

**APLIKASI PENDETEKSIAN OBJEK BUAH-BUAHAN YANG MEMILIKI
KEMIRIPAN MENGGUNAKAN ALGORITMA FASTER R-CNN BERBASIS
ANDROID**

SKRIPSI

FERDINAN MULYANTO

151402056



**PROGRAM STUDI S1 TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2020**

APLIKASI PENDETEKSIAN OBJEK BUAH-BUAHAN YANG MEMILIKI
KEMIRIPAN MENGGUNAKAN ALGORITMA FASTER R-CNN BERBASIS
ANDROID

SKRIPSI

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah
Sarjana Teknologi Informasi

FERDINAN MULYANTO

151402056



PROGRAM STUDI S1 TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

MEDAN

2020

PERSETUJUAN

Judul : APLIKASI PENDETEKSIAN OBJEK BUAH-
BUAHAN YANG MEMILIKI KEMIRIPAN
MENGUNAKAN ALGORITMA FASTER R-CNN
BERBASIS ANDROID

Kategori : SKRIPSI

Nama : FERDINAN MULYANTO

Nomor Induk Mahasiswa : 151402056

Program Studi : SARJANA (S1) TEKNOLOGI INFORMASI

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

Diluluskan di

Medan, 28 Januari 2020

Komisi Pembimbing :

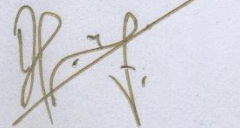
Pembimbing 2



Prof. Dr. Drs. Opim Salim Sitompul, M.Sc.

NIP. 19610817198701 1 001

Pembimbing 1



Dedy Arisandi, ST., M.Kom.

NIP. 19790831200912 1 002

Diketahui/disetujui oleh

Program Studi S1 Teknologi Informasi

Ketua,




Romi Fadillah Rahmat, B.Comp.Sc., M.Sc.

NIP. 19860303201012 1 004

PERNYATAAN


UCAPAN TERIMA KASIH

APLIKASI PENDETEKSIAN OBJEK BUAH-BUAHAN YANG MEMILIKI KEMIRIPAN MENGGUNAKAN ALGORITMA FASTER R-CNN BERBASIS ANDROID

SKRIPSI

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 28 Januari 2020



FERDINAN MULYANTO

151402056

UCAPAN TERIMA KASIH

Puji dan syukur penulis ucapkan kepada Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul “Aplikasi Pendeteksian Objek Buah-Buahan Yang Memiliki Kemiripan Menggunakan Algoritma Faster R-CNN Berbasis Android”, yang merupakan salah satu syarat untuk memperoleh gelar Sarjana Komputer di Fakultas Ilmu Komputer dan Teknologi Universitas Sumatera Utara. Terima kasih yang tak terhingga penulis ucapkan kepada kedua orangtua, Drs. Bonar Pardede dan Romasnida Sormin, S.E. yang selalu memberikan doa, dukungan, motivasi serta materil sehingga skripsi ini dapat terselesaikan. Penulis juga mengucapkan terima kasih kepada banyak pihak yang telah membantu kelancaran penulisan skripsi ini, diantaranya:

1. Bapak Prof. Dr. Runtung Sitepu, SH., M.Hum selaku Rektor Universitas Sumatera Utara.
2. Bapak Prof. Dr. Drs. Opim Salim Sitompul, M.Sc. selaku Dekan Fasilkom-TI Universitas Sumatera Utara.
3. Bapak Romi Fadhilah Rahmat, B.Comp.Sc., M.Sc. selaku Ketua Program Studi S1 Teknologi Informasi Universitas Sumatera Utara.
4. Bapak Dedy Arisandi, ST., M.Kom. selaku Dosen Pembimbing I yang telah memberikan saran dan bimbingan kepada penulis.
5. Bapak Prof. Dr. Drs. Opim Salim Sitompul, M.Sc. selaku Dosen Pembimbing II yang telah memberikan bimbingan dan saran kepada penulis.
6. Bapak Indra Aulia, S.TI., M.Kom selaku Dosen Penguji I yang telah memberikan kritik dan saran untuk penyempurnaan skripsi ini.
7. Bapak Fahrurrozi Lubis, B.IT., M.Sc.IT selaku Dosen Penguji II yang telah memberikan kritik dan saran untuk penyempurnaan skripsi ini.
8. Orangtua yang sangat penulis sayangi, Drs. Bonar Pardede dan Romasnida Sormin, S.E.
9. Kakak penulis, Albert Marudut Pardede yang selalu memberikan doa, dukungan dan motivasi kepada penulis.

10. Kepada sahabat seperjuangan saya Bayu Prabowo, dan sahabat saya yang tidak dapat disebutkan satu persatu, yang telah memberikan motivasi dan semangat pada penulis.
11. Seluruh teman-teman angkatan 2015 yang tidak dapat penulis sebutkan satu persatu, yang telah menyediakan waktunya untuk bertukar pikiran kepada penulis.

