumlah proses *convolution* pada rancangan ini memiliki dua lapisan *convolution*. Setiap *convolution* memiliki jumlah *filter* dan ukuran kernel yang berbeda. Kemudian dilakukan *dropout* yang di mulai dengan proses *flatten* atau proses mengubah *feature map* hasil *pooling layer* kedalam bentuk *vector*. Proses ini biasa disebut dengan tahap *fully Connected layer*. Berikut adalah rancangan dari arsitektur CNN pada penelitian ini yang dapat dilihat pada Gambar 16.



**Gambar 16.** Rancangan Arsitektur CNN Proses Training

Berdasakan arsitektur tersebut dijelaskan terdapat dua tahap dalam arsitektur CNN, yaitu *Feature Learning* dan *classification*. *Feature learning* adalah teknik yang memungkinkan sebuah system berjalan secara otomatis untuk menentukan representasi dari sebuah *image* menjadi *features* yang berupa angka-angka yang merepresentasikan *image* tersebut. Tahap *Classification* adalah sebuah tahap dimana hasil dari *feature learning* akan digunakan untuk proses klasifikasi berdasarkan *subclass* yang sudah ditentukan. Jika *flowchart* diatas diubah kedalam bentuk gambar, maka dapat dilihat seperti Gambar 17. Berdasarkan gambar tersebut pada proses *convolution* pertama menggunakan jumlah *filter* sebanyak 64 dan kernel dengan *matriks* 3x3. Kemudian dilakukan proses *pooling* menggunakan ukuran *pooling* 2x2 dengan pergeseran *mask* sebanyak dua langkah. Kemudian pada tahapan *convolution* kedua dengan menggunakan jumlah *filter* sebanyak 32 dan *kernel* dengan *matriks* 2x2. Kemudian di lanjutkan dengan *flatten* yaitu merubah *output* dari proses *convolution* yang berupa *matriks* menjadi sebuah *vector* yang selanjutnya akan diteruskan pada proses klasifikasi dengan menggunakan MLP (*Multi-Layer Perceptron*) dengan jumlah *neuron* pada lapisan tersembunyi yang telah ditentukan. Kelas dari citra kemudian diklasifikasikan berdasarkan nilai dari *neuron* pada lapisan tersembunyi dengan menggunakan fungsi aktivasi *softmax*.



**Gambar 17.** Flowchart Proses Training

## Rancangan Pengujian Model

Pengujian dilakukan untuk melakukan evaluasi terhadap model yang dihasilkan oleh CNN. Tahap *Testing* adalah tahap pengujian model yang sudah dilakukan pada tahap pelatihan. Jumlah data uji dalam penelitian ini sebanyak 80 data gambar, dengan jumlah gambar perkelas sebanyak 26 gambar. Pada tahap ini model di uji dengan gambar yang berbeda dengan tujuan menguji apakah model sudah menghasilkan performa yang baik dalam mengklasifikasikan sebuah gambar. Evaluasi model ini dilakukan dengan menggunakan fitur dari *Tensorflow* yaitu *model.evaluate*  dan juga menggunakan teknik *confusion matrix* untuk melihat lebih rinci dari evaluasi modelnya. Pada Tabel 4 merupakan rancangan *confusion matriks* penelitian ini.

**Tabel 4.** Rancangan Confusion Matriks

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Matriks* | | Prediksi Kelas | | |
| Mentah | Matang | Busuk |
| Kelas sebenarnya | Mentah |  |  |  |
| Matang |  |  |  |
| Busuk |  |  |  |

Setelah mendapatkan nilai prediksi selanjutnya dilakukan perhitungan akurasi dari keseluruhan *matriks* dengan persamaan sebagai berikut:

(Fawcett, 2006).

## Rancangan Pengujian Gambar

Pada proses ini dilakukan untuk mengimplementasikan model terhadap data citra. Pada proses ini dilakukan beberapa tahap untuk menghasilkan nilai akurasi dari citra yang di uji, tahap pertama yaitu dengan *input* parameter untuk data citra, lalu definisikan label yang akan dilihat hasilnya, setelah itu lakukan definisi praproses data yang mana data tersebut sebelum dilihat hasil ujinya akan di praproses terlebih dahulu supaya data lebih terstruktur, kemudian load model yang telah di buat sebelumnya, selanjutnya *input*kan gambar yang akan di uji dan yang terakhir yaitu melakukan prediksi citra atau hasil proses pengujian citra, maka akan keluar hasil prediksi sesuai label yang telah di definisi sebelumnya. Jika dijadikan *flowchart* maka proses pengujian gambar dapat di lihat pada Gambar 18 Sebagai berikut:

Load Model

Praproses Data

Define Label

*Input* Parameter

Predict Image

*Input* Image



**Gambar18*.*** Flowchat *proses Testing*.

# IV. HASIL DAN PEMBAHASAN

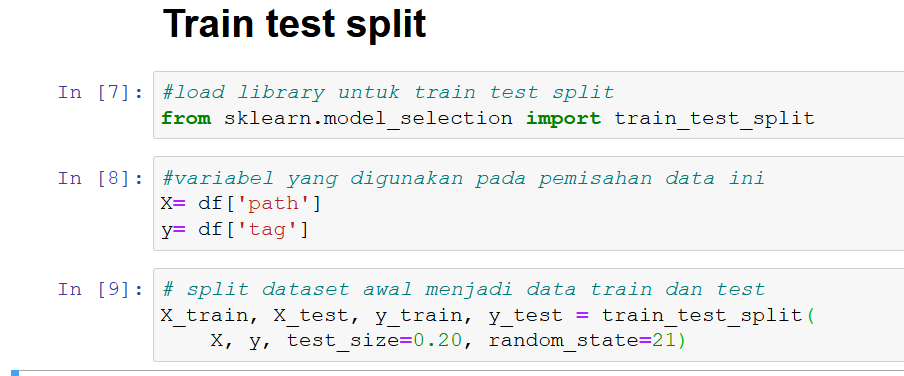
## Hasil Pengolahan Data Penelitian

Pada tahap ini dilakukan pengumpulan data terlebih dahulu, proses pengumpulan data dilakukan dengan dua cara yaitu: mengunduh gambar yang tersedia di internet sesuai dengan kriteria yang telah di tentukan dan melakukan foto secara langsung menggunakan kamera telepon genggam. Setelah data terkumpul selanjutnya gambar tersebut di olah terlebih dahulu menggunakan aplikasi *photosop* untuk menghilangkan *background* dan merubahnya menjadi warna putih, hal ini dilakukan untuk memaksimalkan kinerja program dalam membaca *pixel* gambar, karena jika gambar tersebut menggunakan *background* bawaan yang warnanya *random* akan mempengaruhi pembacaan program terhadap *pixel* gambar. Pada Gambar 19 merupakan citra sebelum dan sesudah dilakukan penghapusan *background*:



**Gambar 19.** Penghapusan background

Setelah proses tersebut selanjutnya dilakukan pembagian data dengan teknik split data yang di sediakan oleh modul sklearn, berikut potongan kode yang digunakan untuk membagi data:

