

**IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORKS* UNTUK
KLASIFIKASI CITRA TOMAT MENGGUNAKAN *KERAS***

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana
Program Studi Statistika



Tiara Shafira

14 611 099

**JURUSAN STATISTIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2018**

HALAMAN PERSETUJUAN PEMBIMBING

TUGAS AKHIR

Judul : Implementasi *Convolutional Neural Networks* untuk
Klasifikasi Citra Tomat Menggunakan Keras

Nama Mahasiswa : Tiara Shafira

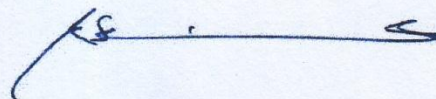
Nomor Mahasiswa : 14 611 099

TUGAS AKHIR INI TELAH DIPERIKSA DAN DISETUJUI UNTUK
DIUJIKAN

Yogyakarta, 13 Februari 2018

الجامعة الإسلامية الاندونيسية

Pembimbing

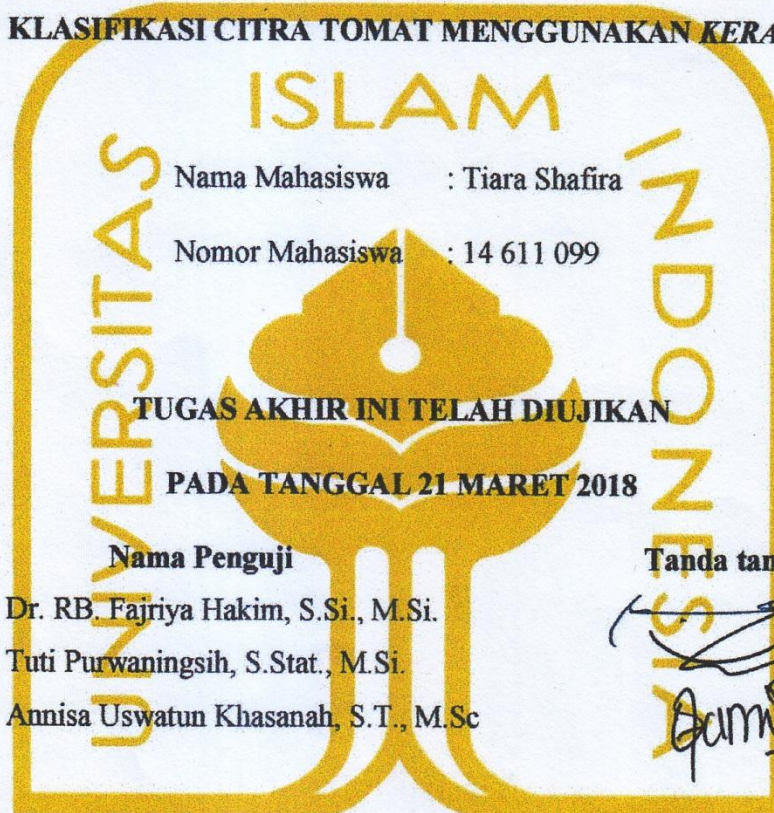


(Dr. RB. Fajriya Hakim, S.Si., M.Si.)

HALAMAN PENGESAHAN

TUGAS AKHIR

IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORKS* UNTUK KLASIFIKASI CITRA TOMAT MENGGUNAKAN *KERAS*



Nama Mahasiswa : Tiara Shafira

Nomor Mahasiswa : 14 611 099

TUGAS AKHIR INI TELAH DIUJIKAN

PADA TANGGAL 21 MARET 2018

Nama Penguji

1. Dr. RB. Fajriya Hakim, S.Si., M.Si.
2. Tuti Purwaningsih, S.Stat., M.Si.
3. Annisa Uswatun Khasanah, S.T., M.Sc

Tanda tangan

Mengetahui,

Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam



(Drs. Allwar, M.Sc., Ph.D.)

KATA PENGANTAR



Assalamu'alaikum Warahmatullaahi Wabarakaatuh

Alhamdulillah rabbil 'aalamiin, puji syukur senantiasa penulis panjatkan atas kehadiran Allah SWT yang telah melimpahkan rahmat dan hidayah yang tak terhingga, sehingga penulis dapat menyelesaikan tugas akhir yang berjudul “Implementasi *Convolutional Neural Networks* Untuk Klasifikasi Citra Tomat Menggunakan *Keras*” sebagai salah satu persyaratan yang harus dipenuhi dalam menyelesaikan jenjang strata satu di Jurusan Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia. Shalawat serta salam *Insyallah* selalu tercurah kepada junjungan umat, yaitu Nabi Muhammad SAW sebagai suri tauladan serta para sahabat dan pengikutnya yang senantiasa berjuang untuk agama Islam hingga akhir.

Penyelesaian tugas akhir ini tidak terlepas dari dukungan, bantuan, arahan, dan bimbingan dari berbagai pihak. Untuk itu pada kesempatan ini penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT yang senantiasa memberikan kesempatan dan kekuatan sehingga penulis dapat melaksanakan penelitian hingga menyusun Tugas Akhir dengan baik.
2. Nabi Muhammad SAW sebagai salah satu hamba terbaik dan motivator penulis, semoga kita mendapatkan syafa'at beliau di akhirat kelak. Aamiin.
3. Kedua orang tua yaitu Mama dan Papa yang selalu menanyakan perkembangan tugas akhir serta memberikan semangat, do'a dan dukungan di setiap langkah penulis.
4. Adik-adik, yaitu Irnanti Wahyurini dan M. Fairuz Nadhir Amrullah, serta keluarga lainnya yang selalu mendo'akan dan memberikan semangat kepada penulis.

5. Drs. Allwar. M.Sc, Ph.D. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam.
6. Bapak Dr. RB. Fajriya Hakim, M.Si. selaku Ketua Program Studi Statistika sekaligus sebagai dosen pembimbing, yang telah banyak meluangkan waktu, memberikan masukan yang membangun serta arahan yang positif saat melakukan bimbingan.
7. Seluruh staf pengajar Program Studi Statistika Universitas Islam Indonesia yang telah memberikan bekal ilmu kepada penulis, sehingga penulis dapat menyelesaikan tugas akhir ini.
8. Keluarga besar Humas IKS UII, KAMMI UII, Formasi Yogyakarta, serta PIK-M Aushaf UII, terimakasih atas kebersamaan, kekeluargaan, kekompakan, keceriaan dan pelajaran berharga lainnya.
9. Sahabat seperjuangan “MMR” yaitu Hani Rahayu Budi Utami, Indah Dewi Fitriani, Nurul Imani, dan Denisha Intan Perihatini yang telah berjuang bersama selama di bangku perkuliahan dan senantiasa menjadi pengingat dalam hal kebaikan. Semoga ukhuwah kita tetap terjaga dan selalu diridhoi Allah SWT.
10. Teman-teman seperjuangan “Project 2 Minggu Selesai!” yaitu Jimmy Pujoseno dan Sendhyka Cakra Pradana yang selalu mau untuk berjuang, belajar, dan berlari bersama dalam menyelesaikan tugas akhir ini.
11. Teman-teman bimbingan tugas akhir lainnya, yang senantiasa berbagi ilmu dan informasi yang bermanfaat sampai tugas akhir ini selesai.
12. Teman-teman “Akal 07” SMAN 1 BOLO, yang selalu mengingatkan dan saling menyemangati untuk menyelesaikan studi di perkuliahan.
13. Teman-teman Statistika UII Angkatan 2014 yang sedang berjuang bersama untuk meraih gelar S.Stat dan Toga UII, terimakasih atas pengalaman berharga selama menjadi mahasiswa Statistika UII.
14. Pihak-pihak lain yang tidak penulis sebutkan satu per satu, yang telah banyak memotivasi dan meningkatkan kembali semangat penulis dalam menyusun tugas akhir.

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 laporan 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.

Wassalamu’alaikum Warahmatullaahi Wabarakaatuh

Yogyakarta, 29 Maret 2018

Penulis

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PERSETUJUAN PEMBIMBING	ii
HALAMAN PENGESAHAN.....	iii
KATA PENGANTAR.....	iv
DAFTAR ISI.....	vii
DAFTAR TABEL	x
DAFTAR GAMBAR.....	xi
DAFTAR LAMPIRAN	xii
PERNYATAAN.....	xiii
INTISARI	xiv
ABSTRACT	xvi
BAB I PENDAHULUAN.....	1
1.1. Latar Belakang Masalah.....	1
1.2. Rumusan Masalah.....	4
1.3. Batasan Masalah	5
1.4. Tujuan Penelitian	5
1.5. Manfaat Penelitian	5
1.6. Sistematika Penulisan	6
BAB II TINJAUAN PUSTAKA	7
BAB III LANDASAN TEORI.....	16
3.1. Tomat	16
3.2. Kecerdasan Buatan.....	16
3.3. Jaringan Syaraf Tiruan	18
3.3.1. Konsep Jaringan Syaraf Tiruan.....	18
3.3.2. Komponen Jaringan Syaraf	21
3.3.3. Arsitektur Jaringan.....	21
3.4. Fungsi Aktivasi	21
3.5. <i>Deep Learning</i>	21
3.6. Citra.....	22

3.7. Citra Warna RGB.....	23
3.8. Histogram Citra.....	23
3.9. Pengolahan Citra.....	24
3.10. <i>Convolutional Neural Networks</i>	25
3.11. Operasi Konvolusi	25
3.12. Stride	26
3.13. <i>Zero Padding</i>	26
3.14. <i>Pooling Layer</i>	27
3.15. Ilustrasi Proses pada Lapisan Konvolusi	28
3.16. <i>Fully-Connected Layer</i>	29
3.17. <i>Dropout Regularization</i>	29
3.18. <i>Softmax Classifier</i>	30
3.19. <i>Crossentropy Loss Function</i>	31
3.20. <i>Stochastic Gradien Descent</i>	31
3.21. <i>Max Norm Constrain</i>	32
3.22. <i>Confusion Matrix</i>	32
3.23. Akurasi, <i>Precission</i> , dan <i>Recall</i>	33
3.24. RStudio	34
3.25. Keras	34
BAB IV METODOLOGI PENELITIAN	35
4.1. Populasi dan Sampel.....	35
4.2. Variabel dan Definisi Operasional Variabel	35
4.3. Jenis dan Sumber Data.....	35
4.4. Metode Analisis Data.....	35
4.5. Tahapan Penelitian.....	36
BAB V ANALISIS DAN PEMBAHASAN	37
5.1. Pengumpulan Data Citra	37
5.2. Histogram Citra.....	38
5.3. <i>Preprocessing</i> Citra	41
5.4. Pengolahan Citra.....	42
5.5. Penentuan Parameter CNN	53

BAB VI PENUTUP	52
6.1. Kesimpulan	56
6.2. Saran	57
DAFTAR PUSTAKA	58
LAMPIRAN.....	63

DAFTAR TABEL

Tabel 2.1	Perbandingan dengan penelitian terdahulu.....	11
Tabel 4.1	Definisi operasional variabel	35
Tabel 5.1	Hasil <i>Loss</i> dan <i>Accuracy</i> Data <i>Training</i> dan <i>Testing</i>	49
Tabel 5.2	Hasil Klasifikasi Data <i>Training</i>	51
Tabel 5.3	Hasil Klasifikasi Data <i>Testing</i>	51
Tabel 5.4	Hasil Klasifikasi Data Uji.....	52
Tabel 5.5	Penentuan Ukuran Filter	53
Tabel 5.6	Penentuan Jumlah Neuron pada Lapisan Tersembunyi.....	54
Tabel 5.7	Pengaruh Jumlah Data Terhadap Akurasi	55
Tabel 5.8	Skenario Pelatihan dan Pengujian Data.....	55

DAFTAR GAMBAR

Gambar 3.1	Bagian Utama dalam <i>Artificial Intelligence (AI)</i>	17
Gambar 3.2	Susunan Syaraf Manusia	18
Gambar 3.3	Struktur Neuron Jaringan Syaraf.....	19
Gambar 3.4	Contoh Jaringan Syaraf dengan Tiga Lapisan.....	20
Gambar 3.5	Proses Digitalisasi Citra	23
Gambar 3.6	Contoh Jaringan CNN	25
Gambar 3.7	Contoh Operasi <i>Max Pooling</i>	27
Gambar 3.8	Ilustrasi Proses Konvolusi	29
Gambar 3.9	Contoh Implementasi <i>Dropout Regularization</i>	30
Gambar 3.10	<i>Confusion Matrix</i>	33
Gambar 4.1	Tahapan Penelitian	36
Gambar 5.1	Perangkat yang Digunakan.....	37
Gambar 5.2	Citra Tomat Layak.....	38
Gambar 5.3	Citra Tomat Tidak Layak	38
Gambar 5.4	Citra Tomat Layak.....	39
Gambar 5.5	Informasi Citra	39
Gambar 5.6	Histogram Citra	40
Gambar 5.7	Tomat Tidak Layak dan Histogramnya.....	40
Gambar 5.8	Contoh Hasil <i>Cropping</i> Citra Tomat Layak	41
Gambar 5.9	Pengaktifan <i>Package Keras</i> dan <i>EBImage</i>	42
Gambar 5.10	Dataset Citra	43
Gambar 5.11	Pemanggilan dan Penyimpanan Citra.....	43
Gambar 5.12	Pengubahan Ukuran Citra	43
Gambar 5.13	Penampilan Dataset	44
Gambar 5.14	Proses <i>Preprocessing</i> Citra	45
Gambar 5.15	Proses Pelabelan Data Citra	45
Gambar 5.16	Model CNN	46
Gambar 5.17	<i>Summary</i> Model CNN	48
Gambar 5.18	<i>Plot Loss</i> dan <i>Accuracy</i> Data.....	49
Gambar 5.19	Probabilitas Data Uji	52

DAFTAR LAMPIRAN

Lampiran 1	<i>Script R untuk Klasifikasi Menggunakan CNN</i>	63
Lampiran 2	Probabilitas Citra pada Data <i>Training</i>	68
Lampiran 3	Probabilitas Citra pada Data <i>Testing</i>	70

PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Tugas Akhir ini tidak terdapat karya yang sebelumnya pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, 29 Maret 2018

Tiara Shafira

IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORKS* UNTUK KLASIFIKASI CITRA TOMAT MENGGUNAKAN *KERAS*

Tiara Shafira

Program Studi Statistika Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Islam Indonesia

INTISARI

Indonesia merupakan salah satu negara dengan potensi pertanian yang luar biasa. Salah satu hasil pertanian di Indonesia yaitu tomat. Berdasarkan data dari Badan Pusat Statistik (BPS) tahun 2012 menyebutkan bahwa tomat merupakan komoditas hortikultura dengan laju produktivitas menempati posisi kedua setelah bawang merah. Tingginya laju produksi dikarenakan tingginya kebutuhan masyarakat baik nasional maupun internasional seiring adanya tuntutan terhadap kualitas tomat yang terjamin. Proses pemilihan produk hasil pertanian dan perkebunan umumnya sangat bergantung pada persepsi manusia. Cara manual dilakukan berdasarkan pengamatan visual secara langsung pada objek yang akan diklasifikasi. Identifikasi dengan cara ini memiliki beberapa kelemahan, di antaranya adalah adanya keterbatasan visual manusia, serta sangat dipengaruhi oleh kondisi psikis pengamatnya. Hal tersebut juga bisa mengakibatkan tidak konsisten dalam proses pemilihannya. Tentu saja cara manual yang dilakukan terlalu banyak memakan waktu terutama bagi perkebunan besar. Perkembangan teknologi dan ilmu pengetahuan sehingga memungkinkan untuk dapat melakukan klasifikasi atau dalam hal pemilihan objek menggunakan teknologi berdasarkan karakteristik yang ditentukan berbasis citra digital. Citra digunakan sebagai sumber informasi yang dapat digunakan untuk melakukan klasifikasi objek. Salah satu metode *Deep Learning* yang efektif untuk digunakan yaitu *Convolutional Neural Networks (CNN)* dikarenakan kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra. Jaringan pada CNN mempunyai lapisan khusus yang disebut dengan lapisan konvolusi. Proses konvolusi citra pada penelitian ini menggunakan *package Keras* pada software RStudio versi 1.1.383, dikarenakan pembuatan model jaringan syaraf menggunakan Keras tidak perlu menuliskan kode untuk mengekspresikan perhitungan matematisnya satu persatu. Pengujian dengan sampel 100 citra tomat menunjukkan tingkat akurasi sebesar 90% yang dinilai telah mampu melakukan identifikasi kelayakan buah tomat.

Kata Kunci : Tomat, Citra, *Deep Learning*, *Convolutional Neural Networks (CNN)*, *Keras*.

CONVOLUTIONAL NEURAL NETWORKS IMPLEMENTATION FOR TOMATO CLASSIFICATION USING KERAS

Tiara Shafira

Departement of Statistics, Faculty of Mathematic and Natural Science
Islamic University of Indonesia

ABSTRACT

Indonesia is one of the countries with agricultural potential. One of agricultural product is tomatoes. Based on data from the Central Bureau of statistics (BPS) in 2012 says that the tomato is a commodity with the second rank of productivity rate after the onion. This high production rate due to the high society needs both national and international society, over claims to the quality of tomatoes is assured. Agricultural product election process generally based on every human opinion. The process of classification performed based on visual observation directly on the object that will be classified. There are several weakness with this manually process, among which was the existence of human visual limitations, as well as greatly influenced by human physic condition. It can also result in inconsistent in election process. The development of science and technology technology making it possible to perform the classification or in terms of object selection using technology based on characteristics that are defined based digital image. The image is used as the source of information that can be used to perform classification of objects. One of Deep Learning method that is effective to use is Convolutional Neural Networks (CNN) due to high network depth and a lot of image data was applied. Network on CNN has a special coating which is called with the convolution layer. Convolution process in this research using Keras package on RStudio software with 1.1.383 version, because we do not need to write code to express its mathematical calculation of one by one. Testing with 100 samples indicates the level of accuracy about 90%. It means this classification have been able to identify the wothiness of tomato fruit.

Keywords : *Tomatoes, Deep Learning, Convolutional Neural Networks (CNN), Keras.*

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Industri pertanian saat ini berkembang dengan pesat. Indonesia merupakan salah satu negara yang memiliki potensi besar dalam bidang pertanian. Iklimnya yang tropis dengan curah hujan yang tinggi sepanjang tahun serta tanah yang subur, memungkinkan tumbuhnya berbagai jenis tanaman. Salah satu dari sekian banyak jenis sayuran potensial yang tumbuh di Indonesia ialah tomat (Hidayatullah, 2013). Tomat memiliki banyak manfaat, yaitu antara lain adalah sebagai bahan makanan sehari-hari, sebagai sayuran, bumbu masak, buah meja, bahan pewarna, kosmetik, serta bahan baku pengolahan makanan seperti saus, sari buah, dan lain-lain. Oleh sebab itu, tomat menjadi salah satu sayuran yang multiguna sehingga memiliki nilai ekonomi yang tinggi (Nurhayati, 2017). Tomat kaya akan nutrisi dan sumber vitamin A yang sangat bermanfaat untuk kesehatan mata. Selain itu, tomat juga mengandung zat antioksidan cukup tinggi yang membantu untuk meningkatkan kekebalan tubuh, menghaluskan dan mencerahkan kulit, mencegah hipertensi, dan lainnya.