Semoga Tuhan Yang Maha Esa memberikan berkah dan rahmat yang berlimpah kepada semua pihak yang telah memberikan bantuan, dukungan, serta doa kepada penulis pada saat pembuatan skripsi ini.

Medan, 28 Januari 2020

Penulis

ABSTRAK

Pendeteksian objek merupakan salah satu hal yang penting dalam pengotomatisan sistem. Dalam skripsi ini penulis membangun suatu sistem yang dapat mengidentifikasi dan mendeteksi objek buah-buahan yang memiliki kemiripan fisik yang diterapkan pada android. Sistem ini dapat membantu untuk pengidentifikasian dan pendeteksian buah-buahan yang ada untuk mempermudah proses penyortiran buah. Pada penelitian ini objek yang akan dideteksi adalah *image* dari buah apel, pir, anggur, blueberry, leci, dan rambutan. Alasan pemilihan dari buah ini adalah karena adanya kemiripan fisik pada buah yaitu buah apel dengan buah pir, buah anggur dengan blueberry dan juga buah rambutan dengan buah leci. Penelitian ini diharapkan akan mempermudah proses penyortiran buah dan mengurangi kesalahan dalam penyortiran akibat kemiripan buah dan dapat dikembangkan dengan penambahan robot agar sistem penyortiran menjadi otomatis. Pada penelitian ini menggunakan 1147 data yang digunakan sebagai data testing dan training dan mencapai akurasi 95,98% dalam pengujian program.

Kata kunci: *Image Processing, Faster R-CNN*, buah-buahan, android.

DETECTION OF FRUITS WITH SIMILARITY USING FASTER R-CNN BASED ON ANDROID

ABSTRACT

Object detection is one of the important things for making autonomous robot system. In this research, the author will build a system that can identify and detect fruit objects that have physical similarities that are applied on Android. This system can help to identify and detect fruits in fruit sorting. In this research the object to be detected is an image of apples, pears, grapes, blueberries, lychees, and rambutans. The reason for the selection of this fruit is because of the physical resemblance of the fruit, as apples with pears, grapes with blueberries and rambutans with lychees. This research is expected to simplify the fruit sorting process and reduce errors in sorting due to the similarity of the fruit and can be developed by adding robots so that the sorting system becomes automatic. This research was using 1147 images used as testing and training data and achieved 95,98% accuracy in testing using android.

Keywords: Image Processing, Faster R-CNN, fruits, android.

DAFTAR ISI

	Hlm.
PERSETUJUAN	ii
PERNYATAAN	iii
UCAPAN TERIMAKASIH	iv
ABSTRAK	vii
ABSTRACT	viii
DAFTAR ISI	ix
DAFTAR TABEL	xii
DAFTAR GAMBAR	xiii
UCAPAN TERIMA KASIH	2
BAB 1	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Batasan Masalah	3
1.4. Tujuan Penelitian	3
1.5. Manfaat Penelitian	3
1.6. Metodologi Penelitian	3
1.7. Sistematika Penulisan	4
BAB 2	6
2.1. Buah	6

2.2.	Pengolahan Citra Digital	7
2.3.	Citra	7
2.4.	Android	8
2.5.	Android Studio	8
2.6.	Tensorflow	9
2.7.	Machine Learning	9
2.8.	Artificial Neural Network (Jaringan Saraf Tiruan)	10
2.9.	Convolutional Neural Netowork (CNN)	11
2.10.	Region Convolutional Neural Netowork (R-CNN)	12
2.11.	Faster Region Convolutional Neural Netowork (Faster R-CNN)	13
2.12.	<i>Googlenet Inception-V2</i>	15
2.13.	Penelitian Terdahulu	16
BAB 3		19
3.1.	Data Yang Digunakan	19
3.2.	Analisis Sistem	19
3.2.3.	Arsitektur Umum	21
3.3.	Flowchart Sistem	23
3.3.1.	Use Case Diagram	24
3.3.2.	Diagram Aktivitas	25
3.4.	Perancangan Antarmuka	26
BAB 4		28
4.1.	Implementasi Sistem	28
4.1.1.	Spesifikasi perangkat keras yang digunakan	28
4.1.2.	Spesifikasi perangkat lunak yang digunakan	29
4.1.3.	Implementasi Data	29
4.1.4.	Pelabelan Objek	30
4.1.5.	Proses Training	31
4.1.6.	Implementasi Perancangan Antarmuka	36