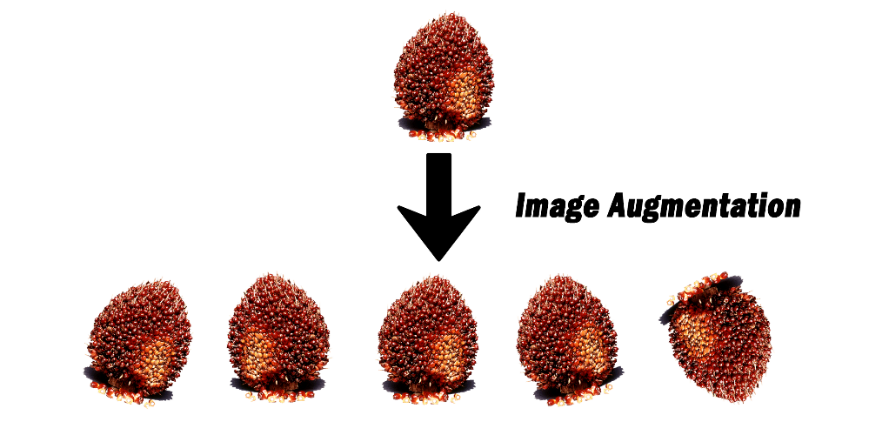


**Gambar 20.** Kode pembagian Data

Pada potongan kode di atas di tentukan terlebih dahulu label yang akan di pakai setelah itu berdasarkan data yang telah terbaca akan di lakukan split data menggunakan teknik *train\_test\_split* dari modul *sklearn* dengan membagi data menjadi 80% dan 20% setelah dilakukan proses diatas maka hasilnya yaitu jumlah data yang akan digunakan dalam penelitian ini terbagi menjadi 2 bagian, yaitu data *training* yang berjumlah 320 citra dan data *testing* berjumlah 81 citra.

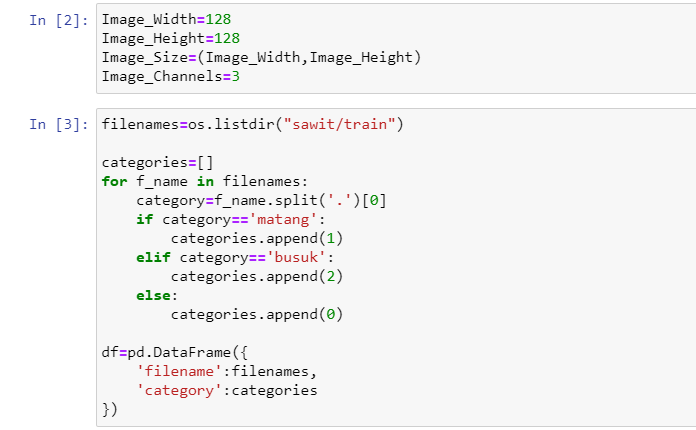
## Hasil Praproses Data

Pada penelitian ini praproses data dilakukan dengan teknik *augmentasi* supaya komputer akan mendeteksi bahwa gambar yang diubah adalah gambar yang berbeda. Proses *augmentasi* yang dilakukan yaitu berupa *rotation range* menjadi 15, *rescaling* menjadi 1/255, *shearing* dengan skala 0.2, melakukan *horizontal flip* kemudian melakukan *width* dan *height shift* dengan *range* 0.1. setelah dilakukan proses tersebut variable yang telah di buat sebelumnya kemudian akan di panggil lagi untuk proses implementasinya terhadap dataset, bagian *train\_generator* begitu juga untuk data yang lainnya. Serta untuk hasil dari praproses dapat di lihat pada Gambar 21.



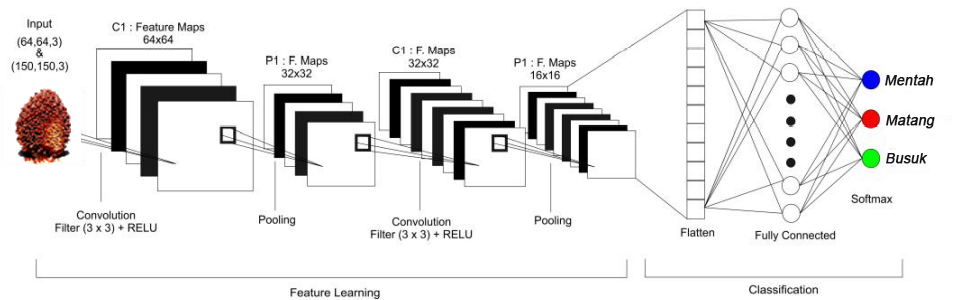
**Gambar 21.** Hasil proses Augmentasi

Namun sebelum melakukan proses praproses, harus dilakukan pendefinisian data. program untuk pendefinisian data dapat dilihat pada Gambar 22. Dapat dilihat pada proses tersebut dilakukan pendefinisian ukuran gambar dan *channel* nya yang berada pada baris 2 kemudian dilakukan pembacaan lokasi data gambar yang akan digunakan untuk inisiasi label yang akan di gunakan, untuk prosesnya dapat dilihat pada Gambar 21 baris ke 3. Hasil dari pendefinisian data pada potongan kode tersebut yaitu data yang digunakan akan menjadi seragam dengan ukuran 128x128, kemudian juga dilakukan pendefinisian label supaya data mampu dibaca oleh sistem sesuai label yang di definisikan.

******

**Gambar 22.** Definisi Parameter

## Hasil Rancangan *Convolutional Neural Network*



**Gambar 23.** Arsitektur Jaringan

Gambar 23 merupakan arsitektur jaringan pada proses *training* untuk menghasilkan model yang optimal. Penelitian ini menggunakan *input* gambar dengan ukuran 128x128x3, tujuannya adalah untuk membandingkan nilai akurasi berdasarkan ukuran gambarnya. Arsitektur diatas dapat dijelaskan sebagai berikut:

1. Proses *Convolution* pertama digunakan kernel berukuran 3x3 dan jumlah *filter* sebanyak 128 *filter*, proses *convolution* ini adalah proses kombinasi antara dua buah *matriks* yang berbeda untuk menghasilkan suatu nilai *matriks* yang baru. Setelah proses *convolution*, maka ditambahkan sebuah aktivasi fungsi yaitu RELU (*Retrified Linear Unit*). Fungsi aktivasi ini bertujuan untuk mengubah nilai negative menjadi nol (menghilangkan nilai negatif dalam sebuah *matriks* hasil *convolution*). Hasil *convolution* ini memiliki ukuran yang sama yakni 128x128 karena pada saat proses *convolution* digunakan nilai *padding* 0.
2. Proses *pooling* pada dasarnya terdiri dari sebuah *filter* dengan ukuran dan tertentu yang akan secara bergantian bergeser pada seluruh area feature map. Penelitian ini menggunakan *max-pooling* untuk mendapatkan nilai *matriks* yang baru hasil dari proses *pooling*. Berdasarkan hasil *pooling* menghasilkan *matriks* baru berukuran 63x63 dengan menggunakan kernel *pooling* 2x2. Cara kerja *max*-*pooling* adalah mengambil nilai paling maksimum berdasarkan pergeseran kernelnya sebanyak nilai stridenya yaitu 2.
3. Proses *Convolution* kedua yaitu meneruskan hasil dari proses *pooling* pertama yakni dengan *input* *matriks* gambar sebesar 32x32 dengan jumlah *filter* sebanyak 64 *filter* dan dengan ukuran kernel 3x3. Proses *convolution* kedua ini sama-sama menggunakan fungsi aktivasi RELU.
4. Proses selanjutnya masuk ke proses *pooling* yang kedua, proses ini hampir sama dengan proses *pooling* yang pertama, namun ada perbedaan pada nilai *output* akhir dari matriksnya. *Output* yang dihasilkan memiliki ukuran gambar 30x30.
5. Proses *Convolution* ketiga yaitu meneruskan hasil dari proses *pooling* pertama yakni dengan *input* *matriks* gambar sebesar 32x32 dengan jumlah *filter* sebanyak 32 *filter* dan dengan ukuran kernel 3x3. Proses *convolution* kedua ini sama-sama menggunakan fungsi aktivasi RELU.
6. Proses selanjutnya masuk ke proses *pooling* yang ketiga, proses ini hampir sama dengan proses *pooling* hang pertama, namun ada perbedaan pada nilai *output* akhir dari *matriks*nya. *Output* yang dihasilkan memiliki ukuran gambar 14x14.
7. Selanjutnya *Flatten* atau *fully-connected*. Pada tahap ini digunakan hanya satu *hidden layer* pada jaringan MLP (*Multi-Layer Perceptron*). *Flatten* disini mengubah *output* *pooling* *layer* menjadi sebuah *vector*. Sebelum melakukan proses klasifikasi atau memprediksi gambar, pada proses ini digunakan nilai *Dropout. Dropout* adalah sebuah teknik regulasi jaringan syaraf dengan tujuan memilih beberapa *neuron* secara acak dan tidak akan dipakai selama proses pelatihan, dengan kata lain *neuron-neuron* tersebut dibuang secara acak. Tujuan dari proses ini yaitu mengurangi *overfitting* pada saat proses *training*. Jadi pada proses *dropout* data yang sebelumnya terbagi menjadi beberapa *neuron* kemudian diambil beberapa *neuron* saja yang akan digunakan dalam proses *training* hal ini supaya dapat mengurangi overfitting dan menghemat waktu pada proses pembuatan model.

Berdasarkan uraian penjelasan dari arsitektur jaringan diatas, arsitektur tersebut digunakan untuk proses *training*. Sehingga dari proses *training* didapatkan model dari arsitektur tersebut. Berikut model yang terbentuk:

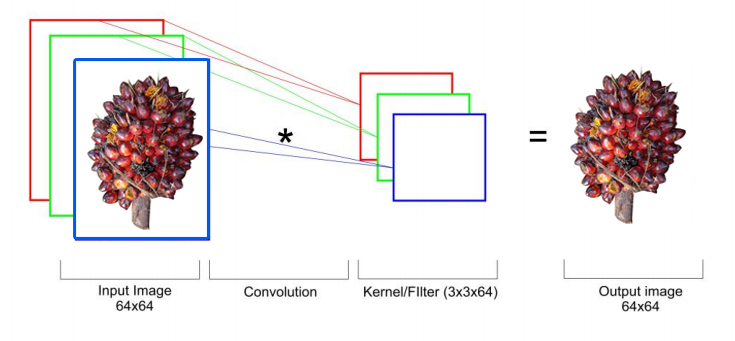
**Tabel 5.** Model CNN

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama | Size | Parameter |
| 0 | *input* | 128\*128\*3 | 0 |
| 1 | conv2d\_2 | 3\*3\*3+1\*128 | 896 |
| 2 | batch\_normalization\_1 | 256+256 | 512 |
| 3 | MaxPool\_1 | 63\*63\*128 | 0 |
| 4 | dropout\_1 | 63\*63\*128 | 0 |
| 5 | conv2d\_2 | 3\*3\*128+1\*64 | 73792 |
| 6 | batch\_normalization\_2 | 128+128 | 256 |
| 7 | MaxPool\_2 | 30\*30\*64 | 0 |
| 8 | dropout\_2 | 61\*61\*64 | 0 |
| 9 | conv2d\_3 | 3\*3\*64+1\*32 | 18464 |
| 10 | MaxPool\_3 | 14\*14\*32 | 0 |
| 11 | dropout\_3 | 14\*14\*32 | 0 |
| 12 | flatten | 6272 | 0 |
| 13 | dense | 6272\*512+512 | 3211776 |
| 14 | Dense *output* | 512+1\*3 | 1539 |
|  | Total | | 3.312.099 |

Tabel diatas merupakan model yang terbentuk dari hasil *training*. Untuk menghitung *input* kedalam konvo digunakan rumus “*input*\_*size* + 2\**padding* - (*filter*\_*size* -1)”. Total parameter yang terbentuk dari model sebanya 3.312.099 *neuron*.

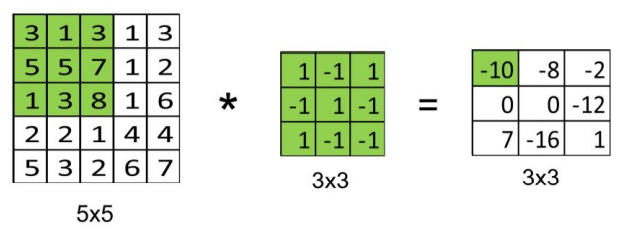
**Proses *Convolution Layer***

Berdasarkan hasil dari arsitektur jaringan mengenai proses *convolution* dapat ditunjukkan pada Gambar 24.



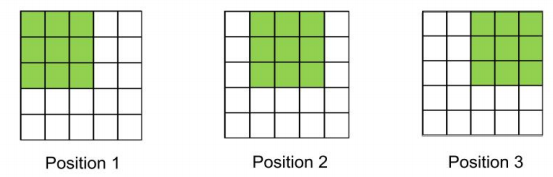
**Gambar 24.** Proses Convolution

*Convolution* merupakan proses mengkombinasi dua buah deret angka yang menghasilkan deret angka yang ketiga. Jika di implementasikan angka pada *convolution* ini adalah berbentuk *matriks* *array*. Pada *input* citra memiliki ukuran piksel 128x128x3, ini menunjukan bahwa tinggi dan lebar piksel dari gambar sebesar 128 dan citra tersebut memiliki 3 *channel* yaitu *red, green*, dan *blue* atau yang biasa disebut dengan RGB. Setiap *channel* piksel memiliki nilai *matriks* yang berbeda-beda. *Input* akan di *convolution* dengan nilai *filter* yang sudah ditentukan. *Filter* merupakan blok lain atau kubus dengan tinggi dan lebar yang lebih kecil namun kedalaman yang sama yang tersapu di atas citra dasar atau citra asli. *Filter* digunakan untuk menentukan pola apa yang akan dideteksi yang selanjutnya di konvolusi atau dikalikan dengan nilai pada *matriks* *input*, nilai pada masing-masing kolom dan baris pada *matriks* sangat bergantung pada jenis pola yang akan dideteksi. Jumlah *filter* pada konvolusi ini sebanyak 128 piksel dengan ukuran kernel (3x3), ini artinya gambar yang dihasilkan dari hasil *convolution* akan sebanyak 128 fitur map. Pada penelitian ini digunakan sampel *matriks* pada *input* citra agar lebih mudah dipahami pada proses *convolution.* Karena *input* *image* memiliki ukuran piksel 128x128, maka pada penelitian ini hanya diambil sebagian nilai *matriks* saja yang dijadikan sampel dalam proses *convolution*.