Kebutuhan tomat di Indonesia terbilang tinggi yang dicerminkan dari angka produksi tomat. Berdasarkan data dari Badan Pusat Statistik (BPS) tahun 2012 menyebutkan bahwa tomat merupakan komoditas hortikultura dengan laju produktivitas menempati posisi kedua setelah bawang merah. Tomat dengan kontribusi produksi sebesar 915.987 ton atau sekitar 7,69% terhadap produksi sayuran nasional berada pada urutan kelima. Sentra produksi tomat di Indonesia adalah Pulau Jawa dengan total produksi sebesar 434.202 ton atau sekitar 47,40% dari total produksi tomat nasional. Adapun provinsi penghasil tomat terbesar adalah Jawa Barat dengan produksi sebesar 304.687 ton 33,26% dari total produksi tomat nasional, diikuti Jawa Timur dan Jawa Tengah. Sedangkan provinsi penghasil tomat terbesar di luar Jawa adalah Sumatera Utara, dengan produksi sebesar 84.339 ton atau sekitar 9,21% dari total produksi tomat nasional, diikuti oleh Sumatera Barat (Kementerian Pertanian, 2015).

Peningkatan kebutuhan akan tomat tidak hanya dirasakan oleh masyarakat nasional tetapi juga internasional. Hal ini memicu petani untuk lebih memaksimalkan produksi tomat. Permintaan kebutuhan tomat yang setiap hari mengharuskan terjaminnya ketersediaan baik itu dari segi kuantitas maupun kualitas. Kebutuhan akan buah tomat segar tidak hanya dimiliki oleh Indonesia, namun juga oleh negara lainnya. Kondisi ini memberikan peluang untuk terjadinya ekspor tomat segar. Kendala yang sering dihadapi yaitu kurang tersedianya tomat varietas unggul yang mempunyai produksi tinggi, buah berkualitas baik, dan tahan terhadap hama dan penyakit. Meskipun tomat merupakan hasil pertanian yang bernilai gizi tinggi, tetapi tomat juga mempunyai sifat-sifat yang kurang menguntungkan, antara lain adalah mudah mengalami penurunan kualitas yang dipengaruhi oleh faktor lingkungan seperti kelembaban, temperatur, dan mutu awal tomat itu sendiri, mudah busuk dan tidak dapat menahan tekanan mekanis yang terlalu besar (Ferdiansyah, 2006).

Proses dalam pemilihan tomat yang dilakukan berbagai perusahaan pengolah maupun pengeksport tomat serta para petani yang menanam tomat pada umumnya hanya dilakukan secara manual dan melibatkan manusia sebagai pengambil keputusan. Proses pengidentifikasian seperti di atas 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.

Maka untuk memeriksa kualitas tomat, layak atau tidaknya didistribusikan ke pasar tradisional, supermarket, industri, ataupun untuk kepentingan ekspor, hingga pada akhirnya dikonsumsi atau digunakan untuk keperluan lainnya, perlu adanya suatu sistem yang dapat mengenali kualitas buah sesuai dua kategori yaitu layak atau tidak layak dan melakukan pemilihannya secara otomatis dengan

automated sorting system dengan memanfaatkan citra tomat (Anggriawan, 2017). Sistem yang dibangun diharapkan dapat menjadi solusi untuk mengenali kelayakan tomat dan dapat diterapkan.

Perkembangan ilmu pengetahuan dan teknologi pengolahan citra digital memungkinkan untuk memilah produk pertanian dan perkebunan tersebut secara otomatis (Kusumaningtyas, 2016). Pengolahan citra merupakan alternatif 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. Kedepan, pengolahan citra ini diharapkan akan menjadi salah satu pilihan dalam pemilihan kelayakan tomat tanpa merusak sampel (objek). Penggunaan teknologi ini, kualitas fisik tomat dapat ditentukan dengan cepat, murah, dengan tingkat ketelitian yang dapat dipercaya (Bustomi, 2014).

Ada banyak metode yang bisa digunakan untuk melakukan pengolahan citra, salah satunya adalah menggunakan teknik *Deep Learning (DL)* yaitu metode *Convolutional Neural Network (CNN)* yang saat ini memiliki hasil paling signifikan dalam pengenalan citra adalah CNN. Hal tersebut dikarenakan CNN berusaha meniru sistem pengenalan citra pada visual cortex manusia, sehingga memiliki kemampuan mengolah informasi citra (Suartika, 2016). Beberapa penelitian mengenai pengolahan citra dengan menggunakan metode CNN mendapatkan hasil akurasi yang bagus, yaitu penelitian yang dilakukan oleh Muhammad Zufar dan Budi Setiyono (2016) untuk pengenalan wajah secara *real-time*. Hasil akurasi yang didapatkan yaitu sebesar 89%. Penelitian terbaru juga dilakukan oleh Kevin Pudi Danukusumo (2017) untuk klasifikasi citra candi berbasis GPU. Hasil pengujian yang optimal terhadap citra candi menunjukkan akurasi sebesar 98,99% pada training set dan 85,57% pada test set dengan waktu pelatihan mencapai 389,14 detik. Penelitian tersebut menyebutkan bahwa teknik *Deep Learning* dengan CNN mampu melakukan klasifikasi citra candi dengan sangat baik.

Deep Learning merupakan salah satu sub bidang dari *Machine Learning* dimana algoritma yang dipakai terinspirasi dari bagaimana otak manusia bekerja. Beberapa orang mungkin lebih mengenalnya dengan jaringan syaraf tiruan. Pada dasarnya *Deep Learning* adalah implementasi konsep dasar *Machine Learning* yang mengadaptasikan algoritma jaringan syaraf tiruan dengan lapisan yang lebih banyak. Lebih banyak lapisan tersembunyi yang digunakan antara lapisan masukan dan lapisan keluaran, maka dapat dibilang jaringan tersebut adalah *deep neural net*. Dalam beberapa tahun terakhir *Deep Learning* telah menunjukkan performa yang luar biasa. Hal ini sebagian besar dipengaruhi faktor komputasi yang lebih kuat, dataset yang besar dan teknik untuk melatih jaringan yang lebih dalam.

Metode *Convolutional Neural Network* (CNN) merupakan salah satu metode *Deep Learning* yang sedang berkembang saat ini. Jaringan pada CNN dibuat dengan asumsi bahwa masukan yang digunakan adalah berupa gambar. Jaringan ini memiliki lapisan khusus yang dinamakan dengan lapisan konvolusi, dimana pada lapisan ini sebuah citra masukan akan diolah berdasarkan filter yang sudah ditentukan. Dari setiap lapisan ini akan menghasilkan sebuah pola dari beberapa bagian citra yang nantinya akan lebih mudah untuk diklasifikasi, sehingga membuat fungsi pembejalaran lebih efisien untuk diimplementasikan. Proses konvolusi citra pada penelitian ini menggunakan *package Keras* pada *software RStudio* versi 1.1.383, dikarenakan pembuatan model jaringan syaraf menggunakan *Keras* tidak perlu menuliskan kode untuk mengekspresikan perhitungan matematisnya satu persatu. Hal ini dikarenakan *Keras* sudah menyediakan beberapa model dasar untuk CNN dan dioptimasi untuk mempermudah penelitian tentang *Deep Learning*. Hasil akhir yang diharapkan yaitu teknik dasar dapat diimplementasikan lebih mudah dan cepat (Danukusumo, 2017).

1.2. Rumusan Masalah

Berdasarkan permasalahan diatas, adapun permasalahan yang akan dikaji dalam penelitian ini adalah sebagai berikut :

1. Bagaimana pengklasifikasian citra tomat dengan menggunakan metode CNN?
2. Bagaimana tingkat akurasi yang didapatkan dalam pengklasifikasian kelayakan tomat menggunakan metode CNN?
3. Bagaimana hasil pengklasifikasian kelayakan buah tomat dengan menggunakan metode CNN?

1.3. Batasan Masalah

Adapun batasan masalah yang digunakan peneliti agar pembahasan dalam penelitian ini tidak menyimpang dari pokok pembahasan. Maka peneliti memiliki batasan masalah sebagai berikut :

1. Objek tomat yang digunakan dalam penelitian ini merupakan varietas tomat merah dengan kelas klasifikasi sebanyak dua, yaitu tomat layak dan tidak layak.
2. Analisis pengolahan citra yang dilakukan dengan metode CNN yaitu dengan bantuan *package* Keras pada *software* RStudio versi 1.1.383.

1.4. Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut :

1. Mengetahui cara klasifikasi citra tomat menggunakan metode CNN.
2. Mengetahui tingkat akurasi yang didapatkan dalam pengklasifikasian kelayakan tomat menggunakan metode CNN.
3. Mengetahui hasil pengklasifikasian kelayakan buah tomat dengan menggunakan metode CNN.

1.5. Manfaat Penelitian

Adapun manfaat dari penelitian ini yakni sebagai berikut :

1. Diperoleh suatu cara untuk melakukan klasifikasi selain cara manual, yaitu klasifikasi dengan bantuan teknologi, seperti komputer.
2. Diperoleh suatu pengembangan algoritma analisis pengolahan citra dan pengidentifikasian menggunakan metode CNN untuk mengidentifikasi tingkat kelayakan buah tomat.

1.6. Sistematika Penulisan

Sistematika penulisan yang dipergunakan dalam penulisan tugas akhir ini dapat diuraikan sebagai berikut :

BAB I PENDAHULUAN

Pada bab ini akan dibahas tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini memaparkan penelitian-penelitian terdahulu yang berhubungan dengan permasalahan yang diteliti dan menjadi acuan konseptual.

BAB III LANDASAN TEORI

Pada bab ini akan dibahas tentang teori-teori dan konsep yang berhubungan dengan penelitian yang dilakukan dan mendukung dalam pemecahan masalahnya. Selain itu, bab ini juga memuat teori-teori dalam pelaksanaan pengumpulan dan pengolahan data serta saat melakukan penganalisaan.

BAB IV METODOLOGI PENELITIAN

Bab ini memaparkan populasi dan sampel, variabel penelitian, jenis dan sumber data, metode analisis data, dan tahapan penelitian.

BAB V ANALISIS DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai analisa yang dilakukan terhadap hasil pengumpulan, pengolahan dan analisa data yang diperoleh dari hasil penelitian.

BAB VI PENUTUP

Pada bab ini akan dibahas mengenai kesimpulan yang diperoleh dari hasil penelitian dan analisa data yang telah dilakukan serta saran-saran yang dapat diterapkan dari hasil pengolahan data yang dapat menjadi masukan yang berguna kedepannya.

BAB II

TINJAUAN PUSTAKA

Terkait dengan penelitian yang dilakukan penulis, maka penelitian terdahulu menjadi sangat penting agar dapat diketahui hubungan antara penelitian yang dilakukan sebelumnya dengan penelitian yang dilakukan pada saat ini, dan terjadinya suatu penjiplakan atau duplikasi dalam penelitian yang dilakukan tersebut mempunyai arti penting sehingga dapat diketahui kontribusi penelitian ini terhadap perkembangan ilmu pengetahuan. Penelitian tentang pengolahan citra tomat sudah banyak dilakukan oleh banyak peneliti sebelumnya dengan metode yang berbeda-beda. Penelitian mengenai identifikasi kematangan tomat dilakukan oleh Ghazali dkk dengan judul penelitian yaitu “Aplikasi Kematangan Tomat Berdasarkan Warna dengan Metode *Linear Discriminant Analysis (LDA)*. Tomat dikelompokkan ke dalam dua kelas yang telah ditentukan yaitu kelas matang dan kelas mentah. Hasil penelitian ini diharapkan dapat menggantikan cara mendeteksi kematangan tomat secara manual.

Pada tahun 2009, penelitian dengan judul “Klasifikasi Tingkat Kematangan Varietas Tomat Merah dengan Metode Perbandingan Kadar Warna” dilakukan oleh Ary Noviyanto. Hasil pengujian yang dilakukan adalah dengan total keberhasilan metode ini mencapai 95%. Penelitian dengan objek yang sama akan tetapi dengan metode berbeda juga dilakukan oleh Dila Deswari, dkk. Penelitian tersebut berjudul “Identifikasi Kematangan Buah Tomat Menggunakan Metoda *Backpropagation*”. Tingkat identifikasi kematangan buah tomat menggunakan metode ini berhasil dengan tingkat keberhasilan identifikasi sebesar 71,76%.

Penelitian lain juga dilakukan oleh Aisyah Hidayatullah (2013) dengan judul “Identifikasi Tingkat Kematangan Buah Tomat (*Lycopersicon Escule Mill*) Menggunakan Metode Pengolahan Citra dan Jaringan Saraf Tiruan”. Hasil pengujian menunjukkan tingkat akurasi sebesar 94,44% yang dinilai mampu melakukan identifikasi kematangan buah tomat dengan baik. Selain itu, penelitian

dengan objek dan metode yang sama juga dilakukan oleh Sella Kusumaningtyas (2016). Hasil penelitian menunjukkan bahwa tingkat keberhasilan identifikasi kematangan buah tomat adalah sebesar 43,33%.

Penelitian mengenai pengolahan citra tomat juga dilakukan oleh Suastika Yulia Riska, dkk (2016) dengan menggunakan metode yang berbeda dari penelitian yang disebutkan sebelumnya. Adapun judul penelitiannya adalah “Klasifikasi Level Kematangan Buah Tomat Berdasarkan Fitur Warna Menggunakan Multi-SVM”. Klasifikasi menggunakan metode ini memperoleh persentase sebesar 77,84%. Penelitian lain yang telah dilakukan oleh Ihsan Nugraha Putra Mukhti dengan judul “Sistem Otomasi dalam Penyortiran Tomat dengan *Image Processing* Menggunakan Metode Deteksi RGB”. Berdasarkan hasil pengujian, didapatkan bahwa akurasi tertinggi mencapai 88% dengan waktu komputasi rata-rata untuk PC dengan RAM 2GB adalah 4500 ms.

Penelitian mengenai tingkat kematangan tomat juga saat ini masih dilakukan akan tetapi dengan metode yang berbeda-beda. Penelitian terbaru yaitu dilakukan oleh Mochamad Angga Anggriawan, dkk (2017) dengan judul “Pengenalan Tingkat Kematangan Tomat Berdasarkan Citra Warna pada Studi Kasus Pembangunan Sistem Pemilihan Otomatis”. Hasil penelitian menunjukkan bahwa tingkat akurasi atau kesesuaian sistem pengenalan tingkat kematangan tomat untuk tomat bervariasi menghasilkan persentase sebesar 83,75% (pengambilan citra secara diam” dan 83,33% (pengambilan citra secara bergerak).

Penelitian mengenai pengolahan citra tidak hanya terbatas mengenai objek tomat, akan tetapi banyak sekali penelitian lain dengan studi kasus berbeda-beda. Hal ini membuktikan bahwa pengolahan citra sangat berguna untuk kehidupan sehari-hari. Beberapa peneliti yang melakukan penelitian mengenai pengolahan citra adalah Endarko, dkk (2005) yaitu melakukan pengenalan pola tulisan tangan dengan metode JST. Nazrul Effendy, dkk juga meneliti tentang deteksi pornografi. Mas’ud Effendy, dkk (2017) melakukan pengolahan citra untuk mengidentifikasi jenis dan mutu teh.

Beberapa penelitian lain mengenai pengolahan citra yang dalam bidang pertanian juga banyak dilakukan oleh peneliti seperti Agus Supriatna Somantri, dkk (2013) yang melakukan indentifikasi mutu fisik beras. Yanuar Putu Wiharja dan Agus Harjoko (2014) juga meneliti tentang klasifikasi mutu buah pisang dengan menggunakan JST. Sementara penelitian yang baru-baru ini dilakukan juga yaitu oleh Doli Garesya Agian, dkk (2015) untuk melakukan identifikasi kematangan buah markisa (*Passiflora Edulis*).

Berdasarkan penelitian yang disebutkan sebelumnya, tidak ada penelitian mengenai pengolahan citra tomat menggunakan metode *Convolutional Neural Network (CNN)*. Salah satu penelitian yang menggunakan metode CNN dalam pengolahan citra pada bidang pertanian adalah penelitian yang dilakukan oleh Rismiyati (2016) dengan judul “Implementasi *Convolutional Neural Network* untuk Sortasi Mutu Salak Ekspor Berbasis Citra Digital”. Beberapa penelitian lain yang menggunakan metode CNN di bidang lain yaitu oleh Muhammad Zufar dan Budi Setiyono (2016) dengan judul “*Convolutional Neural Network* untuk Pengenalan Wajah Secara *Real-Time*”. Hasil uji coba dengan model CNN tersebut sampai kedalaman 7 lapisan meraih rata-rata tingkat akurasi lebih dari 89% dalam ± 2 frame per detik. Penelitian lain dilakukan oleh I Wayan Suartika E. P., dkk (2016) dengan judul “Klasifikasi Citra Menggunakan *Convolutional Neural Network (Cnn)* pada *Caltect 101*”. Hasil penelitian menyebutkan bahwa metode CNN mampu melakukan klasifikasi dengan baik. Penelitian terbaru dilakukan oleh Kevin Pudi Danukusumo (2017) dengan judul “Implementasi *Deep Learning* Menggunakan *Convolutional Neural Network* untuk Klasifikasi Citra Candi Berbasis GPU”.