4.2. Prosedur Penggunaan	37
4.3. Pengujian Sistem	38
BAB 5	40
5.1. Kesimpulan	40
5.2. Saran	40
DAFTAR PUSTAKA	42

DAFTAR TABEL

	Halaman.
Tabel 2.1. Penelitian Terdahulu	17
Tabel 3.1. Definisi Aktor	24
Tabel 3.2. Definisi Use Case	25
Tabel 3.3. Aktivitas Penggunaan	25
Tabel 3.4. Aktivitas Deteksi Objek	26
Tabel 3.5. Perancangan Antarmuka Aplikasi	27
Tabel 4.1. Spesifikasi Perangkat Keras	28
Tabel 4.2. Spesifikasi Perangkat Lunak	29
Tabel 4.3. Parameter Training	35
Tabel 4.4. Confusion Matrix	38

DAFTAR GAMBAR

	Halaman.
Gambar 2.1. Apel	6
Gambar 2.2. Pear	6
Gambar 2.3. Lyche	7
Gambar 2.4. Rambutan	7
Gambar 2.5. Anggur	7
Gambar 2.6. Blueberry	7
Gambar 2.7. Hirarki Tensorflow	9
Gambar 2.8. Struktur Sederhana Sebuah Neuron	10
Gambar 2.9. Model Tiruan Sebuah Neuron	11
Gambar 2.10. Arsitektur Umum dari CNN	12
Gambar 2.11. Arsitektur Umum dari R-CNN	13
Gambar 2.12. Arsitektur Umum dari Faster R-CNN	14
Gambar 2.13. Faktorisasi Konvolusi <i>Google Inception</i>	15
Gambar 2.14. Faktorisasi Konvolusi <i>Google Inception-v2</i>	16
Gambar 3.1. Arsitektur Umum	21
Gambar 3.2. Flowchart Sistem	23
Gambar 3.3. Use Case Diagram	24
Gambar 4.1. Contoh Hasil Pelabelan	30
Gambar 4.2. Kumpulan Image dan <i>File XML</i> dari <i>Image</i>	31
Gambar 4.3. Contoh isi File train_label.csv	31
Gambar 4.4. Contoh isi File test_label.csv	32
Gambar 4.5. Contoh isi File labelmap.pbtxt	33
Gambar 4.6. Iterasi <i>Training Data</i>	34
Gambar 4.7. <i>Loss Function</i>	34
Gambar 4.8. Pendeklarasian Kelas Kelas Objek Pada Android	36
Gambar 4.9. Logo Aplikasi	36
Gambar 4.10. Halaman Utama	37

Gambar 4.11. Tampilan Show Process

37

BAB 1

PENDAHULUAN

1.1. Latar Belakang

Pendeteksian objek merupakan salah satu hal yang penting dalam pembuatan sistem otomatis pada robot, untuk dapat melakukan pendeteksian objek tersebut pemanfaatan teknologi pengolahan citra adalah hal yang paling tepat untuk dapat mewujudkan hal ini. Pengolahan citra atau *digital image processing* adalah ilmu yang mempelajari tentang cara manipulasi dan modifikasi citra digital, untuk memperbaiki atau mengubah kualitas dari citra dan pengambilan informasi ciri citra yang bertujuan untuk menganalisis, dan mengekstrak informasi dari citra dengan menggunakan komputer untuk mendapatkan hasil yang lebih baik. Proses pengolahan citra digital dimaksudkan agar gambar awal yang memiliki *noise* dapat diolah lebih mudah dengan cara mengubah citra tersebut menjadi citra lain (Kadir & Susanto, 2013). Namun, sampai sekarang masih sedikit teknologi pengolahan citra yang diterapkan pada android yang dapat mengenali objek yang terdapat pada gambar khususnya objek buah-buahan, proses pengenalan objek ini sulit dilakukan karena beberapa objek buah memiliki ukuran yang kecil sehingga tidak terdeteksi. Teknologi ini dibutuhkan karena komputer/mesin hanya dapat mengenali gambar dalam bentuk pixel sebagai representasi dari gambar. Berdasarkan Maarten Christenhusz pada publikasi yang berjudul “*The number of known plant species in the world and its annual increase*” jumlah tumbuhan yang ada di dunia berkisar 374.000 tumbuhan dimana lebih dari 1000 tumbuhan tersebut merupakan tumbuhan yang menghasilkan buah. Karena begitu banyaknya jumlah buah yang ada maka kemungkinan terdapatnya buah-buahan yang memiliki kemiripan fisik baik berupa bentuk maupun warna yang dapat membuat keliru dalam mengenali buah-buahan yang ada. Oleh karena itu, pembuatan aplikasi ini dapat membantu untuk pengenalan buah-buahan yang ada. Pada penelitian ini objek yang akan dideteksi adalah *image* dari buah apel, pir, anggur, blueberry, leci, dan rambutan. Alasan pemilihan dari buah

ini adalah karena adanya kemiripan fisik pada buah yaitu buah apel dengan buah pir, buah anggur dengan blueberry dan juga buah rambutan dengan buah leci.