**Gambar 25.** Perhitungan Proses Convolution

Pada Gambar 25 menunjukan proses *convolution* dengan menggunakan ukuran kernel 3x3, dengan menggunakan *stride* 1. *Stride* adalah jumlah pergeseran *kernel* terhadap *matriks* *input* berjumlah satu. Hasil *stride* ditunjukkan pada Gambar 26 sebagai berikut:



**Gambar 26.** Posisi Kernel pada Convolution

Pada Gambar 26 menunjukan perhitungan *dot product* pada proses *convolution* dimana sebuah *kernel* ukuran 3x3 yang dimulai pada sisi bagian kiri. Proses ini disebut dengan *sliding window*. Namun pada penelitian ini diberikan nilai padding 1, yaitu adanya penambahan nilai 0 disekeliling nilai *matriks* *input* supaya *input* dan *output* memiliki nilai *matriks* yang sama, sehingga tidak mengurangi informasiinformasi pada gambar. Proses ini dilakukan dari ujung kiri atas sampai ujung kiri bawah. Syarat agar dua buah *matriks* dapat dikalikan adalah *matriks* pertama harus memiliki jumlah kolom yang sama dengan jumlah baris pada *matriks* kedua. Ordo *matriks* hasil perkalian dua buah *matriks* adalah jumlah baris pertama dikali jumlah kolom ke dua. Perhitungan *dot product* dapat ditunjukkan pada Tabel 6 sebagai berikut:

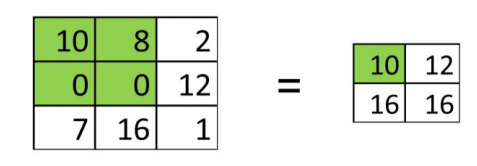
**Tabel 6.** Perhitungan dot product

|  |  |  |
| --- | --- | --- |
| Posisi | Perhitungan | Hasil |
| 1 | (3𝑥1) + (5𝑥(−1) + (1𝑥1) + (1𝑥(−1)) + (5𝑥1) + (3𝑥(−1)) + (3x1) + (7x (−1)) + (8x1) | -10 |
| 2 | (1𝑥1) + (5𝑥(−1) + (3𝑥1) + (3𝑥(−1)) + (7𝑥1) + (8𝑥(−1)) + (1x1) + (1x(−1)) + (1x1) | -8 |
| 3 | 3𝑥1) + (7𝑥(−1) + (8𝑥1) + (1𝑥(−1)) + (1𝑥1) + (1𝑥(−1)) + (3x1) + (2x(−1)) + (6x1) | -2 |
| 4 | (5𝑥1) + (1𝑥(−1) + (2𝑥1) + (5𝑥(−1)) + (3𝑥1) + (2𝑥(−1)) + (7x1) + (8x(−1)) + (1x1) | 0 |
| 5 | (5𝑥1) + (3𝑥(−1) + (2𝑥1) + (7𝑥(−1)) + (8𝑥1) + (1𝑥(−1)) + (1x1) + (1x(−1)) + (4x1) | 0 |
| 6 | (7𝑥1) + (8𝑥(−1) + (1𝑥1) + (1𝑥(−1)) + (1𝑥1) + (4𝑥(−1)) + (2x1) + (6x(−1)) + (4x1) | -12 |
| 7 | (1𝑥1) + (2𝑥(−1) + (5𝑥1) + (3𝑥(−1)) + (2𝑥1) + (3𝑥(−1)) + (8x1) + (1x(−1)) + (2x1) | 7 |
| 8 | (3𝑥1) + (2𝑥(−1) + (3𝑥1) + (8𝑥(−1)) + (1𝑥1) + (2𝑥(−1)) + (1x1) + (4x(−1)) + (6x1) | -16 |
| 9 | (8𝑥1) + (1𝑥(−1) + (2𝑥1) + (1𝑥(−1)) + (4𝑥1) + (6𝑥(−1)) + (6x1) + (4x(−1)) + (7x1) | 1 |

Kemudian sebelum di lanjutkan ke proses *pooling* *layer*, untuk menghilangkan nilai *negative* pada hasil, pada arsitektur jaringan digunakan aktivasi ReLU (*Rectified Linear Unit*) setelah proses *convolution*. Fungsi dari aktivasi ini adalah melakukan *threshold* atau memberikan nilai batas dari 0 hingga tak terhingga. Nilai yang ada pada hasil *convolution* yang bernilai *negative* akan diubah dengan aktivasi ini menjadi nol dan yang lainnya sampai tak terhingga.

**Proses *Pooling***

*Pooling* merupakan pengurangan ukuran *matriks* dengan menggunakan operasi *pooling* (penggabungan). Metode yang digunakan dalam proses *pooling* ini menggunakan *max*-*pooling*. *Max*-*pooling* merupakan salah satu metode umum yang biasa digunakan oleh peneliti yang berkaitan dengan penelitian *deep learning*. Dalam penelitian yang dilakukan oleh Dominik Scherer dkk (Scherer, 2010) menunjukan bahwa penggunaan metode *max*-*pooling* lebih unggul dibanding dengan metode *sub-sampling*. penggunaan metode ini menjadi salah satu metode terbaik dalam proses *pooling*. Berikut ini gambaran dari proses *pooling*:

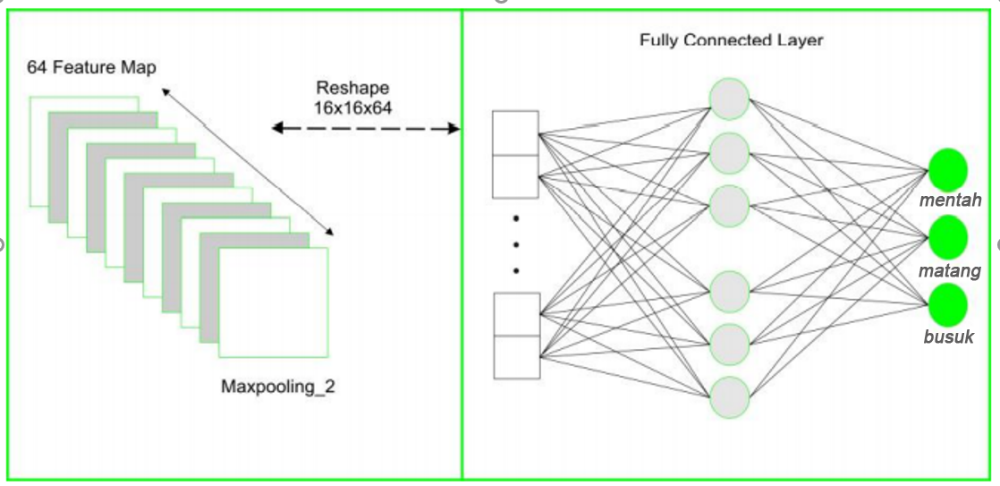


**Gambar 27.** Proses Pooling

Proses *pooling* ini menggunakan ukuran 2x2 dengan *stride* 1 dimana jumlah pergeseran *kernel* terhadap *matriks* *input* berjumlah satu. Dalam proses *pooling* ini digunakan metode *max-pooling*, dimana *window* akan bergeser sesuai dengan ukuran dan stridennya untuk mendapatkan nilai paling maksimum. Terlihat pada Gambar 27 *output* dari proses ini memiliki nilai yang paling maksimum yang di ambil dari *matriks* fitur map hasil *convolution*. Hasil *max-pooling* tersebut berukuran 2x2.

**Proses *Fully Connected***

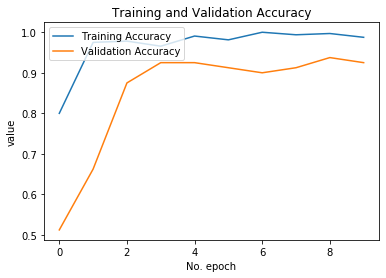
Selanjutnya adalah *Fully connected Layer*. Proses ini bertujuan untuk untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear.



**Gambar 28.** Proses Fully Connected

Gambar 28 merupakan proses *converting* hasil dari fitur map *max*-*pooling* menjadi *flatten* atau *vector*, *activation map* yang dihasilkan dari *feature extraction* layer masih berbentuk *multidimensional array*, harus melakukan *reshape activation map* menjadi sebuah vektor agar bisa digunakan sebagai *input* dari *fully-connected* *layer*. Proses ini sama dengan proses MLP (Multilayer Perceptron). Jaringan ini umumnya menggunakan lapisan yang terhubung sepenuhnya di mana setiap piksel dianggap sebagai *neuron* terpisah. Dalam proses ini biasanya diterapkan metode “*dropout*”. Metode ini bertujuan untuk menonaktifkan beberapa *edge* yang terhubung ke setiap *neuron* untuk menghindari *overfitting*. Setelah itu proses terakhir adalah klasifikasi. Dalam proses ini digunakan aktivasi fungsi *softmax*. Aktivasi ini akan membantu MLP untuk mengklasifikasikan *input* terhadap targetnya, yaitu kedalam 3 kelas, yaitu Mentah, Matang dan Busuk. Setiap *neuron* pada *convolution layer* perlu ditransformasi menjadi data satu dimensi terlebih dahulu sebelum dapat dimasukkan ke dalam sebuah *fully-connected layer*. Karena hal tersebut menyebabkan data kehilangan informasi spasialnya dan tidak *reversibel*, sedangkan *fully-connected layer* hanya dapat diimplementasikan diakhir jaringan.

**Hasil *Training***

****Setelah melalui beberapa proses dalam algoritma *Convolutional Neural Network* (CNN) didapatkan hasil *training* dan *validation*. Proses *train* akan menghasilkan sebuah model yang nantinya akan digunakan dalam proses pengujian, model yang disimpan berisi hasil training yang dilakukan oleh sistem. Secara garis besar isi dari proses *train* model yaitu ditunjukkan pada Tabel 5. Proses ini menggunakan jumlah 10 *epoch*. berikut grafik hasil *training* menggunakan *tensorflow.keras*:

**Gambar 29.** Grafik Hasil Training

Berdasarkan Gambar 29 terdapat 4 indikator, yaitu *Training* *Loss* dan *Accuracy* serta *Validation Loss* dan *Validation Accuracy*. *Training* *Loss* merupakan hasil *train* yang datanya tidak terbaca, semakin kecil nilai *loss* maka akan semakin baik, sedangkan *Training* *Accuracy* merupakan nilai akurasi yang di dapat dari proses *training*, semakin tinggi nilai akurasi maka semakin baik model yang telah dibuat. *Validation Loss* merupakan nilai pembanding dari proses *train*, semakin rendah nilai *validation loss* maka semakin baik modelnya, sedangkan *Validation Accuracy* merupakan nilai pembanding yang diperoleh dari proses *train model*, semakin tinggi nilai akurasi validasi maka semakin baik model yang telah dibuat. Dalam proses *train* kali ini didapat *accuracy* dari *training* model mencapai 0.98 dengan nilai loss sebesar 0.04. Proses *training* disini menggunakan *learning rate* 0.001 dengan *input* gambar sebesar 64 x 64 piksel. Waktu pelatihan yang dibutuhkan untuk 10 epoch dalam menjalankan *training* model ini yaitu 30 menit. Semakin Banyak *epoch* maka semakin lama juga waktu yang dibutuhkan untuk *training* model. Kemudian *accuracy* dari data *validation* mencapai 0.92 dengan nilai *loss* sebesar 0.21.

## Hasil Pengujian Model

Proses *testing* menggunakan data sebanyak 81 citra. Hasil *confusion* *matriks* adalah sebagai berikut:

**Tabel 7.** Confusion Matriks

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Matriks* | | Prediksi Kelas | | |
| busuk | matang | mentah |
| Kelas sebenarnya | busuk | 20 | 2 | 6 |
| matang | 2 | 23 | 2 |
| Mentah | 6 | 1 | 19 |

Berdasarkan Tabel 7 hasil hasil prediksi dari model terhadap data *testing* menunjukan hasil yang baik, yaitu prediksi terhadap buah sawit busuk di klasifikasikan benar sebanyak 20 dan *missing* data dari *input* sawit busuk di klasifikasikan sebagai sawit matang sebanyak 2 data serta sawit busuk yang terklasifikasi sebagai sawit mentah sebanyak 6 data. Prediksi kedua pada Buah sawit matang diklasifikasikan benar sebanyak 23 dan *missing* data dari *input* sawit matang diklasifikasikan sebagai sawit busuk sebanyak 2 data serta sawit matang yang diklasifikasikan sawit mentah sebanyak 2 data. Kemudian yang terakhir adalah prediksi pada Buah Sawit mentah diklasifikasikan benar sebagai sawit mentah sebanyak 19 dan *missing* data dari *input* sawit mentah diklasifikasikan sebagai sawit busuk sebanyak 6 data serta sawit mentah yang diklasifikasikan sebagai sawit matang sebanyak 1 data. Perhitungan akurasi dari keseluruhan *matriks* diatas adalah sebagai berikut:

Jadi akurasi yang dihasilkan oleh model dengan jumlah sampel *testing* 81 citra didapatkan nilai akurasi sebsesar 76%. Pada penggunaan *confusion matrix* tentu saja tidak hanya berpatokan pada nilai akurasi tersebut tetapi juga terdapat *precision, recall dan f1-score* yang ditunjukkan pada Gambar 30 sebagai berikut:

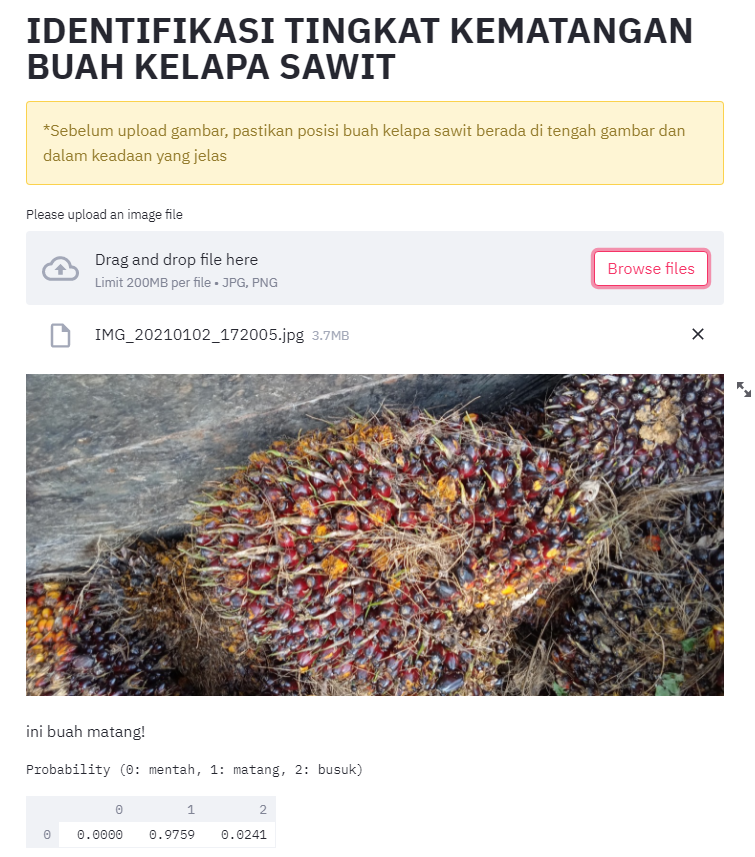


**Gambar 30.** *Precision, Recall dan F1-score*

Berdasarkan Gambar 30 diperoleh hasil akurasi dari *precision* sebesar 0.76, *precision* yang dimaksud disini merupakan tingkat ketepatan antara informasi yang diminta dengan jawaban system, selanjutnya yaitu *recall* yang merupakan tingkat keberhasilan system dalam menemukan kembali sebuah informasi, hasil akurasi *recall* adalah sebesar 0.76, sedangkan *accuracy/f1-score* merupakan tingkat kedekatan antara nilai prediksi dengan nilai actual dan hasil dari akurasi *f1-score* adalah sebesar 0.76.

## Hasil Klasifikasi Gambar

Pengujian akan dilakukan dengan menggunakan data uji sebanyak 6 gambar yang berbeda. Gambar di bedakan menjadi tiga kelas, yaitu: mentah, matang dan busuk. Sebelum dilakukan pengujian pastikan posisi citra buah kelapa sawit berada ditengah dan dalam keadaan yang jelas (tidak terlalu jauh). Pengujian dilakukan menggunakan *website*. Pengujian dilakukan menggunakan *website* yang bertujuan untuk memudahkan dalam proses identifikasi tingkat kematangan buah kelapa sawit. *Website* tersebut dibangun menggunakan bahasa pemograman python dengan bantuan *library streamlit* yang merupakan salah satu *framework website* dari *python* yang masih baru tetapi cukup populer dikalangan *data science* karena simpel dan mudah dipelajari. Hasil yang ditampilkan dalam *website* ini menggunakan model klasifikasi yang telah dibuat sebelumnya. Berikut merupakan tampilan *website* yang telah dibuat:



**Gambar 31.** Tampilan Website

Pada penelitian ini pengujian dilakukan dengan scenario membandingkan citra dengan *background* dan tanpa *background* kemudian untuk hasilnya nanti akan dibandingkan apakah perbedaan antara pengujian citra menggunakan background dengan tanpa background akan berbeda jauh. Berikut merupakan tabel hasil pengujian gambar:

**Table 8.** Hasil Pengujian

|  |  |  |  |
| --- | --- | --- | --- |
| No | Gambar | Akurasi | Kelas |
| 1 |  | 0.9372 | Matang |
| 2 |  | 0.9916 | Mentah |
| 3 |  | 0.4481 | Busuk |
| 4 |  | 0.7787 | Matang |
| 5 |  | 0.9576 | Mentah |
| 6 |  | 0.9752 | Busuk |

Dapat dilihat pada Tabel 8 proses pengujian pertama dilakukan dengan menguji buah kelapa sawit mentah dengan *background* mendapat akurasi sebesar 0.9916 dan buah kelapa sawit mentah tanpa *background* mendapat akurasi sebesar 0.9576, artinya tidak terlalu jauh perbedaannya pada pengujian buah kelapa sawit mentah. Pengujian kedua dilakukan dengan citra buah kelapa sawit matang dengan background memperoleh akurasi sebesar 0.9372 dan buah kelapa sawit matang tanpa background memperoleh akurasi sebesar 0.7787, artinya pada pengujian kedua tidak terlalu jauh perbedaanya. Pengujian ketiga dilakukan dengan menguji buah kelapa sawit busuk dengan background memperoleh akurasi sebesar 0.4481 dan buah kelapa sawit busuk tanpa background memperoleh akurasi sebesar 0.9752, artinya pada pengujian ketiga terdapat perbedaan yang cukup jauh, hal tersebut dikarenakan warna background pada citra buah sawit busuk memiliki warna yang hampir sama dengan buah sawit tersebut. Berdasarkan pengujian pada penelitian ini ternyata warna dan bentuk background dapat mempengaruhi tingkat akurasi pengujian. Dalam pengujian tidak ada *missing* data dari 6 citra yang di uji. Namun terdapat beberapa pengujian yang mendapat akurasi cukup rendah, yaitu pada pengujian citra buah busuk dengan akurasi sebesar 0.4481. Proses penghitungan akurasi merupakan proses akhir pada penelitian ini. Akurasi dalam penelitian ini merupakan variabel yang merepresentasikan kinerja yang digunakan untuk menilai tolak ukur keberhasilan model CNN dalam mengidentifikasi tingkat kematangan buah kelapa sawit. Persamaan yang digunakan untuk menghitung akurasi ditunjukkan pada persamaan berikut ini:

(Maulana dan Rochmawati, 2019).