Tabel 2.1 Perbandingan dengan penelitian terdahulu

No.	Penulis	Judul	Metode	Persamaan	Perbedaan
1.	Ghazali, dkk (2014)	Aplikasi Kematangan Tomat Berdasarkan Warna dengan Metode <i>Linear Discriminant Analysis (LDA)</i>	<i>Linear Discriminant Analysis (LDA)</i>	Sama-sama meneliti tentang identifikasi tingkat kematangan tomat	Metode yang digunakan berbeda
2.	Novianto (2009)	Klasifikasi Tingkat Kematangan Varietas Tomat Merah dengan Metode Perbandingan Kadar Warna	Perbandingan Kadar Warna	Sama-sama meneliti tentang identifikasi tingkat kematangan tomat dengan jumlah klasifikasi tingkat kematangan tomat yang sama	Metode yang digunakan berbeda
3.	Deswari, dkk (2013)	Identifikasi Kematangan Buah Tomat Menggunakan Metoda <i>Backpropagation</i>	<i>Backpropagation</i>	Sama-sama meneliti tentang identifikasi tingkat kematangan tomat dengan jumlah	Metode yang digunakan berbeda

				klasifikasi tingkat kematangan tomat yang sama	
4.	Hidayatul lah (2013)	Identifikasi Tingkat Kematangan Buah Tomat (<i>Licopersicon Escule Mill</i>) Menggunakan Metode Pengolahan Citra dan Jaringan Saraf Tiruan	Jaringan Saraf Tiruan (JST)	Sama-sama meneliti tentang identifikasi tingkat kematangan tomat	Metode yang digunakan berbeda, serta ada banyak jumlah klasifikasi tomat yang digunakan yaitu <i>green, break, turning, pink,</i> <i>red light</i> , dan <i>red</i>
5.	Kusuman ingtyas (2016)	Identifikasi Kematangan Buah Tomat Berdasarkan Warna Menggunakan Metode Jaringan Saraf Tiruan (JST)	Jaringan Saraf Tiruan (JST)	Sama-sama meneliti tentang identifikasi tingkat kematangan tomat dengan jumlah klasifikasi tingkat kematangan tomat yang sama	Metode yang digunakan berbeda
6.	Riska, dkk (2016)	Klasifikasi Level Kematangan Buah Tomat Berdasarkan Fitur Warna	Multi-SVM	Sama-sama meneliti tentang identifikasi tingkat kematangan	Metode yang digunakan berbeda

		Menggunakan Multi-SVM		tomat	
7.	Mukhti (2015)	Sistem Otomasi dalam Penyortiran Tomat dengan <i>Image Processing</i> Menggunakan Metode Deteksi RGB	Deteksi RGB	Sama-sama meneliti tentang objek tomat	Metode yang digunakan berbeda
8.	Anggriawan, dkk (2017)	Pengenalan Tingkat Kematangan Tomat Berdasarkan Citra Warna pada Studi Kasus Pembangunan Sistem Pemilihan Otomatis	<i>Learning Vector Quantization (LVQ)</i>	Sama-sama meneliti tentang identifikasi tingkat kematangan tomat	Metode yang digunakan berbeda, serta hanya ada dua klasifikasi tomat yang digunakan yaitu matang dan belum matang
9.	Endarko (2005)	Aplikasi Pengolahan Citra dan Jaringan Saraf Tiruan untuk Pengenalan Pola Tulisan Tangan	Jaringan Saraf Tiruan (JST)	Sama-sama melakukan pengolahan citra	Objek dan metode yang digunakan berbeda

10.	Effendy, dkk (2013)	Deteksi Pornografi pada Citra Digital Menggunakan Pengolahan Citra dan Jaringan Saraf Tiruan	Jaringan Saraf Tiruan (JST)	Sama-sama melakukan pengolahan citra	Objek dan metode yang digunakan berbeda
11.	Effendi, dkk (2017)	Identifikasi Jenis dan Mutu Teh Menggunakan Pengolahan Citra Digital dengan Metode Jaringan Syaraf Tiruan	Jaringan Syaraf Tiruan (JST)	Sama-sama melakukan pengolahan citra	Objek dan metode yang digunakan berbeda
12.	Somantri, dkk (2013)	Identifikasi Mutu Fisik Beras dengan Menggunakan Teknologi Pengolahan Citra dan Jaringan Syaraf Tiruan	Jaringan Saraf Tiruan (JST)	Sama-sama melakukan pengolahan citra	Objek dan metode yang digunakan berbeda
13.	Yanuar (2014)	Pemrosesan Citra Digital untuk Klasifikasi Mutu Buah Pisang Menggunakan Jaringan Saraf Tiruan	Jaringan Saraf Tiruan (JST)	Sama-sama melakukan pengolahan citra	Objek dan metode yang digunakan berbeda

14.	Agian, dkk (2015)	Identifikasi Kematangan Buah Markisa (<i>Passiflora Edulis</i>) dengan Pengolahan Citra Menggunakan Jaringan Saraf Tiruan	Jaringan Saraf Tiruan (JST)	Sama-sama melakukan pengolahan citra	Objek dan metode yang digunakan berbeda
15	Rismayati (2016)	Implementasi <i>Convolutional Neural Network</i> untuk Sortasi Mutu Salak Ekspor Berdasarkan Citra Digital	<i>Convolutional Neural Network (CNN)</i>	Metode yang digunakan sama	Objek yang digunakan berbeda
16.	Zufar (2016)	<i>Convolutional Neural Network</i> untuk Pengenalan Wajah Secara <i>Real-Time</i>	<i>Convolutional Neural Network (CNN)</i>	Metode yang digunakan sama	Objek yang digunakan berbeda
17.	Suartika (2016)	Klasifikasi Citra Menggunakan <i>Convolutional Neural Network (Cnn)</i> pada <i>Caltech 101</i>	<i>Convolutional Neural Network (CNN)</i>	Metode yang digunakan sama	Objek yang digunakan berbeda

18.	Danukusumono (2017)	Implementasi <i>Deep Learning</i> Menggunakan <i>Convolutional Neural Network</i> untuk Klasifikasi Citra Candi Berbasis GPU	<i>Convolutional Neural Network (CNN)</i>	Metode yang digunakan sama	Objek yang digunakan berbeda
-----	---------------------	--	---	----------------------------	------------------------------

BAB III

LANDASAN TEORI

3.1 Tomat

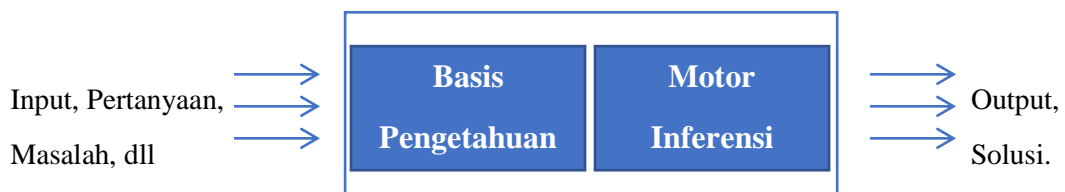
Tomat (*Lycopersicum esculentum*) merupakan tumbuhan asli Amerika Tengah dan Selatan dari keluarga *Solanaceae*. Kata “tomat” berasal dari kata dalam bahasa Nahuatl, dimana tomat merupakan keluarga dekat dari kentang. Tomat merupakan tumbuhan siklus hidup singkat, dapat tumbuh setinggi sekitar 1 sampai 3 meter. Tomat sering dianggap sebagai sayuran meskipun strukturnya adalah struktur buah. Tomat banyak dimanfaatkan sebagai sayuran, bumbu masak, buah meja, bahan pewarna, dan kosmetik. Tomat mengandung zat antioksidan cukup tinggi yang membantu untuk meningkatkan kekebalan tubuh, menghaluskan dan mencerahkan kulit, mencegah hipertensi, dan lainnya.

Tomat merupakan komoditas unggulan hortikultura dengan nilai penting di Indonesia. Di Indonesia, kebutuhan pasar sayuran terutama buah tomat dari tahun ke tahun mengalami peningkatan yang dicerminkan dari angka produksi tomat. Kendala yang sering dihadapi yaitu kurang tersedianya tomat varietas unggul yang mempunyai produksi tinggi, buah berkualitas baik, dan tahan terhadap hama dan penyakit. Tingginya kebutuhan akan tomat, sehingga sangat penting untuk melakukan pemilihan atau sortasi tomat dengan kualitas baik untuk memenuhi kebutuhan dalam maupun luar negeri yang dalam hal ini sering kali terjadi ketidaksesuaian antara kualitas yang diperlukan dengan kualitas produk yang dihasilkan.

3.2 Kecerdasan Buatan (*Artificial Intelligence*)

Artificial Intelligence (AI) merupakan bagian dari ilmu komputer yang mempelajari bagaimana menjadikan mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan manusia, bahkan bisa lebih baik. *Artificial Intelligence* (AI) menurut John McCarthy (1956) yaitu untuk mengetahui atau memodelkan proses berpikir manusia dan mendesain mesin sehingga bisa menirukan perilaku manusia. Mesin bisa bertindak seperti manusia

dengan dibekali pengetahuan serta kemampuan menalar yang baik. Bagian utama yang dibutuhkan dalam AI adalah basis pengetahuan (*knowledge base*) dan motor inferensi (*inference engine*). Basis pengetahuan berisi fakta, pemikiran, serta teori-teori. Sedangkan motor inferensi merupakan kemampuan dalam penarikan kesimpulan yang didasarkan oleh pengetahuan.



Gambar 3.1 Bagian Utama dalam Artificial Intelligence (AI)

Artificial Intelligence (AI) merupakan salah satu disiplin ilmu yang luas, beberapa lingkup utama AI antara lain adalah Sistem Pakar (*Expert System*), Pengolahan Bahasa Alami (*Natural Language Processing/NLP*), Pengenalan Ucapan (*Speech Recognition*), *Computer Vision*, *Intelligent Computer-Aided Instruction*, dan lainnya. Sistem pakar adalah usaha untuk menirukan seorang pakar. Tujuan dari sistem pakar yaitu untuk mentransfer kepakaran dari seorang pakar ke komputer, kemudian ke orang lain (orang yang bukan pakar). Pengolahan Bahasa Alami yaitu dimana pengguna bisa melakukan komunikasi dengan komputer menggunakan bahasa sehari-hari. Pengenalan ucapan yaitu dimana manusia dapat melakukan komunikasi dengan komputer menggunakan suara. *Computer vision* yaitu dalam hal menginterpretasikan objek atau gambar yang tampak melalui komputer. *Intelligent Computer-Aided Instruction* yaitu bagaimana komputer dapat berperan sebagai tutor yang dapat mengajar atau melatih.

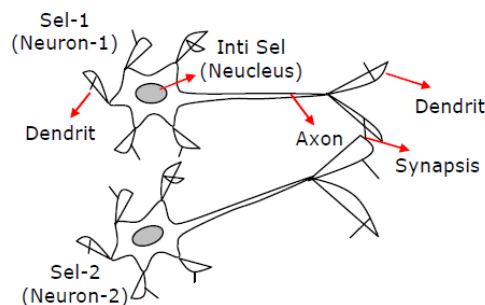
Artificial Intelligence (AI) dibuat berdasarkan sistem yang memiliki keahlian seperti manusia pada domain tertentu yaitu disebut dengan *soft computing*. *Soft computing* merupakan inovasi baru dalam membangun sistem cerdas yang mampu beradaptasi dan bekerja lebih baik jika terjadi perubahan lingkungan. *Soft computing* juga mengeksplorasi adanya toleransi terhadap ketidakpastian, ketidaktepatan, dan kebenaran parsial sehingga dapat diselesaikan

dan dikendalikan dengan mudah agar sesuai dengan realita. Metodologi yang sering digunakan dalam *soft computing* salah satunya adalah Jaringan Syaraf (menggunakan pembelajaran), yaitu Jaringan Syaraf Tiruan (*Artificial Neural Network/ANN*). Metodologi lain yang juga digunakan adalah Sistem *Fuzzy* (mengakomodasi ketepatan), *Probabilistic Reasoning* (Mengakomodasi Ketidakpastian), *Evolutionary Computing* (Optimasi).

3.3 Jaringan Syaraf Tiruan (*Artificial Neural Network/ANN*)

3.3.1 Konsep Jaringan Syaraf Tiruan

Ide mendasar dari Jaringan Syaraf Tiruan (JST) dimulai dari otak manusia yang memuat sekitar 10^{11} neuron yang berfungsi untuk memproses setiap informasi yang masuk. Komponen neuron terdiri dari : satu inti sel yang akan melakukan pemrosesan informasi, satu akson (*axon*) dan minimal satu *dendrit*. Informasi yang masuk akan diterima oleh *dendrit*. Selain itu, *dendrit* juga menyertasi akson sebagai keluaran dari suatu pemrosesan informasi. Setiap sel syaraf saling berhubungan dengan syaraf lainnya yang akan menghasilkan kemampuan tertentu pada kerja otak manusia dengan jumlah sekitar 10^4 sinapsis.



Gambar 3.2 *Susunan Syaraf Manusia*

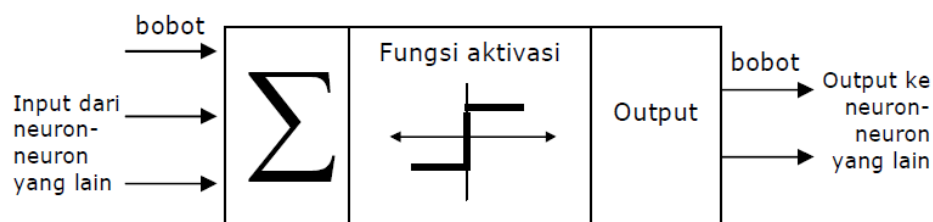
Gambar 3.2 di atas merupakan gambaran bagaimana susunan dari syaraf manusia. Hasil informasi yang diproses sebelumnya akan menjadi masukan bagi neuron lain dimana antar *dendrit* kedua sel tersebut dipertemukan dengan *synapsis*. Informasi yang dikirimkan antar neuron ini berupa rangsangan yang dilewatkan melalui *dendrit*. Informasi yang datang dan diterima oleh *dendrit* akan dijumlahkan dan dikirim melalui akson ke *dendrit* akhir yang bersentuhan dengan

dendrit dari neuron lain. Informasi tersebut akan diterima oleh neuron lain jika memenuhi batasan tertentu yang disebut dengan *threshold* atau nilai ambang. Pada kasus tersebut, neuron tersebut dikatakan teraktivasi. Hubungan antar neuron satu dengan neuron lainnya terjadi secara adaptif, yang berarti bahwa struktur hubungan tersebut terjadi secara dinamis. Otak manusia selalu memiliki kemampuan untuk belajar dengan melakukan adaptasi.

Konsep awal di atas menjadikan adanya konsep Jaringan Syaraf Tiruan (JST) yaitu merupakan bentuk representasi buatan dari otak manusia yang selalu mencoba untuk menstimulasikan proses pembelajaran pada otak manusia tersebut. Istilah buatan digunakan karena jaringan syaraf ini diimplementasikan ke dalam program komputer sehingga mampu menyelesaikan sejumlah proses perhitungan selama proses pembelajaran.

3.3.2 Komponen Jaringan Syaraf

Jaringan syaraf memiliki beberapa tipe yang berbeda, akan tetapi hampir semua komponen yang dimiliki sama. Jaringan syaraf juga terdiri dari neuron sama halnya dengan otak manusia, antar neuron juga berhubungan antar satu sama lain. Neuron-neuron tersebut akan melakukan transformasi informasi yang diterima melalui sambungan keduanya menuju neuron yang lain. Hubungan ini dikenal dengan sebutan bobot. Informasi tersebut disimpan pada suatu nilai tertentu pada bobot tertentu. Berikut ditunjukkan neuron pada jaringan syaraf.

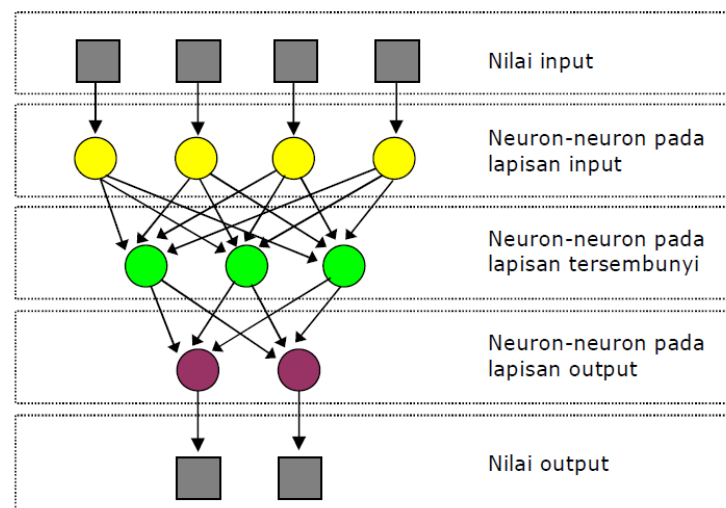


Gambar 3.3 Struktur Neuron Jaringan Syaraf

Cara kerja neuron pada jaringan syaraf mirip dengan cara kerja neuron biologis pada otak manusia. Informasi (*input*) akan dikirimkan dengan bobot kedatangan tertentu. Input tersebut kemudian diproses oleh suatu fungsi

perambatan yang akan menjumlahkan nilai-nilai semua bobot yang datang. Hasil penjumlahan ini kemudian akan dibandingkan dengan suatu nilai ambang (*threshold*) tertentu melalui fungsi aktivasi setiap neuron. Jika input tersebut melewati suatu nilai ambang tertentu, maka neuron tersebut akan diaktifkan. Jika tidak, neuron tersebut tidak akan diaktifkan. Apabila neuron diaktifkan, selanjutnya neuron tersebut akan mengirimkan *output* melalui bobot-bobot *outputnya* ke semua neuron yang berhubungan dengannya, begitu seterusnya.

Penempatan neuron-neuron pada jaringan syaraf yaitu dengan cara dikumpulkan dalam lapisan-lapisan (*layer*) yang disebut dengan lapisan neuron (*neuron layers*). Neuron pada satu lapisan dihubungkan dengan lapisan setelah dan sebelumnya, kecuali pada lampiran *output* dan *input*. Informasi (*input*) yang masuk pada jaringan syaraf akan dirambatkan dari lapisan *input* ke lapisan *output* melalui lapisan lainnya yang biasa dikenal dengan lapisan tersembunyi (*hidden layer*). Beberapa jaringan syaraf ada juga yang tidak mempunyai lapisan tersembunyi, serta ada juga jaringan syaraf dengan neuron yang disusun dalam bentuk matriks. Informasi juga bisa dirambatkan secara mundur pada jaringan, tergantung algoritma pembelajaran yang digunakan. Berikut ditunjukkan contoh gambar jaringan syaraf dengan tiga lapisan.



Gambar 3.4 Contoh Jaringan Syaraf dengan Tiga Lapisan

3.3.3 Arsitektur Jaringan

Neuron yang terletak pada lapisan yang sama juga memiliki keadaan yang sama. Hal penting yang menentukan kelakuan suatu neuron adalah pola bobot dan fungsi aktivasinya. Neuron pada lapisan yang sama akan memiliki fungsi aktivasi yang sama. Jika neuron pada suatu lapisan (misalnya pada lapisan tersembunyi) akan dihubungkan dengan neuron pada lapisan lain (misalnya neuron pada lapisan *output*), maka tiap neuron pada lapisan tersebut (misalnya lapisan tersembunyi) juga harus dihubungkan dengan setiap lapisan pada lapisan lainnya (misalnya lapisan *output*). Beberapa arsitektur jaringan syaraf antara lain adalah jaringan dengan lapisan tunggal (*single layer net*), jaringan dengan lapisan banyak (*multilayer net*), dan jaringan dengan lapisan kompetitif (*competitive layer net*).

3.4 Fungsi Aktivasi

Beberapa fungsi aktivasi yang digunakan pada jaringan syaraf antara lain adalah fungsi Undak Biner (Hard Limit), fungsi Undak Biner (Treshold), fungsi Bipolar (Symetric Hard Limit), fungsi Bipolar (dengan Treshold), fungsi Linear (Identitas), fungsi Saturating Linear, fungsi Symetric Saturating Linear, fungsi Sigmoid Biner, fungsi Sigmoid Bipolar, dan sebagainya. Fungsi aktivasi yang sering digunakan untuk model CNN adalah fungsi aktivasi ReLU (*Rectified Linear Unit*). ReLU (*Rectified Linear Unit*) merupakan lapisan aktivasi pada model CNN yang mengaplikasikan fungsi $f(x)=\max(0,x)$ yang berarti fungsi ini melakukan *thresholding* dengan nilai nol terhadap nilai piksel pada input citra. Aktivasi ini membuat seluruh nilai piksel yang bernilai kurang dari nol pada suatu citra akan dijadikan 0.

3.5 Deep Learning

Deep Learning (DL) merupakan salah satu bagian dari *Machine Learning* yang memanfaatkan JST untuk membantu menyelesaikan pada dataset yang besar. Teknik DL menjadikan arsitektur menjadi sangat kuat untuk *Supervised Learning* (Data pembelajaran mencakup keluaran yang sudah ditentukan). Penambahan lebih banyak lapisan menjadikan model pembelajaran yang bisa mewakili citra

berlabel dengan lebih baik. Aplikasi konsep JST yang lebih dalam atau yang memiliki banyak lapisan dapat ditanggihkan pada algoritma *Machine Learning* yang sudah ada sehingga komputer bisa belajar dengan skala yang besar, kecepatan, dan akurasi. Prinsip tersebut semakin berkembang hingga DL semakin sering digunakan pada komunitas industri dan riset dalam menyelesaikan masalah data besar seperti pada *Computer Vision*, *Speech Recognition*, dan *Neural Language Processing*.