Penelitian ini akan diaplikasikan pada *android* agar dapat mengenali citra dari objek yang ada ditangkap melalui kamera. Pada penelitian ini citra yang akan dikenali adalah citra buah apel, pir, anggur, blueberry, leci, dan rambutan dengan menggunakan algoritma *Faster Region Convolutional Neural Network* (Faster R-CNN). *Faster Region Convolutional Neural Network* (R-CNN) adalah sebuah metode yang berbasis *deep learning object detection* yang biasa digunakan untuk mendeteksi objek. Penelitian terdahulu yang relevan dengan algoritma penulis gunakan adalah Xiaochun Mai et.al (2018) dalam penelitian yang berjudul “Faster R-CNN with *Classifier Fusion for Small Fruit Detection*”. Dalam penelitian ini dirancang suatu sistem untuk mengenali citra buah almond pada pohonnya menggunakan algoritma *Faster R-CNN*. Persamaan dengan penelitian yang penulis lakukan adalah sama-sama meneliti mengenai pengolahan citra buah dan menggunakan algoritma *Faster R-CNN*, sedangkan perbedaannya adalah penulis berfokus pada pengenalan citra buah apel, pir, anggur, blueberry, leci, dan rambutan sedangkan penelitian terdahulu berfokus pada satu buah saja. Berdasarkan latar belakang di atas, maka penulis mengajukan penelitian dengan judul “Aplikasi Pendeteksian Objek Buah-Buahan Yang Memiliki Kemiripan Menggunakan Alogirtma *Faster R-CNN* Berbasis Android”. Hasil dari penelitian ini diharapkan dapat mencapai akurasi yang tinggi dan bermanfaat untuk mengenali objek yang terdapat pada gambar khususnya objek buah-buahan dan dapat dimanfaatkan sebagai sistem penyortiran buah yang akan mempermudah proses penyortiran buah dan mengurangi kesalahan dalam penyortiran yang disebabkan kemiripan buah.

1.2. Rumusan Masalah

Pendeteksian objek merupakan salah hal yang penting dalam pembuatan sistem otomatis. Oleh karena itu dibutuhkan sebuah teknik yang dapat membuat komputer untuk dapat mendeteksi dan melakukan pelabelan, mendetesi letak serta mengidentifikasi objek yang ada pada gambar khususnya objek yang memiliki kemiripan fisik seperti buah apel dengan buah pir, buah anggur dengan blueberry dan juga buah rambutan dengan buah leci.

1.3. Batasan Masalah

Beberapa hal yang akan dijadikan batasan masalah dalam penelitian ini, yaitu:

1. Aplikasi ini hanya digunakan untuk pendeteksian objek buah apel, pir, anggur, blueberry, leci, dan rambutan.
2. Aplikasi dapat mendeteksi banyak buah yang berada dalam jangkauan kamera selama masih bagian dari buah yang telah ditentukan.
3. Aplikasi dapat memberi pelabelan objek pada buah yang telah dideteksi.
4. Aplikasi dibuat berbasis android.

1.4. Tujuan Penelitian

Penelitian ini bertujuan untuk mendeteksi dan melakukan pelabelan objek pada gambar, agar komputer dapat mengenali objek, khususnya objek yang memiliki kemiripan fisik seperti buah apel dengan buah pir, buah anggur dengan blueberry dan juga buah rambutan dengan buah leci yang diimplementasikan pada *smartphone*.

1.5. Manfaat Penelitian

Beberapa manfaat dari penelitian ini adalah

1. Sistem dapat mengenali dan memberikan pelabelan pada objek yang ada pada gambar dan dapat dikembangkan sebagai alat penyortiran buah otomatis.
2. Menambahkan kuantitas penelitian tentang penggunaan algoritma *Faster Region Convolutional Neural Network (Faster R-CNN)* dan penggunaan *smartphone*.
3. Membantu pengidentifikasian buah - buahan.

1.6. Metodologi Penelitian

Tahapan-tahapan yang akan dilakukan pada penelitian ini adalah sebagai berikut :

1. Studi Literatur

Pada tahap ini dilakukan peninjauan terhadap buku, artikel, jurnal, maupun hasil penelitian terdahulu sebagai referensi yang diperlukan dalam melakukan penelitian. Hal ini dilakukan untuk mendapatkan informasi mengenai metode

penggunaan algoritma *Faster Region Convolutional Neural Network (Faster R-CNN)*.

2. Analisa dan Perancangan

Tahap ini digunakan untuk mengolah data dari hasil studi literatur yang kemudian dilakukan analisis dan perancangan menggunakan algoritma *Faster Region Convolutional Neural Network (Faster R-CNN)*, sehingga menjadi sebuah aplikasi yang terstruktur dan jelas. Proses ini meliputi pembuatan algoritma program, *use case diagram*, *flowchart* sistem, rancangan aplikasi, dan pembuatan *User Interface* aplikasi.