**Tabel 9.** Margin Error Pengujian Gambar

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Kategori | Jumlah Data Uji | Salah | Benar |
| 1 | Matang | 2 | 0 | 1 |
| 2 | 0 | 1 |
| 3 | Mentah | 2 | 0 | 1 |
| 4 | 0 | 1 |
| 5 | Busuk | 2 | 0 | 1 |
| 6 | 0 | 1 |
| Total | |  | - | 6 |

Pada Tabel 9 hasil pengujian terhadap dataset yang diuji menunjukan hasil yang baik. Proses perhitungan akurasi dapat ditunjukkan pada perhitungan berikut ini:

Akurasi yang didapatkan dari proses pengujian yang menggunakan 6 data uji menunjukkan nilai akurasi 100%.

# V. KESIMPULAN DAN SARAN

## Kesimpulan

Adapun kesimpulan yang diberikan pada penelitian ini sebagai berikut:

1. Dari hasil penelitian dan pembahasan didapatkan hasil bahwa identifikasi tingkat kematangan buah kelapa sawit menggunakan algoritma *Convolutional Neural Network* (CNN) mendapatkan nilai akurasi yang tinggi yaitu sebesar 98% pada proses *training* dan 76% pada proses pengujian model.
2. Kemudian penelitian ini menggunakan data baru acak untuk menguji model yang dibuat dan menghasilkan nilai akurasi yang baik dalam mengidentifikasi buah kelapa sawit. Sehingga dapat disimpulkan bahwa model yang telah dibuat dengan mengimplementasikan metode *deep learning* menggunakan algoritma *Convolutional Neural Network* (CNN) mampu melakukan identifikasi dan klasifikasi buah kelapa sawit.

## Saran

Adapun saran yang diberikan pada penelitian ini sebagai berikut:

1. Penelitian selanjutnya diharapkan dapat menambah jumlah dataset buah kelapa sawit sesuai kelas yang telah ditentukan.
2. Menambahkan parameter seperti perbandingan *input* gambar yang lebih besar ukuran pikselnya, variasi jumlah *epoch* dan *learning rate*, serta penggunaan *optimizer* yang lainnya. Sehingga dengan menghasilkan model dengan penggunaan hyperparameter terbaik.
3. Penelitian ini dapat di kembangkan kedalam robotika atau alat yang dapat mengenali gambar tersebut dan dapat di impelementasikan ke perusahaan atau petani.

# DAFTAR PUSTAKA

Astuti, D. L. Z., Samsuryadi, S., & Rini, D. P. (2019). Real-Time Classification of Facial Expressions Using a Principal Component Analysis and Convolutional Neural Network. *Sinergi*, *23*(3), 239. https://doi.org/10.22441/sinergi.2019.3.008.

Badan Pusat Statistik Nasional, 2020. Statistik Perkebunan Kelapa Sawit Tahun 2020. Jakarta: Badan Pusat Statistik.

Danukusumo, K.P. (2017). Implementasi Deep Learning Menggunakan Convolutional Neural Network Untuk Klasifikasi Citra Candi Berbasis Gpu.

Deng, L. and Yu, D. (2014) Deep Learning: Methods and Applications. Foundations and Trends in Signal Processing, 7, 197-387.

Direktorat Jenderal Perkebunan. 2020. Statistik Perkebunan Indonesia Kelapa Sawit Indonesia 2018-2020. Direktorat Jenderal Perkebunan. Jakarta.

Dzulqarnain, M.F, Suprapto and Makhrus, F. 2019. Improvement of Convolutional Neural Network Accuracy on Salak Classification Based Quality on Digital Image. Indonesian Journal of Computing and Cybernetics Systems Vol.13, No.2, April 2019, pp. 189~198 ISSN (print): 1978-1520, ISSN (online): 2460-7258 DOI: 10.22146/ijccs.42036.

Fadhilah, H., Djamal, E. C., Ilyas, R., & Najmurrokhman, A. (2019). Non-Halal Ingredients Detection of Food Packaging Image Using Convolutional Neural Networks. *Proceeding - 2018 International Symposium on Advanced Intelligent Informatics: Revolutionize Intelligent Informatics Spectrum for Humanity, SAIN 2018*, 131–136. https://doi.org/10.1109/SAIN.2018.8673376.

Fawcett, Tom. (2006). Introduction to ROC analysis. Pattern Recognition Letters. 27. 861-874. 10.1016/j.patrec.2005.10.010.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Hadinisa, S., Koyimatu, M., Irawan, A., Nugroho, H., Penelitian, M., Dasar, T., & Learning, M. (2018). Analisis Learning Rate pada Metode Transfer Learning untuk Sistem Pendeteksi Api. *Seminar Nasional Microwave, Antena Dan Propagasi (SMAP) 2018 Unpak*, 8–11.

Han, J. and Kamber, M. (2006) Data Mining: Concepts and Techniques. 2nd Edition, Morgan Kaufmann Publishers, San Francisco.

Khan, S., Rahmani, H., Ali Shah, S. A., & Bennamoun, M. (2018). A Guide to Convolutional Neural Networks for Computer Vision. Synthesis Lectures on Computer Vision, 8(1).

Krizhevsky, A., Sutskever, I. dan Hinton, G.E., 2012, ImageNet Classification with Deep Convolutional Neural Networks, Advances In Neural Information Processing Systems, pp.1–9.

Kusumadewi, S., 2004, Membangun Jaringan Syaraf Tiruan, Yogyakarta: Graha

Ilmu.

Labach, Alex & Salehinejad, Hojjat & Valaee, Shahrokh. (2019). Survey of Dropout Methods for Deep Neural Networks.

LeCun, Y., Kavukcuoglu, K., & Farabet, C. (2010). Convolutional networks and applications in vision. *ISCAS 2010 - 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, 253–256. https://doi.org/10.1109/ISCAS.2010.5537907.

Lau, M.M., Lim, K.H. dan Gopalai, A.A., 2015. Malaysia Traffic Sign Recognition with Convolutional Neural Network. 2015 IEEE International Conference on Digital Signal Processing (DSP), IEEE (978-1-4799-8058-1/15), 1006–1010.

Maulana, F. F., & Rochmawati, N. (2019). Klasifikasi Citra Buah Menggunakan Convolutional Neural Network. *Journal of Informatics and Computer Science*, *01*, 104–108. jurnalmahasiswa.unesa.ac.id › article.

Minarni, Salumbae, R. & Hasbi, Z. (2018). *Komunikasi Fisika Indonesia*. *15*(01), 36–45.

Nugroho, A. (2019). *Buku Teknologi Agroindustri Kelapa Sawit* (Issue August).

Pahan, I. (2015). *Panduan Teknis Budidaya Kelapa Sawit Untuk Praktisi Perkebunan*, Jakarta:Penebar Swadaya.

Prasetyo, E., 2012. Data Mining Konsep dan Aplikasi Menggunakan Matlab, Andi Offset, Yogyakarta.