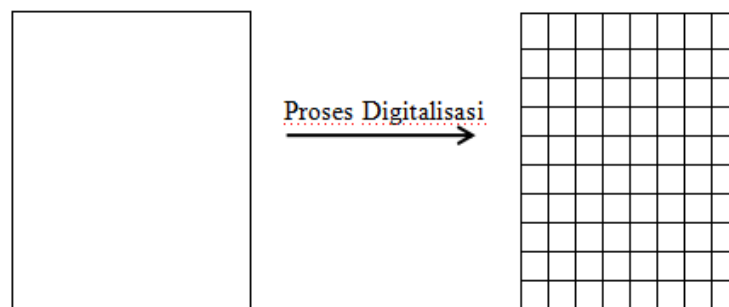
Apabila dihadapkan dengan kasus klasifikasi, salah satu fitur DL yang sering kali digunakan adalah *Feature Engineering* yang merupakan teknik yang penting untuk mencapai hasil yang baik pada tugas prediksi dengan cara mengekstrak pola yang berguna dari data, sehingga bisa lebih mudah dalam membedakan kelas. Akan tetapi, fitur tersebut lebih sulit dipelajari dikarenakan kumpulan data dan berbagai jenis data yang berbeda perlu diselesaikan dengan pendekatan teknik yang berbeda pula. Dalam hal ini, metode *Convolutional Neural Network* sangat bagus untuk menemukan fitur yang baik pada citra ke lapisan berikutnya dalam kasus DL. Metode ini membentuk hipotesis nonlinear sehingga bisa meningkatkan kekompleksitasan sebuah model. Model yang kompleks membutuhkan waktu pelatihan yang lama, sehingga penggunaan *Graphical Processor Unit* (GPU) sangat umum di dunia DL.

3.6 Citra

Citra merupakan kemiripan atau representasi dari suatu benda atau objek. Citra yang tampak merupakan cahaya yang direfleksikan dari sebuah objek. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut dan pantulan cahaya ditangkap oleh berbagai alat optik seperti kamera, mata manusia, *scanner*, dan lainnya, kemudian direkam. Menurut Usman Ahmad (2005) yang disebutkan dalam penelitian Muhammad Zunaidi (2015), citra merupakan kumpulan piksel yang disusun dalam larik 2 dimensi.

Secara matematis, citra dinyatakan sebagai suatu fungsi kontinu dari intensitas cahaya pada bidang dua dimensi. Fungsi intensitas cahaya pada bidang dwimatra disimbolkan dengan $f(x,y)$ dimana $f(x,y)$ adalah koordinat pada bidang

dwimantra dan f adalah intensitas cahaya (*brightness*) pada titik (x,y) . Pengolahan citra menggunakan komputer dapat dilakukan dengan merepresentasikan suatu citra (kontinu) secara numerik dengan nilai-nilai diskrit yang disebut dengan digitalisasi. Hasil dari proses tersebut dinamakan dengan citra digital. Secara umum, citra digital berbentuk empat persegi panjang dan dimensi ukurannya dinyatakan sebagai lebar x panjang atau tinggi x lebar.



Gambar 3.5 *Proses Digitalisasi Citra*

3.7 Citra Warna RGB (*Red, Green, Blue*)

Persepsi visual citra berwarna (*color images*) umumnya lebih kaya dibandingkan dengan citra hitam putih (*greyscale*), sehingga cira berwarna lebih disenangi dibandingkan dengan citra hitam putih. Citra warna menampilkan warna objek seperti warna aslinya, meskipun tidak selalu tepat demikian. Penelitian memperlihatkan bahwa kombinasi warna yang memberikan rentang warna paling lebar adalah *red*, *green*, dan *blue* atau sering disingkat dengan RGB. Warna-warna lain dapat diperoleh dengan mencampurkan ketiga warna pokok tersebut dengan perbandingan tertentu.

3.8 Histogram Citra

Informasi penting mengenai isi citra digital bisa diketahui dengan membuat histogram citra berupa grafik yang menggambarkan penyebaran nilai intensitas piksel dari bagian tertentu pada suatu citra. Histogram juga dapat menunjukkan banyak hal tentang kecerahan (*brightness*) dan kontras (*contrast*) dari citra. Hal ini menjadikan histogram sebagai alat bantu dalam pengolahan citra

baik secara kualitatif maupun kuantitatif. Khusus untuk citra berwarna, histogram dibuat untuk setiap kanal RGB (*Red, Green, Blue*). Sehingga jumlah histogram untuk citra warna sebanyak 3 buah, masing-masing untuk komponen merah, hijau, dan biru.

3.9 Pengolahan Citra

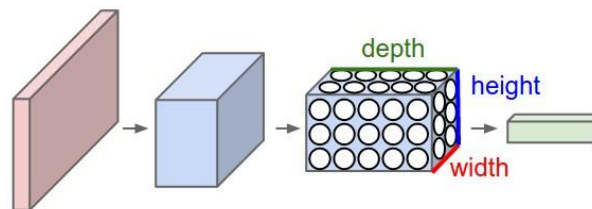
Pengolahan citra merupakan pemrosesan citra atau bisa disebut dengan *image processing*, lebih khususnya dengan menggunakan komputer menjadi citra dengan kualitas yang lebih baik. Ini berarti bahwa pengolahan citra merupakan kegiatan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin. Langkah-langkah dalam pengolahan citra digital antara lain adalah :

1. Akuisisi citra, yaitu merupakan proses untuk menangkap atau mengambil citra yang dibutuhkan dengan menggunakan sensor pencitraan berupa kamera, *scanner*, dan lainnya.
2. *Preprocessing* citra, yaitu dilakukan untuk menyiapkan citra untuk diproses lebih lanjut, bisa berupa ekstraksi fitur maupun kebutuhan klasifikasi. Teknik *preprocessing* citra yang umum digunakan antara lain adalah *cropping* dan perubahan ukuran citra.
3. Segmentasi, yaitu membagi sebuah citra menjadi beberapa bagian penyusunnya. Proses segmentasi dilakukan sampai objek yang diinginkan dalam suatu aplikasi terpisah dari objek aslinya. Tingkat kesuksesan dari sebuah sistem pengenalan citra juga dipengaruhi oleh segmentasi yang akurat.
4. Representasi dan deskripsi. Representasi yaitu menyatakan data piksel ke dalam bentuk data yang mampu diolah oleh komputer. Sementara proses deskripsi dilakukan untuk mengekstrak atribut yang menghasilkan informasi kuantitatif yang diinginkan atau yang merupakan fitur untuk membedakan citra antar kelas.
5. Pengenalan, yaitu proses pemberian label pada objek sesuai dengan fitur yang dimiliki objek.

3.10 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan jaringan syaraf yang digunakan khusus untuk mengolah data berstruktur grid, salah satunya berupa citra dua dimensi. Proses konvolusi merupakan operasi aljabar linear yang mengkalikan matriks dari filter pada citra yang akan diproses. Proses tersebut dinamakan dengan lapisan konvolusi dan merupakan salah satu jenis dari banyak lapisan yang bisa dimiliki dalam satu jaringan.

Meskipun demikian, lapisan konvolusi merupakan lapisan utama dan terpenting untuk digunakan. Jenis lapisan lain yang dapat digunakan adalah *Pooling Layer*. Lapisan *Pooling Layer* digunakan untuk mengambil nilai rata-rata atau nilai maksimal dari bagian piksel pada citra. Berikut ditampilkan struktur jaringan CNN. Setiap lapisan *input* yang dimasukkan memiliki volume yang berbeda dan diwakili dengan kedalaman, tinggi, dan lebar. Besaran yang akan didapatkan bisa berbeda-beda, tergantung hasil filtrasi dari lapisan sebelumnya serta banyaknya *filter* yang digunakan. Model jaringan seperti pada gambar 3.6 berikut sudah terbukti ampuh dalam menangani permasalahan klasifikasi citra.



Gambar 3.6 Contoh Jaringan CNN

3.11 Operasi Konvolusi

Operasi konvolusi merupakan operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi *output* sebagai *feature map* dari input citra. *Input* dan *output* ini dapat dilihat sebagai dua argumen bernilai riil. Operasi konvolusi dapat dituliskan sebagai berikut.

$$s(t) = (x * w)(t) \quad (1)$$

Fungsi $s(t)$ memberikan *output* tunggal berupa *feature map*. Argumen pertama adalah *input* yang merupakan x dan argumen kedua w sebagai kernel atau filter. Apabila dilihat *input* sebagai citra dua dimensi, maka bisa dikatakan t sebagai

piksel dan menggantinya dengan i dan j . Maka dari itu, operasi untuk konvolusi ke *input* dengan lebih dari satu dimensi dapat menulis sebagai berikut.

$$S(i, j) = (K * I)(i, j) = \sum_{-\infty}^{\infty} \sum_{-\infty}^{\infty} I(i-m, j-n) K(m, n) \quad (2)$$

Persamaan 3.2 di atas merupakan perhitungan dasar dalam operasi konvolusi, dengan i dan j adalah piksel dari citra. Perhitungan tersebut bersifat kumulatif dan muncul saat K sebagai kernel/filter, I sebagai *input* dan kernel yang dapat dibalik relatif terhadap *input*. Sebagai alternatif, operasi konvolusi dapat dilihat sebagai perkalian matriks antara citra masukan dan kernel dimana keluarannya dihitung dengan *dot product*. Selain itu, penentuan volume *output* juga dapat ditentukan dari masing-masing lapisan dengan *hyperparameters*. *Hyperparameter* yang digunakan pada persamaan di bawah ini digunakan untuk menghitung banyaknya neuron aktivasi dalam sekali *output*.

$$(W - F + 2P) / S + 1 \quad (3)$$

Berdasarkan persamaan di atas, dapat dihitung ukuran spasial dari volume *output* dimana *hyperparameter* yang dipakai adalah ukuran volume (W), filter/kernel (F), *Stride* yang diterapkan (S) dan jumlah *padding* nol yang digunakan (P). *Stride* merupakan nilai yang digunakan untuk menggeser *filter* melalui *input* citra dan *Zero Padding* adalah nilai untuk mendapatkan angka nol di sekitar *border* citra.

3.12 *Stride*

Stride merupakan parameter yang menentukan berapa jumlah pergeseran filter. Jika nilai *stride* adalah satu, maka filter akan bergeser sebanyak satu piksel secara horizontal lalu vertikal. Semakin kecil *stride* yang digunakan, maka semakin detail informasi yang didapatkan dari sebuah input, namun membutuhkan komputasi lebih jika dibandingkan dengan *stride* yang besar.

3.13 *Zero Padding*

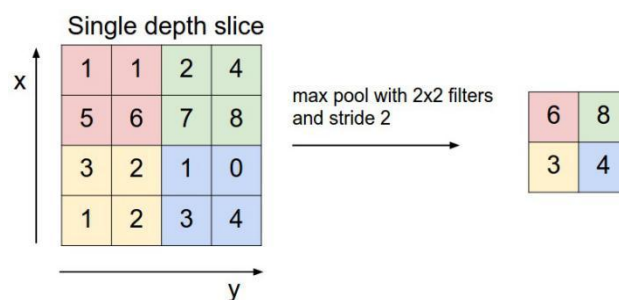
Zero padding merupakan parameter yang menentukan jumlah piksel (berisi nilai nol) yang akan ditambahkan di setiap sisi dari input. Hal ini

digunakan untuk memanipulasi dimensi *output* dari *convolutional layer* (*feature map*). Penggunaan *zero padding* memungkinkan untuk mengatur dimensi *output* agar tetap sama dengan dimensi input, atau setidaknya tidak berkurang secara drastis. Sehingga selanjutnya bisa dilakukan ekstraksi *feature* yang lebih mendalam. Selain itu, penggunaan *zero padding* bisa menjadikan performa dari model menjadi meningkat, karena filter akan fokus pada informasi yang sebenarnya yaitu yang berada di antara *zero padding* tersebut.

3.14 Pooling Layer

Pooling Layer merupakan lapisan yang menggunakan fungsi dengan *feature map* sebagai masukan dan mengolahnya dengan berbagai macam operasi statistik berdasarkan nilai piksel terdekat. Lapisan *pooling* pada model CNN biasanya disisipkan secara teratur setelah beberapa lapisan konvolusi. Lapisan *pooling* yang dimasukkan di antara lapisan konvolusi secara berturut-turut dalam arsitektur model CNN dapat secara progresif mengurangi ukuran volume output pada *feature map*, sehingga jumlah parameter dan perhitungan di jaringan berkurang, serta untuk mengendalikan *overfitting*.

Hal terpenting dalam pembuatan model CNN adalah dengan memilih banyak jenis lapisan *pooling*. Hal ini dapat menguntungkan kinerja model (Lee, Gallagher, & Tu, 2015). Lapisan *pooling* bekerja di setiap tumpukan *feature map* dan mengurangi ukurannya. Bentuk lapisan *pooling* yang paling umum adalah dengan menggunakan filter berukuran 2x2 yang diaplikasikan dengan langkah sebanyak 2 dan kemudian beroperasi pada setiap irisan dari *input*. Bentuk seperti ini akan mengurangi *feature map* hingga 75% dari ukuran aslinya. Berikut gambar contoh operasi *Max Pooling*.



Gambar 3.7 Contoh Operasi Max Pooling

Lapisan *pooling* akan beroperasi pada setiap irisan kedalaman volume input secara bergantian. Pada gambar di atas, lapisan *pooling* menggunakan salah satu operasi maksimal yang merupakan operasi yang paling umum. Gambar 3.4. menunjukkan operasi dengan langkah 2 dan ukuran filter 2x2. Dari ukuran input 4x4, pada masing-masing 4 angka pada input operasi mengambil nilai maksimalnya dan membuat ukuran *output* baru menjadi 2x2.

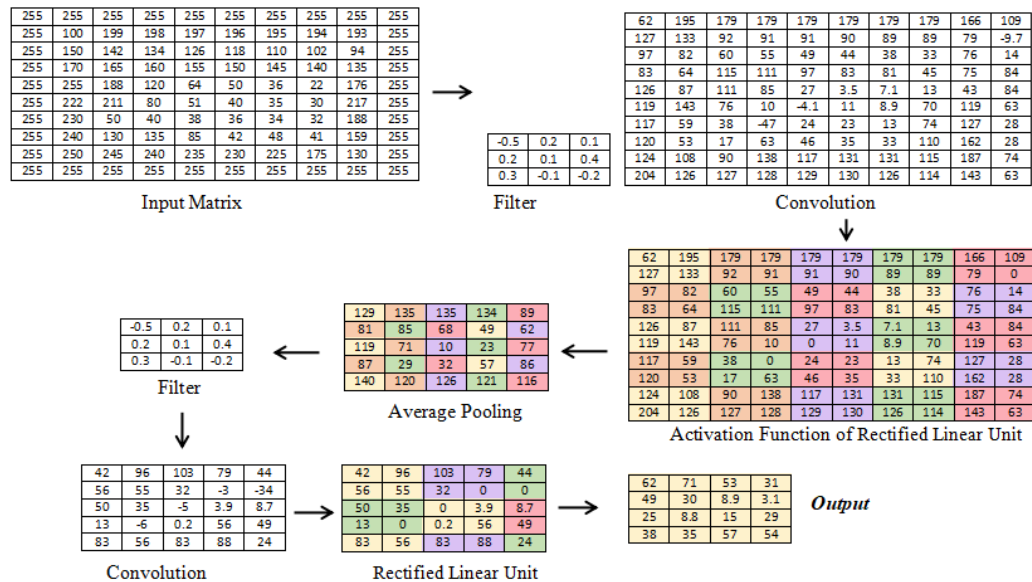
3.15 Ilustrasi Proses pada Lapisan Konvolusi

Lapisan konvolusi yang diaplikasikan untuk mendapatkan *feature map*. Contoh proses konvolusi dengan input berupa citra satu *channel* digambarkan seperti pada gambar berikut. Pada gambar tersebut, sebuah citra berukuran 10x10 piksel direpresentasikan sebagai matriks. Matriks awal diproses dengan dua *layer* konvolusi untuk mendapatkan *feature map*. Pada *layer* konvolusi pertama, filter yang digunakan berukuran 3x3 dengan bobot yang telah ditentukan. Hasil dari konvolusi pertama berupa matriks dengan ukuran 9x9.

Setelah melalui proses konvolusi, fungsi aktivasi dikenakan pada hasil konvolusi. Fungsi aktivasi yang digunakan adalah *reLu*. *Output* dari fungsi *reLu* kemudian dikenakan *pooling* dengan filter berukuran 2x2 dan *stride* sebesar dua. Sebelum melakukan *pooling*, dapat digunakan *zero padding* sehingga matriks hasil *pooling* berukuran 5x5. Matriks ini kemudian melalui tahap konvolusi kedua dengan ukuran filter sama seperti sebelumnya, tetapi dengan bobot yang berbeda. Dalam hal ini, ukuran tidak harus sama dengan konvolusi tahap pertama dan merupakan parameter yang bisa dioptimalkan. Sementara bobot matriks merupakan nilai yang dicari melalui proses pembelajaran.

Output dari proses konvolusi tahap kedua dikenakan dengan fungsi aktivasi yang sama, yaitu *reLu*. *Pooling* yang dikenakan berukuran 2x2 dengan *stride* satu, sehingga menghasilkan matriks dengan ukuran 4x4. Proses konvolusi bisa dilanjutkan sesuai dengan matriks akhir yang diinginkan. Dalam hal ini, jika konvolusi dihentikan sampai tahap kedua, maka matriks berukuran 4x4 tersebut menjadi input bagi *neural network*. Jika filter yang digunakan sejumlah n , maka input bagi *neural network* adalah $n \times 4 \times 4$ *nodes*. Pada praktiknya, penggunaan fungsi aktivasi dan *pooling* bisa

dibalik urutannya tanpa mengubah hasil dari konvolusi. Pembalikan ukuran ini bertujuan untuk mengurangi proses yang digunakan sehingga menjadi lebih cepat.



Gambar 3.8 Ilustrasi Proses Konvolusi

3.16 Fully-Connected Layer

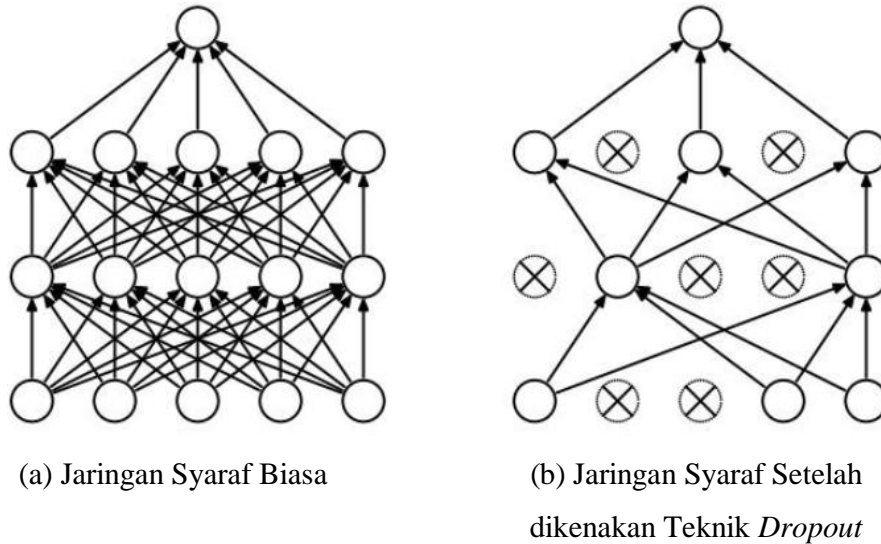
Hasil dari proses konvolusi menjadi input pada *fully-connected layer*. *Fully-connected layer* adalah lapisan dimana semua neuron aktivasi dari lapisan sebelumnya terhubung semua dengan neuron di lapisan selanjutnya seperti halnya jaringan saraf tiruan biasa. Setiap aktivasi dari lapisan sebelumnya perlu diubah menjadi data satu dimensi sebelum dapat dihubungkan ke semua neuron di *fully-connected layer*. Lapisan ini biasanya digunakan pada metode *Multi-Layer Perceptron (MLP)* dan bertujuan untuk mengolah data sehingga bisa diklasifikasikan.