3. Implementasi

Faster R-CNN (Faster Region Convolutional Neural Network) diimplementasikan dalam pembuatan suatu aplikasi dengan menggunakan bahasa pemrograman Python, PHP dan Java.

4. Pengujian

Pengujian dilakukan untuk melihat apakah aplikasi yang dibuat telah berhasil berjalan sesuai dengan tugasnya dan melakukan perbaikan program jika terjadi *error* pada aplikasi.

5. Dokumentasi dan Penyusunan Laporan

Pada tahap ini dilakukan dokumentasi dan penyusunan laporan setelah aplikasi diuji dan fungsinya sudah memenuhi harapan maka akan dibuat laporan hasil penelitian dalam bentuk skripsi.

1.7. Sistematika Penulisan

Sistematika penulisan dari skripsi ini terdiri dari lima bagian utama yaitu sebagai berikut :

Bab 1: Pendahuluan

Bab ini berisi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

Bab 2: Landasan Teori

Bab ini berisi teori-teori yang digunakan untuk memahami permasalahan yang dibahas pada penelitian ini serta teori pendukungnya yang berhubungan dengan *Image Processing* dan penerapannya.

Bab 3: Analisis dan Perancangan Sistem

Bab ini menjabarkan arsitektur umum, pengumpulan data yang dilakukan, dan implemtasi algoritma *Faster R-CNN* untuk mengidentifikasi objek buah-buahan secara real- time.

Bab 4: Implementasi dan Pengujian Sistem

Bab ini berisi pembahasan tentang implementasi dari analisis dan perancangan yang disusun pada Bab 3 dan pengujian yang dilakukan untuk mengetahui apakah sistem yang dibangun sesuai dengan yang diharapkan.

Bab 5: Kesimpulan dan Saran

Bab ini berisi kesimpulan dari keseluruhan uraian bab-bab sebelumnya dan saran-saran yang diajukan untuk pengembangan penelitian yang dilakukan selanjutnya.

BAB 2

LANDASAN TEORI

Bab ini membahas tentang teori penunjang dan penelitian sebelumnya yang berhubungan dengan penerapan metode *Faster R-CNN* dalam melakukan pendeteksian terhadap berbagai macam objek.

2.1. Buah

Buah adalah organ pada tumbuhan yang memiliki bunga yang merupakan perkembangan lanjutan dari bakal buah (ovarium). Buah terbentuk dari hasil penyerbukan putik dan benang sari, buah biasanya membungkus dan melindungi biji, hal ini tidak terlepas kaitannya dari fungsi utama buah yaitu sebagai pemencar biji tumbuhan dan alat perkembangbiakan secara generatif. Terdapat 374.000 jumlah tumbuhan yang ada didunia dimana lebih dari 1000 tumbuhan tersebut merupakan tumbuhan yang menghasilkan buah (Christenhusz, 2016). Oleh karena begitu banyaknya jumlah buah yang ada maka ada kemungkinan bahwa terdapat buah - buahan yang memiliki kemiripan fisik baik berupa bentuk maupun warna yang dapat membuat kesalahan dalam mengenali buah - buahan yang ada. Pada penelitian ini ada 6 buah yang akan dideteksi oleh sistem yaitu buah apel, pir, anggur, blueberry, leci, dan rambutan. Alasan dari pemilihan buah tersebut adalah adanya kemiripan fisik berupa bentuk dan warna buah-buahan.



Gambar 2.1 Apel (google.com)



Gambar 2.2 Pear (google.com)



Gambar 2.3 Lyche (google.com)



Gambar 2.4 Rambutan (google.com)



Gambar 2.5 Anggur (google.com)



Gambar 2.6 Blueberry (google.com)

2.2. Pengolahan Citra Digital

Pengolahan gambar citra atau *digital image processing* adalah ilmu yang mempelajari tentang cara manipulasi dan modifikasi citra digital, untuk memperbaiki atau mengubah kualitas dari citra dan pengambilan informasi ciri citra yang bertujuan untuk menganalisis, dan mengekstrak informasi dari citra dengan menggunakan komputer untuk mendapatkan hasil yang lebih baik. Proses pengolahan citra digital dimaksudkan agar gambar awal yang memiliki *noise* dapat diolah lebih mudah dengan cara mengubah citra tersebut menjadi citra lain (Kadir & Susanto, 2013). Teknik pengolahan yang digunakan pada penelitian ini adalah :

2.2.1 Resizing

Resizing adalah proses pengolahan citra digital untuk mengubah ukuran citra dengan memperbesar atau memperkecil ukuran citra secara *horizontal* dan/atau *vertical* dengan cara merubah nilai pada masing - masing *pixel*.