Prathivi, R. (2019). *Optimasi Model TL-CNN Untuk Klasifikasi Citra CIFAR-10*. *1*(10), 3–7.

Putra, I. S., Wijaya, A. Y., & Soelaiman, R. (2016). Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) pada Caltech 101. JURNAL TEKNIK ITS, 5(1).

Rangkuti, I.U.P. dan A. Syahputra. (2019). Warna Minyak Sawit Mentah dan Stabilitas Warna Berdasarkan Tingkat Kematangan Buah yang Berasal dari Kebun dengan ketinggian 800 mdpl. Jurnal Agro Fabrica vol. 1 (2): 32-37.

W. Rawat and Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review,” Neural Computation, vol.29, no.9, pp. 2352- 2449, September 2017. Available: https://ieeexplore.ieee.org/document/8016501

Razak, T. R. B., Othman, M. B., Nazari, M., Adilah, K., & Mansor, A. R. (2018). *Analysis*.

Rismiyati, & Azhari, S. N. (2017). Convolutional Neural Network implementation for image-based Salak sortation. *Proceedings - 2016 2nd International Conference on Science and Technology-Computer, ICST 2016*, 77–82. https://doi.org/10.1109/ICSTC.2016.7877351.

Sari, N., Shiddiq, M., Fitra, R. H., & Yasmin, N. Z. (2019). Ripeness Classification of Oil Palm Fresh Fruit Bunch Using an Optical Probe. *Journal of Aceh Physics Society*, *8*(3), 72–77. https://doi.org/10.24815/jacps.v8i3.14122.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15(56), 1929-1958.

Trnovszky, T., Kamencay, P., Orjesek, R., Benco, M., & Sykora, P. (2018). Animal recognition system based on convolutional neural network. *Advances in Electrical and Electronic Engineering*, *15*(3), 517–525. https://doi.org/10.15598/aeee.v15i3.2202.

Wang, J., & Perez, L. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning.

Zhu, Qiuyu & He, Zikuang & Zhang, Tao & Cui, Wennan. (2020). Improving Classification Performance of Softmax Loss Function Based on Scalable Batch-Normalization. Applied Sciences. 10. 2950. 10.3390/app10082950.

# LAMPIRAN

**Lampiran 1**

import numpy as np

import pandas as pd

from tensorflow.keras.preprocessing.image import ImageDataGenerator,load\_img

from tensorflow.keras.utils import to\_categorical

from sklearn.model\_selection import train\_test\_split

import matplotlib.pyplot as plt

import random

import os

Image\_Width=128

Image\_Height=128

Image\_Size=(Image\_Width,Image\_Height)

Image\_Channels=3

filenames=os.listdir("sawit/train")

categories=[]

for f\_name in filenames:

category=f\_name.split('.')[0]

if category=='matang':

categories.append(1)

elif category=='mentah':

categories.append(2)

else:

categories.append(0)

df=pd.DataFrame({

'filename':filenames,

'category':categories

})

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D,MaxPooling2D,\

Dropout,Flatten,Dense,Activation,\

BatchNormalization

model=Sequential()

model.add(Conv2D(128,(3,3),activation='relu',input\_shape=(Image\_Width,Image\_Height,Image\_Channels)))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.25))

model.add(Conv2D(64,(3,3),activation='relu'))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.25))

model.add(Conv2D(32,(3,3),activation='relu'))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(512,activation='relu'))

model.add(BatchNormalization())

model.add(Dropout(0.5))

model.add(Dense(3,activation='softmax'))

model.compile(loss='categorical\_crossentropy',

optimizer='rmsprop',metrics=['accuracy'])

from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau

earlystop = EarlyStopping(patience = 10)

learning\_rate\_reduction = ReduceLROnPlateau(monitor = 'val\_acc',patience = 2,verbose = 1,factor = 0.5,min\_lr = 0.00001)

callbacks = [earlystop,learning\_rate\_reduction]

df["category"] = df["category"].replace({0:'busuk',1:'matang',2:'mentah'})

train\_df,validate\_df = train\_test\_split(df,test\_size=0.20,

random\_state=42)

train\_df = train\_df.reset\_index(drop=True)

validate\_df = validate\_df.reset\_index(drop=True)

total\_train=train\_df.shape[0]

total\_validate=validate\_df.shape[0]

batch\_size=15

train\_datagen = ImageDataGenerator(rotation\_range=15,

rescale=1./255,

shear\_range=0.1,

zoom\_range=0.2,

horizontal\_flip=True,

width\_shift\_range=0.1,

height\_shift\_range=0.1

)

train\_generator = train\_datagen.flow\_from\_dataframe(train\_df,

"sawit/train",x\_col='filename',y\_col='category',

target\_size=Image\_Size,

class\_mode='categorical',

batch\_size=batch\_size)

validation\_datagen = ImageDataGenerator(rescale=1./255)

validation\_generator = validation\_datagen.flow\_from\_dataframe(

validate\_df,

"sawit/train",

x\_col='filename',

y\_col='category',

target\_size=Image\_Size,

class\_mode='categorical',

batch\_size=batch\_size

)

test\_datagen = ImageDataGenerator(rotation\_range=15,

rescale=1./255,

shear\_range=0.1,

zoom\_range=0.2,

horizontal\_flip=True,

width\_shift\_range=0.1,

height\_shift\_range=0.1)

test\_generator = train\_datagen.flow\_from\_dataframe(train\_df,

"sawit/test",x\_col='filename',y\_col='category',

target\_size=Image\_Size,

class\_mode='categorical',

batch\_size=batch\_size)

EPOCH = 50

history = model.fit(x=train\_generator,

steps\_per\_epoch=len(train\_generator),

epochs=EPOCH,

validation\_data=validation\_generator,

validation\_steps=len(validation\_generator),

shuffle=True,

verbose =1)

model.save("sawit5.h5")

from matplotlib import pyplot as plt

# Plot history: MAE loss

plt.plot(history.history['loss'], label='Training Loss')

plt.plot(history.history['val\_loss'], label='Validation Loss')

plt.title('Training and Validation Loss')

plt.ylabel('value')

plt.xlabel('No. epoch')

plt.legend(loc="upper left")

plt.show()

# Plot history: MSE acc

plt.plot(history.history['accuracy'], label='Training Accuracy')

plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')

plt.title('Training and Validation Accuracy')

plt.ylabel('value')

plt.xlabel('No. epoch')

plt.legend(loc="upper left")

plt.show()

from sklearn.metrics import classification\_report, confusion\_matrix

Y\_pred = model.predict\_generator(test\_generator)

y\_pred = np.argmax(Y\_pred, axis=1)

print('Confusion Matrix')

print(confusion\_matrix(test\_generator.classes, y\_pred))

print('Classification Report')

target\_names = ['mentah','matang']

print(classification\_report(test\_generator.classes,

y\_pred, target\_names=target\_names))

from sklearn.metrics import accuracy\_score

score = accuracy\_score(test\_generator.classes, y\_pred)

print("Accuracy:",score)

**Lampiran 2**