Perbedaan antara *fully-connected layer* dengan lapisan konvolusi biasa adalah neuron di lapisan konvolusi terhubung hanya ke daerah tertentu pada input, sementara pada *fully-connected layer* memiliki neuron yang secara keseluruhan terhubung. Namun, kedua lapisan tersebut masih mengoperasikan produk dot, sehingga fungsinya tidak begitu berbeda.

3.17 Dropout Regularization

Dropout adalah teknik regularisasi jaringan syaraf dimana beberapa neuron akan dipilih secara acak dan tidak dipakai selama pelatihan. Neuron-neuron ini dapat

dibuang secara acak. Hal ini berarti bahwa kontribusi neuron yang dibuang akan diberhentikan sementara jaringan dan bobot baru juga tidak diterapkan pada neuron pada saat melakukan *backpropagation*.



Gambar 3.9 Contoh Implementasi *Dropout Regularization*

Pada gambar 3.9, jaringan syaraf (a) merupakan jaringan syaraf biasa dengan 2 lapisan tersembunyi. Sedangkan pada bagian (b) jaringan syaraf sudah diaplikasikan teknik regularisasi *dropout* dimana ada beberapa neuron aktivasi yang tidak dipakai lagi. Teknik ini sangat mudah diimplementasikan pada model CNN dan akan berdampak pada performa model dalam melatih serta mengurangi *overfitting*.

3.18 Softmax Classifier

Softmax Classifier merupakan bentuk lain dari algoritma *Logistic Regression* yang dapat kita gunakan untuk pengklasifikasian. Standar klasifikasi yang biasa dilakukan oleh algoritma *Logistic Regression* adalah tugas untuk klasifikasi kelas biner. Pada *Softmax* bentuk persamaan yang muncul adalah sebagai berikut.

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (4)$$

Notasi f_j menunjukkan hasil fungsi untuk setiap elemen ke- j pada vektor keluaran kelas. Argumen z adalah hipotesis yang diberikan oleh model pelatihan agar dapat diklasifikasi oleh fungsi *Softmax*.

Softmax juga memberikan hasil yang lebih intuitif dan juga memiliki interpretasi probabilistik yang lebih baik dibanding algoritma klasifikasi lainnya. *Softmax* memungkinkan kita untuk menghitung probabilitas untuk semua label. Dari label yang ada akan diambil sebuah vektor nilai bernilai riil dan merubahnya menjadi vektor dengan nilai antara nol dan satu yang bila semua dijumlah akan bernilai satu.

3.19 *Crossentropy Loss Fuction*

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. *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.

3.20 *Stochastic Gradient Descent*

Gradient Descent adalah salah satu algoritma yang paling populer untuk melakukan optimasi pada model jaringan syaraf tiruan dan algoritma ini adalah cara yang paling sering dipakai dalam berbagai macam model pembelajaran. Pada dasarnya ketika akan melatih sebuah model, kita membutuhkan sebuah *Loss Function* yang dapat memungkinkan kita untuk mengukur kualitas dari setiap bobot atau parameter tertentu. Tujuan pengoptimalan adalah untuk menemukan parameter yang dapat meminimalkan *Loss Function*.

Gradient Descent bekerja dengan cara meminimalkan fungsi $J(\theta)$ yang memiliki parameter θ dengan memperbarui parameter ke suatu arah menurun. *Gradient Descent* memiliki *Learning Rate* (η) yang digunakan untuk menentukan langkah-langkah yang kita ambil untuk mencapai titik minimum. Hal ini bisa

digambarkan dimana suatu objek akan menuruni sebuah bukit dengan langkah tersebut hingga mencapai pada lembah (titik minimum).

Stochastic Gradient Descent (SGD) merupakan metode *Gradient Descent* yang melakukan update parameter untuk setiap data pelatihan $x(i)$ serta label $y(i)$ dan memiliki persamaan dasar sebagai berikut.

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; x^{(i)}; y^{(i)}) \quad (6)$$

SGD sering melakukan *update* dengan varians tinggi yang menyebabkan fungsi objektif meningkat secara tidak beraturan. Di satu sisi hal ini dapat membuat *Loss Function* melompat ke titik minimal yang baru dan berpotensi melompat ke minimum yang tidak pasti. Namun, hal ini dapat dicegah dengan cara mengurangi *learning rate*, dan SGD akan menuruni *Loss Function* ke titik minimum dengan optimal.

3.21 *Max Norm Constrain*

Merupakan salah satu bentuk lain dari regularisasi yang berfungsi untuk merubah bobot pada vektor untuk setiap neuron dan digunakan untuk optimisasi gradien. Saat pelatihan regularisasi ini melakukan *update* parameter pada optimasi seperti biasa, dan kemudian menerapkan suatu batasan yang dapat merubah vektor bobot. Penggunaan regularisasi ini dapat menunjukkan perbaikan hasil dari model yang sudah dilatih. Regularisasi ini juga merupakan cara yang lebih baik untuk membatasi kompleksitas model dengan secara eksplisit merubah parameter setiap iterasi jika nilai parameter melebihi ambang batas kewajaran.

3.22 *Confusion Matrix*

Salah satu metode untuk mengukur performa dari suatu model klasifikasi adalah dengan mencari nilai *precision*, *recall*, serta nilai akurasi dari suatu model. Beberapa istilah yang umum dipakai dalam pengukuran kinerja model klasifikasi adalah *positive tuple* dan *negative tuple*. *Positive tuple* adalah *tuple* yang menjadi fokus pembahasan. Sedangkan *negative tuple* adalah *tuple* selain yang sedang menjadi fokus pembahasan.

Beberapa istilah lain yang merupakan dasar dalam pencarian nilai *precision*, *recall*, dan akurasi nilai *true positive (TP)*, *true-negative (TN)*, *false*

positive (FP), dan *false negative (FN)*. Istilah-istilah tersebut biasa dirangkum sebagai suatu matriks yang disebut *confusion matrix* sebagaimana ditunjukkan pada berikut.

		Predicted class		
		yes	no	Total
Actual class	yes	TP	FN	P
	no	FP	TN	N
Total		P'	N'	P + N

Gambar 3.10 *Confusion Matrix*

Pada gambar tersebut ditunjukkan, nilai *true positive* didefinisikan sebagai *positive tuple* yang diklasifikasikan dengan benar oleh model. *True negative* adalah *negative tuple* yang diklasifikasikan dengan benar oleh model. Sementara itu, *false positive* adalah *negative tuple* yang diklasifikasikan sebagai kelas positif oleh model. *False negative* adalah *positive tuple* yang diklasifikasikan sebagai kelas negatif oleh model klasifikasi. Berdasar *confusion matrix* pada Gambar 3.10 tersebut, kinerja model klasifikasi dapat dihitung.

3.23 Akurasi, *Precision*, dan *Recall*

Akurasi didefinisikan sebagai persentase dari data uji yang diklasifikasikan ke kelas yang benar. Akurasi dapat dinyatakan dalam persamaan berikut.

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

Selain akurasi, nilai *precision* dan *recall* juga dapat digunakan untuk mengetahui kinerja model klasifikasi. *Precision* merupakan ukuran ketepatan berupa persentase dari *tuple* yang diklasifikasikan ke kelas positif yang benar-benar merupakan kelas positif. Sedangkan *recall* adalah ukuran kelengkapan berupa persentase *tuple* positif yang diklasifikasikan sebagai kelas positif. Perhitungan *precision* dan *recall* dilakukan berdasarkan persamaan berikut.

$$presicion = \frac{TP}{TP + FP} \quad (8)$$

$$recall = \frac{TP}{TP + FN} \quad (9)$$

3.24 RStudio

R merupakan salah satu program *open source* yang pertama kali diciptakan oleh Ross Ihaka dan Robert Gentleman. Nama R berasal dari huruf pertama dari kedua statistikawan tersebut. Kini R dikembangkan sebagai upaya kolaborasi pakar-pakar statistikawan dan komputer di seluruh dunia. RStudio merupakan program *opensource* yang akan memberikan kemudahan bagi pemakai R sehingga mereka akan dapat menjalankan program R menjadi lebih menyenangkan karena RStudio memberikan informasi instruksi-instruksi apa saja yang harus dilakukan.

3.25 Keras

Keras merupakan salah satu *package* yang digunakan untuk menyelesaikan permasalahan mengenai jaringan syaraf. *Keras* dikembangkan dan fokusnya adalah mempercepat eksperimen pada proses konvolusi dan *recurrent* pada *neural networks*, maupun kombinasi antar keduanya. Pembuatan model jaringan syaraf menggunakan *Keras* tidak perlu menuliskan kode untuk mengekspresikan perhitungan matematisnya satu persatu. Hal ini dikarenakan *Keras* sudah menyediakan beberapa model dasar untuk CNN dan dioptimasi untuk mempermudah penelitian tentang *Deep Learning*. Proses komputasi menggunakan *Keras* berjalan lancar baik dengan menggunakan CPU maupun GPU.

BAB IV

METODOLOGI PENELITIAN

4.1 Populasi dan Sampel

Populasi dalam penelitian ini adalah citra tomat layak dan tidak layak sebanyak 228 citra. Sedangkan sampel yang digunakan dalam penelitian ini adalah 100 citra tomat dengan setiap kategori (layak dan tidak layak) terdiri dari 50 citra.

4.2 Variabel dan Definisi Operasional Variabel

Variabel yang digunakan dalam penelitian ini ditampilkan dalam Tabel 4.1 tentang penjelasan dan definisi operasional penelitian:

Tabel 4.1 *Definisi Operasional Variabel*

Variabel	Definisi Operasional Variabel
Citra Tomat Layak	Citra dari tomat matang yang memiliki kualitas bagus, ditandai dengan tidak terdapat bagian yang rusak maupun busuk
Citra Tomat Tidak Layak	Citra dari tomat matang yang yang disertai dengan adanya bagian yang rusak, lecet, maupun busuk

4.3 Jenis dan Sumber Data

Jenis data yang digunakan dalam penelitian ini adalah data primer. Data tersebut diperoleh dengan mengambil citra tomat secara langsung dengan menggunakan kamera dari Vivo Y55 *smartphone*.

4.4 Metode Analisis Data

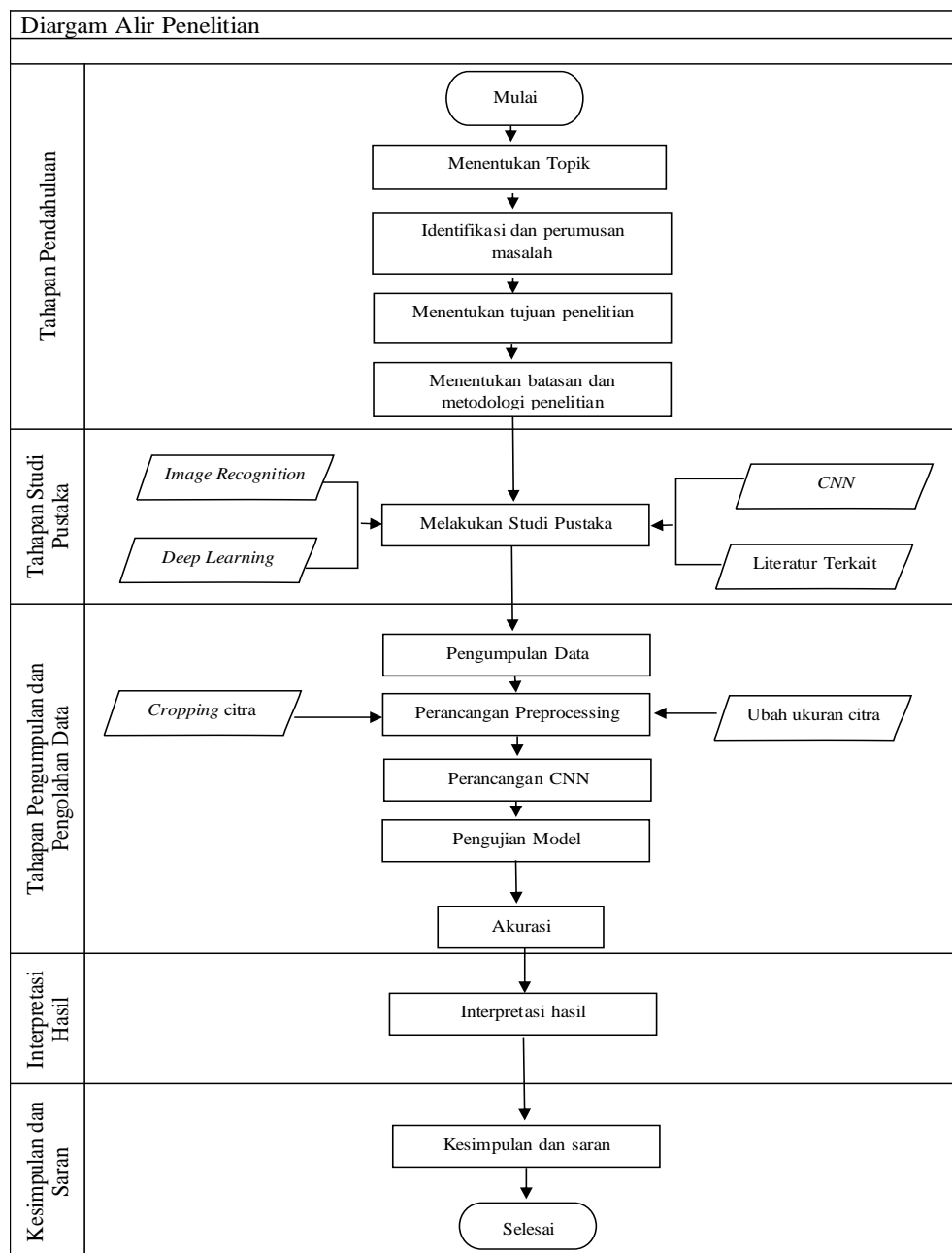
Software yang digunakan pada penelitian ini adalah *software* RStudio versi 1.1.383. Beberapa analisis data yang digunakan pada penelitian ini antara lain adalah sebagai berikut :

1. Histogram Gambar, yaitu digunakan untuk melihat representasi distribusi warna dalam sebuah gambar.

2. Metode *Deep Learning* yaitu *Convolutional Neural Network*, digunakan untuk mengklasifikasikan citra atau gambar.

4.5 Tahapan Penelitian

Langkah atau tahapan yang dilakukan pada penelitian ini digambarkan melalui Gambar 4.1 berikut ini :



Gambar 4.1 Tahapan Penelitian

BAB V

ANALISIS DAN PEMBAHASAN

Kegiatan yang dilakukan selama penelitian ini terdiri dari pengumpulan data citra, *preprocessing* citra, pelatihan model klasifikasi dengan menggunakan metode *Convolutional Neural Networks* (CNN), pemilihan parameter yang tepat untuk mendapatkan model terbaik, serta dilakukan pengujian model terbaik menggunakan data baru.

5.1. Pengumpulan Data Citra

Tahapan pertama yaitu mengumpulkan data citra yang akan digunakan. Data citra tomat diambil menggunakan kamera utama *smartphone* VIVO Y55 dengan resolusi 8 *Megapixel*. Penggunaan kamera dengan resolusi tersebut cukup baik untuk mengambil data citra. Penelitian lain juga menggunakan resolusi yang sama yaitu dilakukan oleh Rismiyati (2016) saat melakukan pengambilan data citra tomat menggunakan kamera Canon Ixus 860 IS dengan resolusi 8 *Megapixel*.



Gambar 5.1 Perangkat yang Digunakan untuk Pengambilan Data Citra

Sebelum dilakukan pengambilan citra, tomat diletakkan di atas kertas berwarna putih untuk mendapatkan citra dengan *background* putih. Hal ini dilakukan untuk menyamakan keadaan pada setiap tomat, fokus untuk mendapatkan objek tomat saja pada citra yang akan diambil, serta untuk mendapatkan citra dengan kualitas yang bagus. Pencahayaan juga diperhatikan saat pengambilan citra tomat. Hasil penelitian yang dilakukan oleh Muhammad Zufar dan Budi Setiyono (2016) menunjukkan bahwa pencahayaan pada citra mempengaruhi hasil akurasi saat melakukan deteksi citra. Sehingga pengambilan

citra pada penelitian ini dilakukan di ruangan terbuka dengan pencahayaan yang cukup, sehingga citra yang dihasilkan terlihat jelas.

Seperti yang disebutkan pada BAB IV sebelumnya, data penelitian terdiri dari dua kelas yaitu tomat yang layak dan tidak layak dengan karakteristik yang berbeda. Citra tomat layak diambil dari berbagai sisi agar bisa mendapatkan data citra yang lebih banyak dari segala sisi, begitu pula dengan citra yang tidak layak. Perbedaan dari tomat layak dan tidak layak sangat terlihat jelas secara visual. Hal ini memudahkan peneliti dalam menentukan kelas keduanya. Adapun contoh citra tomat layak dan tidak layak yang diambil ditunjukkan pada gambar 5.2 dan 5.3 seperti berikut.



Gambar 5.2 *Citra Tomat Layak*



Gambar 5.3 *Citra Tomat Tidak Layak*

5.2. Histogram Citra

Suatu citra digital dapat ditampilkan dalam beberapa format, salah satunya yaitu berupa citra warna yang menggunakan kombinasi dari tiga warna primer,

yaitu merah, hijau, dan biru atau yang dikenal dengan RGB (*Red, Green, Blue*). Adapun setiap titik pada citra yang dinyatakan dalam bentuk lebar x tinggi dengan satuan berupa piksel mewakili kombinasi dari ketiga warna ini. Histogram warna adalah representasi distribusi warna dalam sebuah gambar yang didapatkan dengan menghitung piksel dari setiap bagian *range* warna.

Pengolahan citra yang dilakukan menggunakan komputer diawali dari mencacah citra tersebut ke dalam bentuk titik-titik warna yang ditandai dengan angka yang menunjukkan tingkat kecerahan warna. Berikut ditunjukkan contoh representasi dari citra tomat layak dan tidak layak yang didapatkan dengan bantuan *software* RStudio.