2.3. Citra

Citra adalah tampilan objek dua dimensi yang merupakan representasi dari sebuah objek yang ditangkap oleh kamera. Citra digital terbentuk dari sejumlah piksel yang dimana

setiap piksel tersebut memiliki posisi dan nilai tertentu yang disimpan pada media elektronik dalam bentuk angka (Gonzales & Woods, 2008).

2.4. Android

Android adalah suatu sistem operasi yang bersifat *open source* yang digunakan pada perangkat *mobile* berbasis *linux* yang terdiri dari *middleware*, dan aplikasi. Android menyediakan *platform* yang terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri. Penggunaan android SDK (*Software Developement Kit*) membuat pengembang android dapat membuat aplikasi android mereka sendiri pada *platform* android menggunakan bahasa pemrograman Java (Eko, 2012).

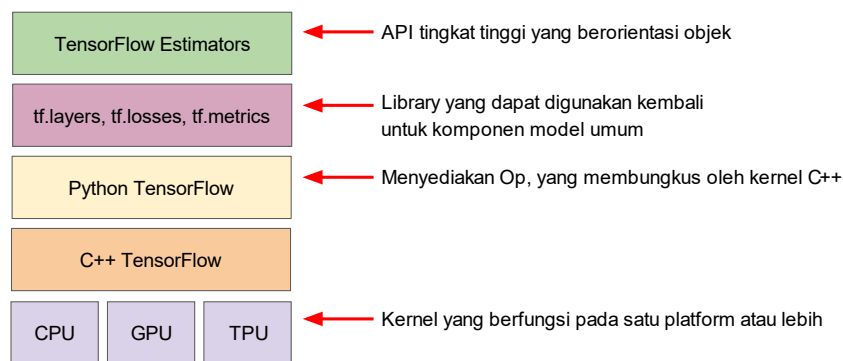
2.5. Android Studio

Android Studio adalah sebuah *environment* yang dibuat untuk membantu *developer* dalam mengembangkan aplikasi yang berbasis android. Android Studio menawarkan banyak fitur untuk meningkatkan produktivitas pengguna saat membuat aplikasi Android, seperti (Android, 2019) :

- Sistem versi berbasis *Gradle* yang fleksibel
- *Emulator* yang cepat dan kaya fitur
- Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android
- *Instant Run* untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
- Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh
- Alat pengujian dan kerangka kerja yang ekstensif
- Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain
- Dukungan C++ dan NDK
- Dukungan bawaan untuk *Google Cloud Platform*, mempermudah pengintegrasian *Google Cloud Messaging* dan *App Engine*.

2.6. Tensorflow

TensorFlow adalah *framework* untuk pembelajaran *machine learning* yang di buat oleh Google yang mendukung beragam bahasa pemrograman yang dibuat untuk mempermudah *developer* melakukan pembelajaran mesin (Devikar., 2016). *Tensorflow* memungkinkan user untuk membuat model *machine learning* sendiri sesuai dengan kebutuhan yang dibutuhkan *developer*. Gambar 2.7 menunjukkan tingkatan *toolkit TensorFlow*.



Gambar 2.7 Hirarki Tensorflow (Tensorflow, 2019)

2.7. Machine Learning

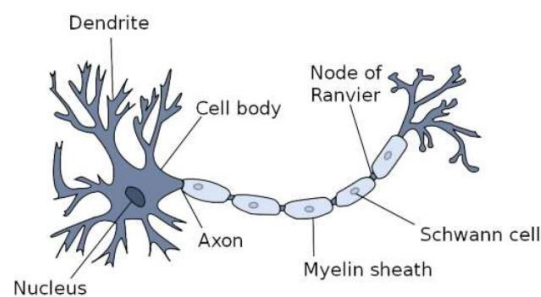
Machine learning merupakan tahap bagaimana sebuah mesin atau komputer dapat belajar dari data yang diberikan kepadanya yang kemudian dapat digunakan untuk menganalisa atau memecahkan sebuah masalah. *Machine learning* membutuhkan data untuk belajar sehingga dapat diartikan bahwa *machine learning* tergantung pada data yang diberikan yang diistilahkan sebagai *learn from data* (Lukman., 2014).

Seperti pada kasus *bait shyness*, dimana tikus belajar untuk menghindari makanan yang beracun. Dimana ketika tikus menemukan sebuah makanan dengan aroma dan atau bentuk yang tidak mereka kenali, mereka akan memakan makanan tersebut dalam jumlah yang sangat sedikit dan selanjutnya akan merasakan efek dari makanan tersebut terhadap tubuh tikus, setelah mereka memakan makanan tersebut dan makanan tersebut menyebabkan kerusakan pada tubuhnya, maka mereka tidak akan memakan kembali makanan tersebut karena mereka beranggapan makanan tersebut akan memberikan pengaruh buruk pada tubuh mereka (Ben-david, et al., 2014). Sama halnya dengan dasar dari proses *training machine learning* dimana komputer belajar

untuk mengambil keputusan dan menyelesaikan sebuah masalah atau mengenali masalah berdasarkan data yang telah dilatihkan kepadanya.