Gambar 5.4 *Citra Tomat Layak*

Citra tomat layak di atas memuat banyak informasi. Proses manipulasi citra dapat dilakukan dengan menggunakan *software* R. Adapun beberapa hasil yang didapatkan dari citra di atas antara lain adalah berupa ukuran piksel, *channel* citra, serta histogram citra yang merepresentasikan warna dasar pembentuk citra tersebut.

```
Image
  colorMode      : Color
  storage.mode    : double
  dim             : 1895 1830 3
  frames.total    : 3
  frames.render   : 1

imageData(object)[1:5,1:6,1]
      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 0.7921569 0.7882353 0.7882353 0.7882353 0.7921569 0.7921569
[2,] 0.7921569 0.7882353 0.7882353 0.7921569 0.7921569 0.7921569
[3,] 0.7882353 0.7882353 0.7882353 0.7921569 0.7921569 0.7921569
[4,] 0.7921569 0.7921569 0.7921569 0.7960784 0.7960784 0.7960784
[5,] 0.7921569 0.7921569 0.7960784 0.7960784 0.7960784 0.7960784
```

Gambar 5.5 *Informasi Citra*

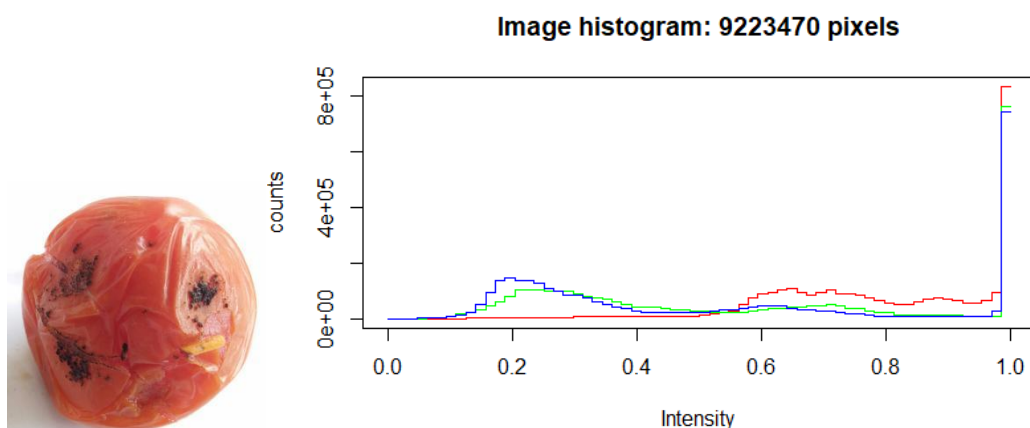
Citra di atas merupakan citra warna RGB (ditandai dengan makna *color* pada *colorMode*). Ukuran citra sebesar 1985x1830 piksel, dengan *channel* RGB sebesar 3. Selain itu, dapat diketahui juga nilai atau angka dalam bentuk matriks yang menggambarkan tingkat kecerahan dari suatu warna di setiap piksel dari citra tomat tersebut. Akan tetapi, nilai yang ditampilkan hanya sampai baris ke lima dan kolom ke enam saja dikarenakan keterbatasan dalam menampilkan *output* pada RStudio.

Adapun representasi distribusi warna RGB citra tomat layak ditunjukkan pada gambar histogram berikut. Pada histogram juga ditunjukkan nilai sebesar 10.403.550. Nilai tersebut didapatkan dari hasil perkalian 1985x1830x3 yang menunjukkan banyaknya elemen atau bagian yang menyusun citra tomat.



Gambar 5.6 *Histogram Citra*

Sedangkan contoh citra tomat tidak layak ditunjukkan pada gambar berikut bersamaan dengan histogramnya.



Gambar 5.7 *Tomat Tidak Layak dan Histogramnya*

Puncak histogram menunjukkan intensitas piksel yang menonjol. Lebar dari puncak menunjukkan rentang kontras dari gambar. Citra yang mempunyai kontras terlalu terang (*overexposed*) atau terlalu gelap (*underexposed*) memiliki histogram yang sempit, yaitu menceng ke kiri dan ke kanan. Citra dengan kontras normal memiliki bentuk yang hampir mirip dengan kurva distribusi normal. Gambar kedua citra di atas memiliki bentuk yang berbeda, akan tetapi dapat dikatakan bahwa kedua citra di atas memiliki kontras atau pencahayaan yang normal dilihat dari histogram yang dihasilkan.

5.3. *Preprocessing Citra*

Setelah didapatkan citra, selanjutnya dilakukan *preprocessing* citra berupa *cropping* dan pengubahan ukuran (*resize*) citra tomat. *Cropping* citra dilakukan untuk mendapatkan lebih banyak objek tomat dibandingkan *background* pada citra. Proses *cropping* menjadi hal yang efektif jika dilakukan, karena jika tidak dilakukan *cropping*, maka semua objek yang ada pada citra akan dianalisis oleh komputer, padahal objek yang ingin diteliti adalah berupa citra tomat saja. Sehingga, peneliti melakukan *cropping* pada bagian *background* dari citra tomat.

Pengubahan ukuran citra dilakukan menggunakan *software* RStudio versi 1.1.383 bersamaan dengan menyelesaikan tahapan pengolahan citra berikutnya, yaitu pelatihan model klasifikasi dan pengujian model dengan menggunakan metode *Convolutional Neural Networks* (CNN). Adapun salah contoh gambar sebelum dan sesudah dilakukan *cropping* dan *resize* ditunjukkan pada gambar berikut.



Gambar 5.8 Contoh Hasil *Cropping Citra Tomat Layak*

5.4. Pengolahan Citra

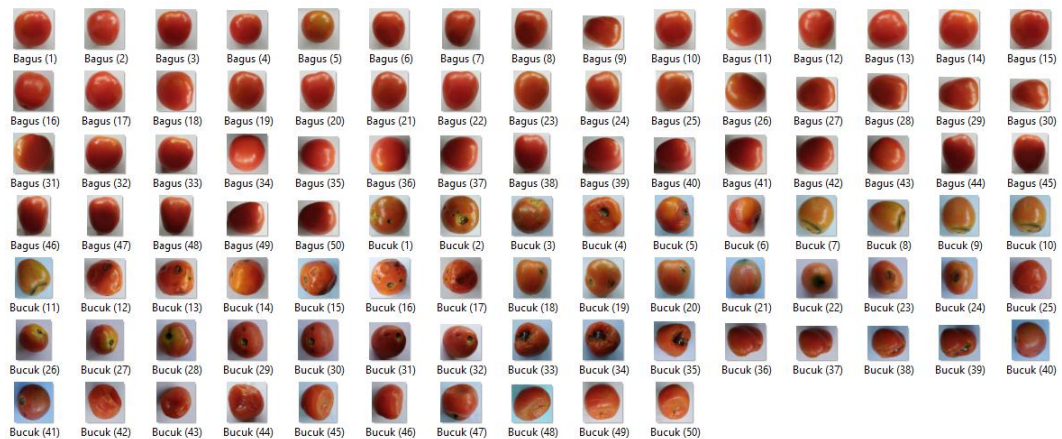
Berikut akan dijelaskan skrip mengenai gambaran proses atau rangkaian pengolahan citra secara umum yang diselesaikan dengan menggunakan *software* RStudio. Skrip selengkapnya dituliskan seperti pada Lampiran 1. Proses pengolahan citra pada *software* RStudio diawali dengan mengaktifkan *package* “Keras” dan “EBImage”. Seperti yang disebutkan sebelumnya, *package* Keras disetting untuk menyelesaikan masalah *Neural Network* (NN), *Convolutional Neural Network* (CNN), maupun *Recurrent NN* (RNN). Sedangkan *package* EBImage digunakan untuk dapat membaca dan menampilkan data citra.

```
library(keras)
library(EBImage)
```

Gambar 5.9 Pengaktifan Package Keras dan EBImage

Proses pengolahan citra pada *software* RStudio diawali dengan melakukan pemanggilan data citra untuk kemudian diproses ke tahapan berikutnya. Data ini berupa data citra tomat layak maupun tidak layak sebanyak 100 citra. Penggunaan data citra sebanyak 100 dikarenakan data tersebut cukup untuk melakukan penelitian mengenai citra tomat dengan dua klasifikasi, dikarenakan bentuk dari setiap citra tomat tidak terlalu variatif. Penggunaan 100 citra juga diharapkan bisa menjawab permasalahan pada penelitian ini.

Citra yang digunakan yaitu citra setelah dilakukan *cropping* yang kemudian disatukan ke dalam satu folder yang bernama “Tomat100”. Letak folder tersebut pada komputer yaitu pada *Local Disk* (D:). Citra tomat layak diberi nama “Bagus” sedangkan citra tomat tidak layak diberi nama “Bucuk”. Penamaan pada data citra ini dibuat untuk memudahkan saat membaca dataset tersebut pada RStudio yang digunakan. Citra tomat layak dibuat berurutan dari nomor 1 hingga 50, sehingga penamaannya menjadi Bagus(1) sampai (50). Kemudian dilanjutkan dengan citra tidak layak yaitu Bucuk(1) sampai Bucuk(50), sehingga urutannya dari nomor 51 sampai 100.



Gambar 5.10 *Dataset Citra*

Citra tersebut dipanggil atau siap untuk dilakukan analisis lanjut, kemudian nantinya akan disimpan dalam folder baru yang bernama “Run100”. Berikut skrip yang digunakan.

```
setwd('D://Tomat100')
save_in <- ("D://Run100/")
```

Gambar 5.11 *Pengambilan dan Penyimpanan Data Citra*

Selanjutnya dilakukan pembacaan dataset citra dari folder awal dengan menggunakan perintah *list.files()*, kemudian diubah ukurannya ke dalam bentuk 100x100 piksel untuk mempermudah proses pengolahan citra selanjutnya. Proses ini dilakukan satu persatu dari citra pertama hingga selesai, sehingga semua citra yang dibaca berukuran 100x100 piksel akan disimpan folder baru yaitu folder “Run100”.

```
images <- list.files()
w <- 100
h <- 100
for(i in 1:length(images))
```

Gambar 5.12 *Pengubahan Ukuran Citra*

Dari 100 citra yang digunakan, data citra ini kemudian dibagi menjadi data *training* dan data *testing* untuk keperluan pembuatan model dan pengecekan model yang terbentuk/validasi model. Pembagian data menjadi data *training* dan *testing* menggunakan skenario 80% dan 20% seperti penelitian pada umumnya.

Sehingga, 80% data digunakan untuk data *training*, 20% sisanya digunakan sebagai data *testing*. Jumlah data *training* yang digunakan yaitu sebanyak 80, setiap kategori memuat 40 data. Sedangkan data *testing* yang digunakan yaitu sebanyak 20 data, setiap kategori memuat 10 data.

Seperti yang disebutkan sebelumnya, penomoran pada dataset juga menunjukkan urutannya. Sehingga, ketika dataset dipanggil pada program, yang digunakan adalah nomornya. Seperti pada skrip berikut. Data *training* yang ditentukan adalah dataset dari nomor 1-40 yaitu citra tomat Bagus(1) sampai Bagus(40) dan dataset dari nomor 51-90 yaitu citra tomat Bucuk(1) sampai Bucuk(40). Begitu juga saat menentukan data *training*, dataset yang digunakan adalah dari nomor 41-50 yaitu Bagus(41)-Bagus(50) dan dataset 91-100 yaitu Bucuk(41)-Bucuk(50). Sehingga ini sesuai dengan skenario yang ditentukan sebelumnya.

Selain menentukan data *training* dan data *testing*, data tersebutpun bisa ditampilkan untuk menentukan apakah data yang akan digunakan tersedia semua atau tidak. Perintah yang bisa digunakan yaitu menggunakan *str*. Data set juga bisa ditampilkan menggunakan perintah *display*. Berdasarkan skrip di bawah, dilakukan perintah untuk menampilkan data ke 28 yang merupakan citra tomat layak.

```
#create train
train <- list_of_images[c(1:40, 51:90)]
str(train)
display(train[[28]])

#create test
test <- list_of_images[c(41:50, 91:100)]
test
display(test[[1]])
```

Gambar 5.13 Penampilan Dataset

Tahapan pengolahan citra selanjutnya yaitu implementasi *preprocessing* citra yaitu dengan mengubah ukuran citra menjadi 32x32 piksel. Ukuran citra yang dianjurkan adalah nilai pangkat dua. Nilai yang lebih kecil dari 32 adalah 16,

sedangkan yang lebih besar adalah 64. Penggunaan ukuran 16x16 menjadikan informasi piksel banyak yang hilang, sedangkan penggunaan ukuran 64x64 menjadikan pemrosesan menjadi lebih lambat. Sehingga pada akhirnya ukuran yang digunakan adalah sebesar 32x32 piksel. Perintah yang digunakan yaitu *resize*. Proses pengubahan ukuran dari data *training* dan *testing* dilakukan satu per satu atau menggunakan perulangan hingga selesai.

```
for (i in 1:80) {train[[i]] <- resize(train[[i]], 32, 32)}
for (i in 1:20) {test[[i]] <- resize(test[[i]], 32, 32)}
```

Gambar 5.14 *Proses Preprocessing Citra yaitu Mengubah Ukuran Citra Menjadi 32x32 piksel*

Proses pengubahan ukuran citra menandakan bahwa data citra siap untuk diproses lebih lanjut menggunakan model CNN. Akan tetapi terlebih dahulu peneliti memberikan label untuk data *training* dan *testing*. Pelabelan menggunakan angka 0 (nol) dan 1 (satu). Angka nol mewakili citra tomat layak, sedangkan angka satu mewakili citra tomat tidak layak. Pelabelan ini digunakan untuk memudahkan pembacaan data antara citra tomat layak dan tidak layak.

```
trainy <- c(rep(0,40),rep(1,40))
testy <- c(rep(0,10),rep(1,10))
```

Gambar 5.15 *Proses Pelabelan Data Citra*

Seperti yang dijelaskan pada bab sebelumnya, umumnya proses yang dilakukan terdiri dari proses konvolusi, fungsi aktivasi, dan *pooling*. Banyaknya proses ini disesuaikan dengan kebutuhan peneliti. Model yang dibentuk pada penelitian ini ditunjukkan pada skrip berikut.

```
# Model
model <- keras_model_sequential()

model %>%
  layer_conv_2d(filters = 32,
                kernel_size = c(3,3),
                activation = 'relu',
                input_shape = c(32, 32, 3)) %>%
```



```

layer_conv_2d(filters = 32,
              kernel_size = c(3,3),
              activation = 'relu') %>%
layer_max_pooling_2d(pool_size = c(2,2)) %>%
layer_dropout(rate = 0.25) %>%
layer_conv_2d(filters = 64,
              kernel_size = c(3,3),
              activation = 'relu') %>%
layer_conv_2d(filters = 64,
              kernel_size = c(3,3),
              activation = 'relu') %>%
layer_max_pooling_2d(pool_size = c(2,2)) %>%
layer_dropout(rate = 0.25) %>%
layer_flatten() %>%
layer_dense(units = 256, activation = 'relu') %>%
layer_dropout(rate=0.25) %>%
layer_dense(units = 2, activation = 'softmax') %>%

compile(loss = 'categorical_crossentropy',
        optimizer = optimizer_sgd(lr = 0.01,
                                   decay = 1e-6,
                                   momentum = 0.9,
                                   nesterov = T),
        metrics = c('accuracy'))

```

Gambar 5.16 Model CNN

Model di atas memuat beberapa jenis *layer* yang berbeda-beda, yaitu antara lain adalah *layer* konvolusi (*layer_conv_2d*), *layer pooling* (*layer_max_pooling_2d*), *layer dropout* (*layer_output*), *layer flatten* (*layer_flatten*), serta *layer dense* (*layer_dense*) dengan banyak dan ciri yang sudah ditentukan. Proses konvolusi dilakukan sebanyak 4 kali ditunjukkan oleh banyaknya *convolution layer* yang digunakan. Umumnya, 2-3 layer cukup untuk mendapatkan model klasifikasi. Akan tetapi, pada penelitian ini digunakan jaringan yang lebih *deep* atau banyak untuk melatih model dan melihat bagaimana kinerja model tersebut.

Sementara fungsi aktivasi (*activation*) yang digunakan yaitu *ReLU* yang menjadikan proses training menjadi lebih cepat. Ukuran kernel/filter yang

digunakan untuk setiap *layer* konvolusi yaitu sebesar 3x3. Sedangkan ukuran *pooling* yang digunakan yaitu sebesar 2x2. *Pooling* dilakukan dua kali, yaitu setelah dua proses konvolusi pertama dan setelah dua proses konvolusi selanjutnya. Hal ini dilakukan agar ukuran input tidak berkurang secara drastis di setiap proses yang dilakukan, sehingga informasi citra input yang dimiliki masih berguna dan bisa digunakan dalam proses klasifikasi.

Sedangkan jumlah filter/kernel yang digunakan juga variatif, yaitu sebanyak 32 filter untuk *layer* konvolusi 1 dan 2 serta 64 filter untuk *layer* konvolusi 3 dan 4. Penggunaan jumlah filter lebih banyak pada dua *layer* konvolusi terakhir dikarenakan ukuran input pada kedua lapisan tersebut lebih kecil, sehingga dibutuhkan lebih banyak filter untuk mengesktrak informasi citra. Pada proses akhir yaitu klasifikasi, *softmax classifier* digunakan untuk memberikan hasil yang lebih intuitif, sehingga memudahkan dalam melakukan klasifikasi dari interpretasi probabilistik untuk semua label yang dihasilkan.

Proses yang terjadi dari model di atas yaitu dimulai dengan melakukan “*encoding*” dari sebuah citra menjadi *features* yang berupa angka yang merepresentasikan citra tersebut. Pada proses konvolusi pertama, citra sebagai *input* berukuran 32x32 piksel sebenarnya adalah multidimensional *array* dengan ukuran 32x32x3 (3 adalah jumlah *channel* RGB). Citra inilah yang akan dikenakan beberapa proses seperti yang disebutkan pada model. Filter akan digerakkan dengan *stride* 1 ke seluruh bagian dari input citra, mulai dari sudut kiri atas sampai kanan bawah. Setiap pergeseran filter pada input citra dilakukan operasi “dot” atau perhitungan matematis. Pada penelitian ini tidak ditambahkan *zero padding*, sehingga ukuran input yang digunakan sama dengan 32x32 piksel. Dengan kata lain, *output* yang dihasilkan dari proses konvolusi memiliki ukuran yang lebih kecil.

Besarnya ukuran citra yang dihasilkan dari proses konvolusi pertama ini menjadi 30x30 piksel. Hasil ukuran ini juga bisa didapatkan menggunakan perhitungan dibawah ini.

$$(W-F+2P)/S+1 = 32-3+2(0)/1 + 1 = 30$$

Begitu juga seterusnya untuk melakukan perhitungan ukuran citra yang terbentuk. Hasil operasi ini kemudian dikenakan dengan fungsi aktivasi yaitu ReLu dan *pooling*. 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. Pada prinsipnya, *pooling layer* terdiri dari sebuah filter dengan ukuran dan *stride* tertentu akan bergeser ke seluruh area *feature map*. *Pooling* yang digunakan yaitu *Max Pooling*. Tujuan dari penggunaan *pooling layer* yaitu untuk mengurangi dimensi dari *feature map* (*downsampling*).

Dengan menggunakan filter ukuran 2x2 dan *stride* 1 pada operasi *max pooling*, didapatkan ukuran citra yang terbentuk setelah dilakukan proses konvolusi pertama, dikenakan fungsi aktivasi, kemudian operasi *max pooling* adalah sebesar 14x14 pixel. Hasil dari proses konvolusi berupa *feature map* yang digunakan kembali sebagai *input* untuk proses konvolusi berikutnya. Proses tersebut berjalan hingga berakhirnya proses konvolusi. Selain itu, metode regularisasi yang digunakan *dropout*, dimana beberapa neuron akan dipilih secara acak dan tidak dipakai selama pelatihan. Neuron-neuron ini dapat dibilang dibuang secara acak. Hal ini berarti bahwa kontribusi neuron yang dibuang akan diberhentikan sementara jaringan dan bobot baru juga tidak diterapkan pada neuron pada saat melakukan *backpropagation*.

Hasil dari proses konvolusi berupa *feature map* masih berbentuk multidimensional *array*, sehingga dilakukan *reshape* atau *flatten feature map* menjadi sebuah vektor agar bisa digunakan sebagai input dari *fully-connected layer*, hingga pada akhirnya dilakukan klasifikasi citra. Pada *fully-connected layer* digunakan neuron sebanyak 256, hingga pada akhirnya model terbentuk dan berhasil melakukan klasifikasi antara citra tomat layak dan tidak layak. Hasil dari model ditunjukkan pada gambar 5.17 berikut.

```
> summary(model)
```

Layer (type)	output shape	Param #
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
conv2d_2 (Conv2D)	(None, 28, 28, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	0
dropout_1 (Dropout)	(None, 14, 14, 32)	0
conv2d_3 (Conv2D)	(None, 12, 12, 64)	18496
conv2d_4 (Conv2D)	(None, 10, 10, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(None, 5, 5, 64)	0
dropout_2 (Dropout)	(None, 5, 5, 64)	0
flatten_1 (Flatten)	(None, 1600)	0
dense_1 (Dense)	(None, 256)	409856
dropout_3 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 2)	514
Total params: 475,938		
Trainable params: 475,938		
Non-trainable params: 0		

Gambar 5.17 *Summary Model CNN*

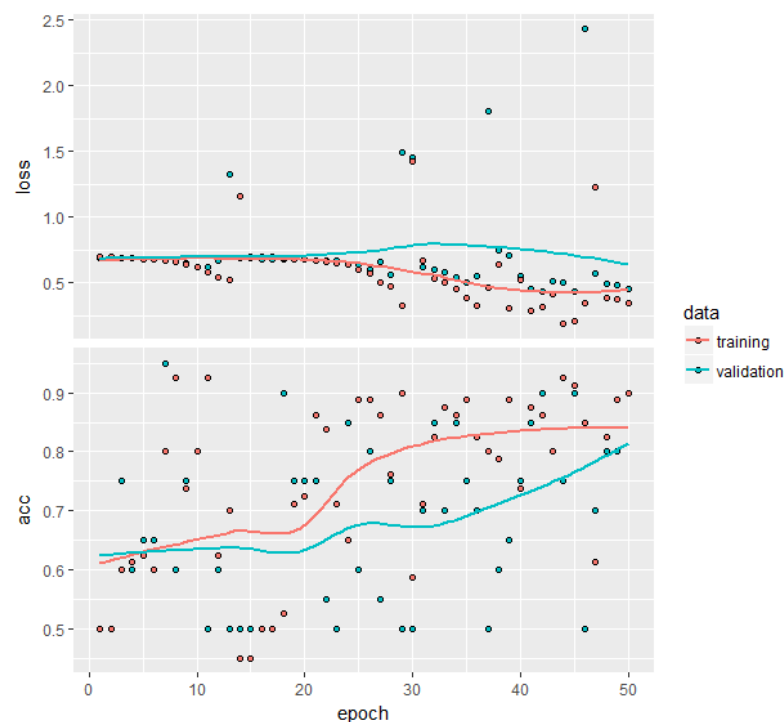
Jumlah parameter yang dilatih dalam model ini sebanyak 475.938 parameter. Ukuran citra pada setiap *layer* konvolusi semakin berkurang. Ukuran citra pada proses konvolusi terakhir sebelum masuk ke *fully connected layer* adalah 5x5 piksel. Sedangkan jumlah filter yang digunakan yaitu sebanyak 64. Kemudian dilakukan *reshape* sehingga ada 1600 neuron yang akan masuk pada *fully connected layer*. Jumlah neuron pada *hidden layer* yang digunakan yaitu sebanyak 256 neuron. Pada *fully connected layer* juga ditambahkan proses *dropout* saat melakukan pelatihan, hingga pada akhirnya dilakukan klasifikasi sebanyak 2 kategori.

Berdasarkan model yang terbentuk, dapat diketahui bagaimana hasil yang ditunjukkan terhadap data *training* dan *testing* yang digunakan. Adapun perbandingan dari hasil *loss* dan *accuracy* kedua data tersebut ditunjukkan pada tabel 5.1 berikut.

Tabel 5.1 Hasil Akurasi Data Training dan Testing

Data	Jumlah Data	Nilai Loss	Nilai Accuracy
Training	80	0.2944121	97.5%
Testing	20	0.4502037	90%

Kinerja dari *loss* dan *accuracy* model yang terbentuk, ditunjukkan melalui plot berikut ini.

**Gambar 5.18** Plot Loss dan Accuracy dari Data Training dan Data Testing

Loss function merupakan fungsi yang menggambarkan kerugian yang terkait dengan semua kemungkinan yang dihasilkan oleh model. *Loss function* yang baik adalah fungsi yang menghasilkan error yang diharapkan paling rendah. Sedangkan *accuracy* didefinisikan sebagai persentase dari data uji yang diklasifikasikan ke kelas yang benar. Berdasarkan tabel 5.1 di atas, nilai *loss* yang dihasilkan untuk data *training* yaitu sebesar 0.2944121. Nilai *loss* pada data *training* lebih rendah dibandingkan dengan data *testing*. Kedua nilai ini bisa dikatakan cukup rendah dan ini bagus dari model yang telah didapat.

Hal ini juga didukung oleh tingginya nilai *accuracy*, yaitu mencapai 97.5% untuk data *training* dan 90% untuk data *testing*. Ini menunjukkan bahwa model yang diperoleh mampu melakukan klasifikasi dengan baik. Pergerakan *loss* yang mendekati nilai nol atau rendah dan pergerakan *accuracy* yang terus meningkat menunjukkan hasil yang baik seiring dengan berjalannya *epoch*. Kondisi berhentinya proses pelatihan ditentukan dari banyaknya *epoch* yang digunakan, dalam hal ini yaitu sebanyak 50 *epoch*. Proses pembelajaran akan dihentikan jika sudah memenuhi kondisi tersebut.

Hasil klasifikasi yang didapatkan dari data *training* dan *testing* ditunjukkan melalui tabel *confusion matrix* seperti berikut ini.

Tabel 5.2 Hasil Klasifikasi Data Training

		Actual	
		0	1
Predicted	0	39	1
	1	1	39

Tabel 5.3 Hasil Klasifikasi Data Testing

		Actual	
		0	1
Predicted	0	9	1
	1	1	9

Pada tabel 5.3 dan 4 ditunjukkan nilai *true positive*, *true negative*, *false positive*, dan *false negative*. Untuk data *training*, citra tomat layak yang berhasil diklasifikasi adalah sebanyak 39 dari 40 citra yang digunakan. Begitu juga dengan citra tomat tidak layak yang memiliki kesalahan 1 klasifikasi dari 40 citra, 39 di antaranya berhasil diklasifikasi. Jumlah kesalahan yang sama juga didapatkan saat melakukan klasifikasi dari data *testing*. Setiap kelas memiliki kesalahan klasifikasi sebesar 1 citra dari masing-masing 10 citra yang digunakan. Penentuan klasifikasi diperoleh dari nilai probabilitas tertinggi antara citra tomat layak dan

tidak layak. Apabila nilai probabilitas citra tomat layak lebih tinggi, maka objek yang diuji masuk ke dalam kategori atau kelas tomat layak, begitu sebaliknya. Nilai probabilitas yang dihasilkan data *training* dan *testing* dapat dilihat pada lampiran 2 dan 3.

Selain mengamati dan melakukan pengujian terhadap hasil klasifikasi dari data *training* dan data *testing*, peneliti juga melakukan pengujian terhadap model menggunakan data baru sebanyak 10 data yang terdiri dari 5 data citra tomat layak dan 5 citra tomat tidak layak. Hasil klasifikasi yang didapatkan yaitu semua citra berhasil diklasifikasi dengan benar yang ditunjukkan melalui tabel *confusion matrix* seperti berikut.

Tabel 5.4 Hasil Klasifikasi Data Uji

		Actual	
		0	1
Predicted	0	5	0
	1	0	5

Dengan kata lain, nilai *accuracy* dalam pengujian data baru ini adalah sebesar 100%, dikarenakan tidak ada data yang salah dalam proses pengklasifikasian. Probabilitas setiap kelas dapat dilihat pada gambar 5.19.

	Bagus	Busuk	Predicted_class	Actual
[1,]	0.8672740	0.13272597	0	0
[2,]	0.9131988	0.08680122	0	0
[3,]	0.7923874	0.20761260	0	0
[4,]	0.7200665	0.27993351	0	0
[5,]	0.6892641	0.31073600	0	0
[6,]	0.3921436	0.60785633	1	1
[7,]	0.2986507	0.70134932	1	1
[8,]	0.2982249	0.70177513	1	1
[9,]	0.1871694	0.81283063	1	1
[10,]	0.2912801	0.70871997	1	1

Gambar 5.19 Probabilitas Data Uji

Dari penelitian yang dilakukan, didapatkan bahwa hasil akurasi untuk data *training*, *testing*, maupun uji coba dengan data baru menghasilkan akurasi yang tinggi, maka dapat dikatakan bahwa metode ini baik untuk melakukan klasifikasi citra tomat. Sehingga, dengan adanya penelitian ini sehingga bisa menjadi

alternatif lain untuk melakukan klasifikasi tomat selain menggunakan cara manual.

5.5. Penentuan Parameter CNN

Hasil pengolahan citra di atas merupakan model terbaik yang didapatkan peneliti. Model terbaik ini didapatkan dari nilai terbaik parameter-parameter dalam CNN. Parameter yang dimaksud antara lain adalah ukuran kernel dan jumlah neuron. Peneliti sebelumnya melakukan perbandingan parameter untuk melihat bagaimana hasil akurasi dari model yang terbentuk. Selain itu, peneliti juga melihat pengaruh jumlah data terhadap akurasi serta skenario data *training* dan data *testing* dalam membentuk model. Adapun hasil perbandingannya adalah seperti berikut ini.

5.5.1 Penentuan Ukuran Filter

Untuk menentukan ukuran filter/kernel yang tepat, percobaan dilakukan dengan menggunakan ukuran filter yang berbeda-beda. Ukuran filter yang sering digunakan pada umumnya yaitu 3x3, 5x5, dan 7x7. Semakin kecil ukuran filter yang digunakan menjadikan informasi yang didapatkan juga lebih banyak.

Tabel 5.5 *Penentuan Ukuran Filter*

Ukuran Filter	Akurasi Testing
3x3	90%
5x5	50%

Hasil percobaan ditunjukkan pada tabel 5.5. Pada tabel tersebut, ukuran filter yang menghasilkan hasil terbaik adalah 3x3. Berkurangnya akurasi dengan bertambahnya ukuran filter disebabkan berkurangnya parameter pelatihan. Ketika ukuran filter 3x3, maka jumlah parameter yang digunakan selama pelatihan yaitu 475.938 (Gambar 5.17). Ketika ukurannya 5x5 menjadikan parameter pelatihan lebih sedikit. Ukuran filter 7x7 tidak digunakan karena ukuran tersebut terlalu besar, sehingga tidak cocok untuk melakukan proses konvolusi dengan jumlah layer 4 serta beberapa parameter lainnya.

5.5.2 Penentuan Jumlah Neuron pada Lapisan Tersembunyi

Percobaan dilanjutkan dengan menentukan neuron yang optimal pada lapisan tersembunyi. Dengan ukuran masukan yaitu model terbaik penelitian ini, jumlah neuron pada *hidden layer* dirubah untuk melihat pengaruh perubahan ukuran terhadap akurasi. Hasil percobaan ditunjukkan pada tabel 5.3 berikut.

Tabel 5.6 *Penentuan Jumlah Neuron pada Lapisan Tersembunyi*

Jumlah Neuron	Akurasi Testing
100	50%
175	80%
200	50%
256	90%

Jumlah neuron yang memberikan akurasi terbaik adalah 256 neuron dengan akurasi 90%. Jumlah neuron sebanyak 256 merupakan *default* dari model CNN pada Keras. Penelitian lain juga banyak yang menggunakan jumlah neuron sebanyak itu. Jumlah neuron sebanyak 175 juga memberikan akurasi yang cukup baik dibandingkan dengan jumlah neuron sebanyak 100 dan 200. Penambahan atau pengurangan jumlah neuron pada lapisan tersembunyi dapat mengurangi akurasi yang didapatkan oleh sistem.

5.5.3 Pengaruh Jumlah Data Terhadap Akurasi

Percobaan dilakukan untuk melihat pengaruh jumlah data latih terhadap akurasi dari CNN. Percobaan ini dilakukan dengan menggunakan data latih dengan jumlah 60, 80, dan 100. Model yang digunakan sama, yaitu model terbaik dalam penelitian ini, menggunakan 4 *layer* konvolusi. Skenario perbandingan data *training* dan data *testing* yang digunakan sama untuk setiap jumlah satanya yaitu 80:20, dimana 80% digunakan untuk data *training* dan 20% digunakan untuk data *testing*.

Tabel 5.7 *Pengaruh Jumlah Data Terhadap Akurasi*

Jumlah Data	Akurasi Testing
60	66.67%
80	87.5%
100	90%

Tabel 5.7 di atas menunjukkan hasil dari percobaan dimana semakin banyak jumlah data latih, maka akurasi menjadi semakin baik.

5.5.4 Pengaruh Skenario Pelatihan dan Pengujian Data

Skenario penggunaan data *training* dan data *testing* dalam pembentukan model juga diperhatikan. Berikut perbandingan skenario penggunaan data yang digunakan. Jumlah data untuk setiap skenarionya sama, yaitu sebanyak 100 data. Hasil dari skenario yang dibuat ditunjukkan pada tabel 5.4 berikut.

Tabel 5.4 *Skenario Pelatihan dan Pengujian Data*

Jumlah Data	Skenario Data <i>Training</i> : Data <i>Testing</i>	Jumlah Data<i>Training</i> : Data <i>Testing</i>	Akurasi Testing
100	60% : 40%	60:40	50%
100	70% : 30%	70:30	50%
100	80% : 20%	80:20	90%

Hasil terbaik didapatkan dengan menggunakan skenario 80%:20% untuk data *training* dan data *testing*. Hal ini dikarenakan proses pembelajaran dilakukan dengan data latih yang lebih banyak, sehingga sistem akan belajar lebih banyak dibandingkan dengan skenario lainnya.

BAB VI

PENUTUP

5.1. Kesimpulan

Berdasarkan hasil analisis yang telah dilakukan, diperoleh beberapa kesimpulan sebagai berikut :

1. Implementasi metode *Convolutional Neural Network (CNN)* untuk klasifikasi citra tomat dilakukan menggunakan *package Keras* pada *software RStudio* versi 1.1.383. Banyaknya *layer* konvolusi yang digunakan yaitu sebanyak empat *layer*, fungsi aktivasi yang digunakan yaitu *ReLU*, dan beberapa penggunaan parameter lainnya.
2. Tingkat akurasi data *testing* yang didapatkan dari model yang terbentuk yaitu sebesar 90% dalam melakukan klasifikasi citra tomat.
3. Hasil klasifikasi dari data baru sebanyak 10 citra tomat untuk menguji model yang terbentuk yaitu semua citra berhasil diklasifikasi dengan benar.

5.2. Saran

Berdasarkan kesimpulan di atas, dapat diberikan beberapa saran sebagai berikut :

1. Tahapan *preprocessing* yang dilakukan juga dengan menambahkan proses segmentasi serta penghilangan *noise* pada citra dengan bantuan komputer untuk mendapatkan kualitas citra yang lebih baik.
2. Klasifikasi citra tomat bisa dilakukan berdasarkan indikator yang lain, misalnya tingkat kematangan tomat, perbandingan antara citra RGB dan *grayscale*, dan lain-lain.
3. Data citra yang digunakan diperbanyak untuk melatih model dan mencapai tingkat akurasi yang tinggi.
4. Spesifikasi komputer yang digunakan lebih tinggi, yaitu dengan menggunakan komputer dengan *Random Access Memory (RAM)* yang tinggi dan *Graphics Processing Unit (GPU)*.

5. Perbandingan antara nilai jumlah filter, *learning rate*, *epoch*, *layer dropout*, serta fungsi aktivasi yang digunakan bisa menjadi penelitian lebih lanjut untuk melihat bagaimana pengaruhnya terhadap tingkat akurasi yang diperoleh.

DAFTAR PUSTAKA

- Achmad, Ir. Balza. 2006. *Kecerdasan Buatan*. Diktat Mata Kuliah. Jurusan Teknik Fisika Fakultas Teknik Universitas Gadjah Mada.
- Agian, Doli Garesya, et.al. *Identifikasi Kematangan Buah Markisa (Passiflora Edulis) dengan Pengolahan Citra Menggunakan Jaringan Saraf Tiruan*. J. Rekayasa Pangan dan Pert., Vol. 3 No. 3 Tahun 2015
- Amaliah, Riska. 2011. *Pengolahan Citra, Grafik Komputer, dan Komputer Vision*. Diakses pada tanggal 27 Januari 2018 pukul 06.44 WIB dari <http://www.riskaamaliah.com/2011/11/pengolahan-citra-grafik-komputer-dan.html>
- Anggriawan, Mochamad Angga. 2017. *Pengenalan Tingkat Kematangan Tomat Berdasarkan Citra Warna pada Studi Kasus Pembangunan Sistem Pemilihan Otomatis*. Jurnal Teknik Informatika dan Sistem Informasi Volume 3 Nomor 3 Desember 2017
- Binus University. 2012. *Dasar Pemahaman Neural Network*. Diakses pada tanggal 27 Januari 2018 pukul 09.34 WIB dari <https://socs.binus.ac.id/2012/07/26/konsep-neural-network/>
- Bustomi, M. Arief (2014). *Analisis Distribusi Intensitas GRB Citra Digital untuk Klasifikasi Kualitas Biji Jagung Menggunakan Jaringan Syaraf Tiruan*. Jurnal Fisika dan Aplikasinya Volume 10, Nomor 3, Oktober 2014
- CRAN. *Keras: R Interface to 'Keras'*. Diakses pada tanggal 08 Februari 2018 pukul 02.22 WIB dari <https://cran.r-project.org/web/packages/keras/index.html>
- Dahria, Muhammad. 2008. *Kecerdasan Buatan (Artificial Intelligence)*. Jurnal SAINTIKOM Vol. 5, No. 2 Agustus 2008
- Danukusumo, Kevin Pudi. 2017. *Implementasi Deep Learning Menggunakan Convolutional Neural Network untuk Klasifikasi Citra Candi Berbasis GPU*. Tugas Akhir. Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Atma Jaya Yogyakarta

- Deswari, Dila., et.al. 2013. *Identifikasi Kematangan Buah Tomat Menggunakan Metoda Backpropagation*. Jurusan Sistem Komputer, Fakultas Teknologi Informasi, Universitas Andalas.
- Effendi, Mas'ud., et.al. 2017. *Identifikasi Jenis dan Mutu Teh Menggunakan Pengolahan Citra Digital dengan Metode Jaringan Syaraf Tiruan*. Jurnal Teknotan Vol. 11 No. 2 Agustus 2017
- Effendy. *Deteksi Pornografi pada Citra Digital Menggunakan Pengolahan Citra dan Jaringan Saraf Tiruan*. Jurusan Teknik Fisika Fakultas Teknik Universitas Gadjah Mada
- Endarko., et.al. 2005. *Aplikasi Pengolahan Citra dan Jaringan Saraf Tiruan untuk Pengenalan Pola Tulisan Tangan*. Jurnal Fisika dan Aplikasinya Volume 1 Nomer 2 Juli 2005
- Fajarfika, Resti. 2014. *Kajian Serologi dan Molekul Crinivirus pada Tanaman Tomat*. Tesis. Jurusan Fitopatologi, Universitas Gadjah Mada
- Familinia. *10 Manfaat Tomat: Selain untuk Kecantikan, Ternyata Tomat Punya Manfaat Lain yang Tak Kalah Penting*. Diakses pada tanggal 07 Februari 2018 pukul 22.29 WIB dari <https://familinia.com/khasiat-dan-manfaat-tomat/>
- Ferdiansyah, M. Riza. 2006. *Sistem Seleksi Kematangan Buah Tomat Waktu-Ntara Berbasis Nilai RGB*. Jurnal TELKOMNIKA ISSN: 1693-6930
- Godliving, Nela. 2013. *Contoh Penerapan Teknologi Computer Vision*. Diakses pada tanggal 27 Januari 2018 pukul 02.43 dari <https://www.scribd.com/doc/135018403/CONTOH-PENERAPAN-TEKNOLOGI-COMPUTER-VISION-docx>
- Hestningsih, Idhawati. *I. Pengantar Kecerdasan Buatan*. Universitas Dian Nuswantoro, Semarang
- Hidayatullah, Aisyah. 2013. *Identifikasi Tingkat Kematangan Buah Tomat (Lycopersicon Esculentum Mill) Menggunakan Metode Pengolahan Citra dan Jaringan Saraf Tiruan*. Tugas Akhir. Jurusan Teknologi Industri Pertanian Fakultas Teknologi Pertanian Universitas Gadjah Mada