2.8. Artificial Neural Network (Jaringan Saraf Tiruan)

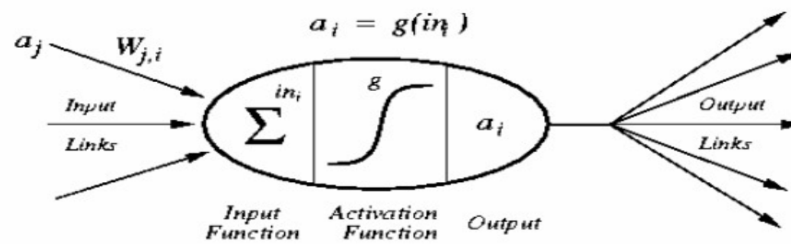
Artificial neural network atau jaringan saraf tiruan (JST) adalah salah satu cabang ilmu dari *artificial intelligence* yang terinspirasi berdasarkan sistem saraf manusia. JST adalah sebuah sistem pemrosesan informasi yang dibangun berdasarkan sistem saraf pada manusia yang terdiri dari *neuron - neuron* (Fausett., 1994). Kumpulan *neuron* dibuat menjadi sebuah jaringan yang saling terhubung dan memiliki fungsi sebagai alat pembelajaran pada mesin. Struktur jaringan saraf tiruan terinspirasi dari jaringan otak manusia yang tersusun lebih dari 10^{13} buah *neuron* yang masing-masing terhubung dengan sekitar 10^{15} buah *dendrit*. *Neuron* adalah satuan unit pemroses terkecil pada otak yang disederhanakan dalam gambar ilustrasi berikut :



Gambar 2.8. Struktur Sederhana Sebuah *Neuron* (Ben-david, et al., 2014)

Gambar 2.8 menunjukkan satu dari 10^{13} *neuron* yang dimodelkan dalam jaringan saraf tiruan. *Dendrit* berfungsi sebagai penyampai *impulse*/sinyal ke badan sel dan dikirimkan ke jaringan lainnya melalui *axon*. Suatu *neuron* tersusun atas 3 komponen utama yaitu:

1. *Dendrites*, merupakan saluran sinyal input yang kekuatan koneksinya dalam pengiriman informasi ke inti sel dipengaruhi oleh sebuah bobot (*weight*).
2. Badan Sel (*Cell Body*), merupakan tempat proses komputasi sinyal *input* dari dendrit yang menghasilkan sinyal output yang akan dikirimkan kepada *neuron* lainnya.
3. *Axon*, merupakan bagian yang bertugas mengirimkan sinyal output kepada *neuron* lain yang terhubung pada *neuron* tersebut.



Gambar 2.9. Model Tiruan dari Sebuah Neuron (Wahyuni, et al., 2015)

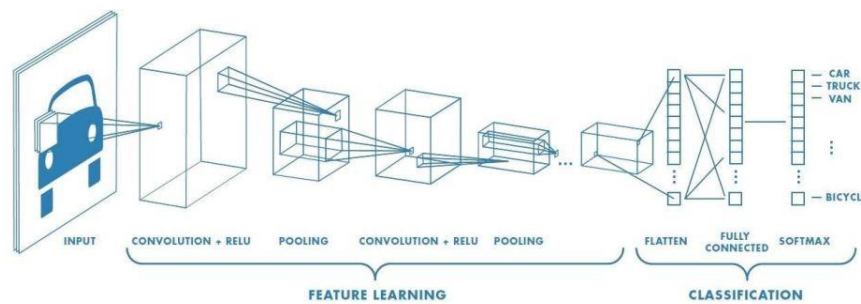
Gambar 2.9 merupakan model tiruan sebuah *neuron* dan proses perkaliannya, dimana sejumlah sinyal masukan a dikalikan dengan masing-masing penimbang yang bersesuaian dan menghasilkan derajat sinyal *output* sinyal $F(a,w)$.

- a_j :Nilai aktivasi dari unit j
- $w_{j,i}$:Bobot dari unit j ke unit i
- in_i :penjumlahan bobot dan masukan ke unit i
- g :fungsi aktivasi
- a_i :nilai aktivasi dari unit i

2.9. Convolutional Neural Netowork (CNN)

Convolutional Neural Network adalah sebuah *neural network* yang digunakan untuk menganalisis gambar, mendeteksi objek dan mengenali objek pada *image*, yang melibatkan berbagai parameter untuk mencirikan jaringan mulai dari pencirian objek sampai dengan pengklasifikasin objek pada saraf tiruan. *Convolutinal neural network* dibuat untuk mengurangi jumlah parameter dan meyederhanakan arsitektur jaringan dalam melakukan pengenalan objek (Karapathy,2018).