- Info Komputer. 2017. *Inilah 3 Tantangan Teknologi Kecerdasan Buatan di Indonesia*. Diakses pada tanggal 27 Januari 2018 pukul 02.04 WIB dari <https://infokomputer.grid.id/2017/07/berita/berita-reguler/3-tantangan-teknologi-kecerdasan-buatan-ai-di-indonesia/>
- Institut Teknologi Sepuluh Nopember. *Jaringan Syarat Tiruan*. Diakses pada tanggal 28 Januari 2018 pukul 08.02 WIB dari http://share.its.ac.id/pluginfile.php/14163/mod_resource/content/1/jst1.pdf
- Institut Teknologi Bandung. *Warna*. Diakses pada tanggal 07 Februari 2018 pukul 23.53 WIB dari [file:///C:/Users/ASUS/Downloads/Bab-6 Histogram%20Citra%20\(1\).pdf](file:///C:/Users/ASUS/Downloads/Bab-6%20Histogram%20Citra%20(1).pdf)
- Januarsjaf, Aswin. 2017. *RStudio*. Diakses pada tanggal 08 Februari 2018 pukul 02.07 WIB dari https://rstudio-pubs-static.s3.amazonaws.com/242015_93ab8636286c4252984f42bf05ab918f.html
- Kementerian Pertanian. 2015. *Statistik Produksi Holtikultura Tahun 2015*. Direktorat Jenderal Holtikultura : Jakarta Selatan
- Kusumaningtyas, Sella., et.al. 2016. *Identifikasi Kematangan Buah Tomat Berdasarkan Warna Menggunakan Metode Jaringan Syaraf Tiruan (JST)*. Jurnal Informatika Polinema ISSN : 2407-070X
- Library Binus. <https://library.binus.ac.id/eColls/eThesisdoc/Bab2/2010-2-00250-if%20bab%202.pdf>
- Novianto, Ary. 2009. *Klasifikasi Tingkat Kematangan Varietas Tomat Merah dengan Metode Perbandingan Kadar Warna*. Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Gadjah Mada Yogyakarta
- Nurhayati, Siti. 2017. *Produksi Tanaman Tomat (Lycopersicum Esculentum Mill.) F1 Hasil Induksi Medan Magnet yang Diinfeksi Dusarium Oxysporum f.sp. Lycopersici*. Tugas Akhir. Jurusan Biologi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung
- Nurmansyah. 2015. *R, Bahasa Pemrograman untuk Analisis Data dan Statistik*. Diakses pada tanggal 08 Februari 2018 pukul 02.12 WIB dari

https://www.kompasiana.com/nurmansyah/r-bahasa-pemrograman-untuk-analisis-data-dan-statistik_552fab4f6ea8349b138b456f

Martiana, Entin. *Introduction to Artificial Intelligence*. Diakses pada tanggal 27 Januari 2018 pukul 02.18 WIB dari

<http://yuliana.lecturer.pens.ac.id/Kecerdasan%20Buatan/ppt/Pengenalan%20AI/Minggu1%20-%20Introduction%20to%20Artificial%20Intelligence.pdf>

Mukhti, Ihsan Nugraha Putra. 2015. *Sistem Otomasi dalam Penyortiran Tomat dengan Image Processing Menggunakan Metode Deteksi RGB*. Universitas Telkom, Bandung.

Politeknik Elektronika Negeri Surabaya. *Jaringan Syaraf Tiruan (Neural Network)*. Diakses pada tanggal 28 Januari 2018 pukul 07.58 WIB dari <http://entin.lecturer.pens.ac.id/Kecerdasan%20Buatan/Buku/Bab%208%20Jaringan%20Syaraf%20Tiruan.pdf>

Pratomo, Aditya Gema. 2017. *Perkembangan Kecerdasan Buatan Tak Bisa Dibendung*. Diakses pada tanggal 27 Januari 2018 pukul 01.47 WIB dari <https://www.suara.com/tekno/2017/11/30/182350/perkembangan-kecerdasan-buatan-tak-bisa-dibendung>

Program Fitnes. *Dibalik Cerahnya Tomat*. Diakses pada tanggal 07 Februari 2018 pukul 22.30 WIB dari <http://programfitnes.com/dibalik-cerahnya-tomat/>

Ramdani, Irpan. 2012. *Pengertian Artificial Intelligence (AI)/Kecerdasan Buatan*. Diakses pada tanggal 27 Januari 2018 pukul 02.29 WIB dari <https://irpantips4u.blogspot.co.id/2012/12/konsep-dasar-artificial-intelligence-ai.html>

Rohman, Nanan., et.al. 2008. *Pencapaian Goal State Pada Permainan 8 Puzzle dengan Menggunakan Metode Best First Search*. Jurnal Computech & Bisnis, Vol. 2, No. 2, Desember 2008, 66-79.

Riska, Yulia Suastika., et.al (2016). *Klasifikasi Level Kematangan Buah Tomat Berdasarkan Fitur Warna Menggunakan Multi-SVM*. Jurnal Ilmiah Informatika Volume 1 No. 1/Desember 2016.

- Sena, Samuel. 2017. *Pengenalan Deep Learning Part 7: Convolutional Neural Networks (CNN)*. Diakses pada tanggal 08 Februari 2018 pukul 00.20 WIB dari <https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutional-neural-network-cnn-b003b477dc94>
- Somantri, Agus Supriatna., et.al. 2013. *Identifikasi Mutu Fisik Beras dengan Menggunakan Teknologi Pengolahan Citra dan Jaringan Syaraf Tiruan*. J. Pascapanen 10(2). 2013 : 95 - 103
- Suartika, I. W., et.al. 2016. *Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) pada Caltect 101*. Jurnal Teknik ITS Vol. 5, No. 1, (2016) ISSN: 2337-3539 (2301-9271 Print)
- Subha, Ghazali Eko., et.al. 2014. *Aplikasi Kematangan Tomat Berdasarkan Warna dengan Metode Linear Discriminant Analysis (LDA)*. Universitas Brawijaya, Malang
- STMIK Jakarta STI&K. *Jaringan Syaraf Tiruan*. Diakses pada tanggal 28 Januari 2018 pukul 08.45 WIB dari http://jak-stik.ac.id/e-learning/MATERI04/Kecerdasan_Buatan/11-Jaringan-Syaraf-Tiruan.pdf
- Universitas Brawijaya. *Architecture Net, Simple Neural Net*. Diakses pada tanggal 28 Januari 2018 pukul 08.08 WIB dari http://share.its.ac.id/pluginfile.php/14163/mod_resource/content/1/jst1.pdf
- Universitas Gunadarma. *Histogram Citra*. Diakses pada tanggal 07 Februari 2018 pukul 23.41 WIB dari [file:///C:/Users/ASUS/Downloads/Bab-6_Histogram%20Citra%20\(1\).pdf](file:///C:/Users/ASUS/Downloads/Bab-6_Histogram%20Citra%20(1).pdf)
- Universitas Sumatera Utara. *Jaringan Syarat Tiruan*. Diakses pada tanggal 28 Januari 2018 pukul 08.12 WIB dari <http://repository.usu.ac.id/bitstream/handle/123456789/47283/Chapter%20II.pdf?sequence=4&isAllowed=y>
- Wijanarko, Rony., et.al. *Deteksi Wajah Berbasis Segmentasi Warna Kulit Menggunakan Ruang Warna YCbCr & Template Matching*. Jurnal Ilmiah Cendekia Eksakta ISSN 2528-5912
- Wikipedia. *Tomat*. Diakses pada tanggal 02 Februari 2018 pukul 23.10 WIB dari <https://id.wikipedia.org/wiki/Tomat>

- Zufar, Muhammad., et.al. 2016. *Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time*. Jurnal Sains dan Seni ITS Vol. 5 No. 2 (2016) 2337-3520 (2301-928X Print)
- Zunaidi, Muhammad., et.al. 2015. *Faktor Penentu Tingkat Keberhasilan Sistem Deteksi Wajah pada Citra Digital*. Jurnal Ilmiah Saintikom Sains dan Komputer ISSN : 1978-6603

LAMPIRAN

Lampiran 1 Script R untuk Klasifikasi Menggunakan CNN

```
# Convolutional Neural Networks
setwd('D://Tomat100')
# Load packages
library(keras)
library(EBImage)

save_in <- ("D://Run100/")
# Load images names

images <- list.files()
# Main loop resize images and set them to greyscale
w <- 100
h <- 100
for(i in 1:length(images))

{

  # Try-catch is necessary since some images
  # may not work.

  result <- tryCatch({

    # Image name

    imgname <- images[i]

    # Read image

    img <- readImage(imgname)

    # Path to file
    img_resized <- resize(img, w = w, h = h)

    path <- paste(save_in, imgname, sep = "")

    # Save image

    writeImage(img_resized, path, quality = 70)

    # Print status

    print(paste("Done",i,sep = " ")),

    # Error function
```

```

    error = function(e){print(e)})
}

setwd("D://Run100")

# Read Images
images <- list.files()
images
summary(images)

list_of_images = lapply(images, readImage)
head(list_of_images)
display(list_of_images[[15]])
#create train
train <- list_of_images[c(1:40, 51:90)]
str(train)
display(train[[28]])

#create test
test <- list_of_images[c(41:50, 91:100)]
test
display(test[[1]])

par(mfrow = c(10,10))
for (i in 1:80) plot(train[[i]])

# Resize & combine
str(train)
for (i in 1:80) {train[[i]] <- resize(train[[i]], 32, 32)}
for (i in 1:20) {test[[i]] <- resize(test[[i]], 32, 32)}

for (f in 1:80) {print(dim(train[[f]]))}

train <- combine(train)
str(train)
x <- tile(train, 5)
display(x, title='Pictures')

test <- combine(test)
y <- tile(test, 2)
display(y, title = 'Pics')

# Reorder dimension
train <- aperm(train, c(4,1,2,3))
test <- aperm(test, c(4,1,2,3))
str(train)
str(test)

rep(0,5)
# Response

```

```

trainy <- c(rep(0,40),rep(1,40))
testy <- c(rep(0,10),rep(1,10))

# One hot encoding
trainLabels <- to_categorical(trainy)
testLabels <- to_categorical(testy)

# Model
model <- keras_model_sequential()

model %>%
  layer_conv_2d(filters = 32,
                kernel_size = c(3,3),
                activation = 'relu',
                input_shape = c(32, 32, 3)) %>%
  layer_conv_2d(filters = 32,
                kernel_size = c(3,3),
                activation = 'relu') %>%
  layer_max_pooling_2d(pool_size = c(2,2)) %>%
  layer_dropout(rate = 0.01) %>%
  layer_conv_2d(filters = 64,
                kernel_size = c(3,3),
                activation = 'relu') %>%
  layer_conv_2d(filters = 64,
                kernel_size = c(3,3),
                activation = 'relu') %>%
  layer_max_pooling_2d(pool_size = c(2,2)) %>%
  layer_dropout(rate = 0.01) %>%
  layer_flatten() %>%
  layer_dense(units = 256, activation = 'relu') %>%
  layer_dropout(rate=0.01) %>%
  layer_dense(units = 2, activation = 'softmax') %>%

  compile(loss = 'categorical_crossentropy',
          optimizer = optimizer_sgd(lr = 0.01,
                                    decay = 1e-6,
                                    momentum = 0.9,
                                    nesterov = T),

          metrics = c('accuracy'))
summary(model)

# Fit model
history <- model %>%
  fit(train,
      trainLabels,
      epochs = 50,
      batch_size = 32,
      validation_split = 0.2,
      validation_data = list(test, testLabels))
plot(history)

```

```

# Evaluation & Prediction - train data
model %>% evaluate(train, trainLabels)
pred <- model %>% predict_classes(train)
table(Predicted = pred, Actual = trainy)

prob <- model %>% predict_proba(train)
cbind(prob, Predicted_class = pred, Actual = trainy)

# Evaluation & Prediction - test data
model %>% evaluate(test, testLabels)
pred <- model %>% predict_classes(test)
table(Predicted = pred, Actual = testy)

prob <- model %>% predict_proba(test)
cbind(prob, Predicted_class = pred, Actual = testy)

#save model
save_model_hdf5(model,filepath='D://Run100.hdf5')

#Pengujian Model
setwd('D:/Uji/')

model=load_model_hdf5(filepath='D://Run100.hdf5')
images <- list.files()
images
summary(images)

list_of_images = lapply(images, readImage)
head(list_of_images)
display(list_of_images[[3]])
# Get the image as a matrix
testt <- list_of_images

for (i in 1:10) {testt[[i]] <- resize(testt[[i]], 32, 32)}
for (i in 1:10) {testt[[i]] <- toRGB(testt[[i]])}

fixx <- combine(testt)
y <- tile(fixx, 2)
display(y, title = 'Pics')

str(fixx)
Uji <- aperm(fixx, c(4, 1, 2, 3))
str(Uji)

testy <- c(rep(0,5), rep(1,5))

# One hot encoding
testLabels <- to_categorical(testy)

```

```
pred <- model %>% predict_classes(Uji)
table(Predicted = pred, Actual = testy)

model %>% evaluate(Uji, testLabels)

prob <- model %>% predict_proba(Uji)
prob
colnames(prob)<- c('Bagus', 'Busuk')
Tiara<-cbind(prob, Predicted_class = pred, Actual = testy)
Tiara
```

Lampiran 2 Probabilitas Citra pada Data Training

```
> cbind(prob, Predicted_class = pred, Actual = trainy)
      Predicted_class Actual
[1,] 0.8737766 0.12622343      0      0
[2,] 0.8370513 0.16294876      0      0
[3,] 0.8006354 0.19936462      0      0
[4,] 0.8870764 0.11292362      0      0
[5,] 0.8038660 0.19613402      0      0
[6,] 0.8953133 0.10468666      0      0
[7,] 0.8988026 0.10119746      0      0
[8,] 0.7428121 0.25718793      0      0
[9,] 0.8183152 0.18168481      0      0
[10,] 0.8638702 0.13612977      0      0
[11,] 0.7613391 0.23866086      0      0
[12,] 0.5601127 0.43988720      0      0
[13,] 0.8353794 0.16462062      0      0
[14,] 0.6673064 0.33269367      0      0
[15,] 0.8651581 0.13484192      0      0
[16,] 0.6528507 0.34714928      0      0
[17,] 0.8344434 0.16555655      0      0
[18,] 0.6955613 0.30443874      0      0
[19,] 0.6617920 0.33820799      0      0
[20,] 0.7420762 0.25792369      0      0
[21,] 0.7155682 0.28443182      0      0
[22,] 0.7663691 0.23363088      0      0
[23,] 0.8065594 0.19344063      0      0
[24,] 0.6942354 0.30576459      0      0
[25,] 0.7243642 0.27563575      0      0
[26,] 0.8621627 0.13783723      0      0
[27,] 0.7908984 0.20910157      0      0
[28,] 0.9146817 0.08531834      0      0
[29,] 0.6457630 0.35423699      0      0
[30,] 0.8334321 0.16656782      0      0
[31,] 0.5496712 0.45032883      0      0
[32,] 0.8566043 0.14339568      0      0
[33,] 0.7625030 0.23749693      0      0
[34,] 0.4956425 0.50435752      1      0
[35,] 0.7349216 0.26507837      0      0
[36,] 0.6703793 0.32962066      0      0
[37,] 0.5084631 0.49153689      0      0
[38,] 0.5792546 0.42074540      0      0
[39,] 0.8113568 0.18864314      0      0
[40,] 0.7312427 0.26875740      0      0
[41,] 0.4037736 0.59622633      1      1
[42,] 0.1951157 0.80488431      1      1
[43,] 0.1301855 0.86981452      1      1
[44,] 0.3717985 0.62820154      1      1
[45,] 0.4872248 0.51277524      1      1
[46,] 0.5269876 0.47301242      0      1
[47,] 0.2406974 0.75930262      1      1
[48,] 0.3539308 0.64606923      1      1
[49,] 0.2337475 0.76625246      1      1
[50,] 0.2458449 0.75415516      1      1
[51,] 0.2443620 0.75563800      1      1
[52,] 0.3627666 0.63723338      1      1
[53,] 0.2124900 0.78751004      1      1
[54,] 0.1253962 0.87460375      1      1
[55,] 0.1660678 0.83393216      1      1
```


[56,]	0.1554565	0.84454346	1	1
[57,]	0.1383177	0.86168230	1	1
[58,]	0.1742526	0.82574737	1	1
[59,]	0.1453165	0.85468358	1	1
[60,]	0.1430621	0.85693789	1	1
[61,]	0.2110923	0.78890771	1	1
[62,]	0.3459404	0.65405953	1	1
[63,]	0.2388523	0.76114768	1	1
[64,]	0.2341483	0.76585168	1	1
[65,]	0.2868594	0.71314067	1	1
[66,]	0.3291027	0.67089731	1	1
[67,]	0.1583741	0.84162599	1	1
[68,]	0.1552423	0.84475774	1	1
[69,]	0.1838128	0.81618720	1	1
[70,]	0.1914502	0.80854982	1	1
[71,]	0.1887667	0.81123328	1	1
[72,]	0.1970770	0.80292290	1	1
[73,]	0.2104821	0.78951782	1	1
[74,]	0.4397347	0.56026536	1	1
[75,]	0.1070890	0.89291102	1	1
[76,]	0.0942388	0.90576124	1	1
[77,]	0.3158497	0.68415034	1	1
[78,]	0.2056593	0.79434073	1	1
[79,]	0.4844573	0.51554275	1	1
[80,]	0.2100907	0.78990924	1	1

Lampiran 3 *Probabilitas Citra pada Data Testing*

```
> cbind(prob, Predicted_class = pred, Actual = testy)
```

	prob	Predicted_class	Actual
[1,]	0.8327861 0.1672139	0	0
[2,]	0.7394238 0.2605762	0	0
[3,]	0.7795511 0.2204488	0	0
[4,]	0.6448731 0.3551268	0	0
[5,]	0.1991311 0.8008689	1	0
[6,]	0.5812714 0.4187285	0	0
[7,]	0.5507752 0.4492248	0	0
[8,]	0.6328692 0.3671308	0	0
[9,]	0.5687083 0.4312917	0	0
[10,]	0.7299782 0.2700218	0	0
[11,]	0.3919158 0.6080843	1	1
[12,]	0.2201188 0.7798812	1	1
[13,]	0.1987032 0.8012967	1	1
[14,]	0.5742494 0.4257506	0	1
[15,]	0.2038143 0.7961857	1	1
[16,]	0.4024925 0.5975075	1	1
[17,]	0.3301020 0.6698979	1	1
[18,]	0.2343587 0.7656413	1	1
[19,]	0.2599685 0.7400314	1	1
[20,]	0.1885805 0.8114195	1	1