CNN merupakan *neural network* yang tidak teralu berbeda dengan *neural network* lainnya. CNN terdiri dari *neuron* yang memiliki *weight*, *bias* dan *activation function*, namun arsitektur dari *Convolutional Neural Network* dibagi menjadi dua bagian besar, yaitu *Feature Extraction Layer* dan *Fully-Connected Layer (Multilayer Perceptron)*.



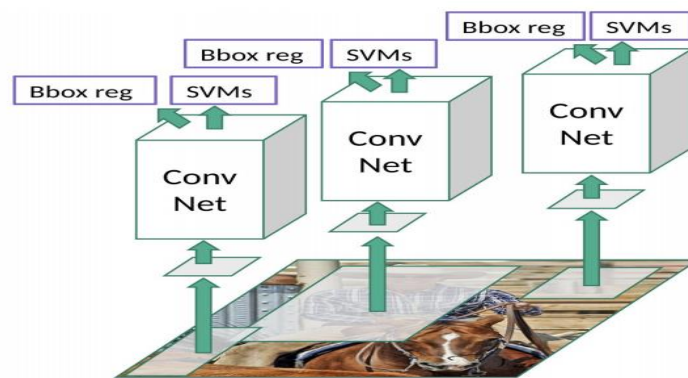
Gambar 2.10. Arsitektur Umum dari CNN (Karapathy, 2018)

2.10. Region Convolutional Neural Netowork (R-CNN)

Region Convolutional Neural Network (R-CNN) adalah sebuah metode yang berbasis *deep learning object detection* yang biasa digunakan untuk pendeteksian objek. R-CNN menggunakan algoritma *selective search* untuk membuat proposal dari *image*, dimana gambar yang di *input* akan dikelompokkan menjadi 2000 *region* yang dipilih berdasarkan tekstur, intentsitas, dan warna. Hal tersebut dilakukan untuk menutupi kelemahan CNN yang membagi *region* gambar dengan skala *region* besar yang membuat proses pengidentifikasian menjadi lebih lambat.

Kelemahan pada R-CNN :

1. Proses training data yang relatif lambat karena menggunakan 2000 *region proposal* untuk setiap *image*.
2. Tidak dapat diimplementasikan untuk klasifikasi *real time* karena membutuhkan waktu sekitar 47 – 50 detik untuk proses per-*image*.
3. Hanya bisa menggunakan algoritma *selective search* pada proses pengenalanya, tidak dapat menggunakan algoritma lain selain *selective search*.



Gambar 2.11. Arsitektur Umum dari R-CNN (Gandhi., 2018)

2.11. Faster Region Convolutional Neural Network (Faster R-CNN)

Faster R-CNN adalah sebuah metode yang berbasis *deep learning object detection* yang biasa digunakan untuk pendeteksian objek yang dikembangkan dari algoritma *R-CNN* untuk menutupi kelemahan yang ada pada *R-CNN*. Kelebihan pada *Faster R-CNN* adalah *Faster R-CNN* menggunakan *RPN*, dimana *RPN* adalah sebuah *neural network* yang menggantikan peran *selective search* untuk mengajukan *region*, peran *selective search* digantikan karena prosesnya yang lambat dalam mengolah image yaitu sekitar 2 sekon per gambar (Ren, et al., 2017). *RPN* berfungsi untuk menghasilkan beberapa *bounding box* dimana setiap *box* memiliki 2 skor probabilitas apakah pada lokasi tersebut terdapat objek atau tidak, dengan adanya *RPN* pemrosesan tidak dilakukan berulang ulang seperti yang dilakukan pada *R-CNN* dan membuat keseluruhan model dapat di-*train* secara *end-to-end*.

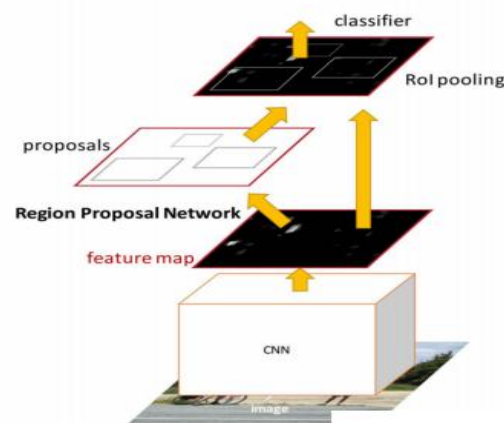
Cara Kerja *Faster R-CNN* :

1. *Convolution layer* membuat dan mengirimkan *feature map* kepada *RPN*.
2. *RPN* memproses *feature map* yang ada dan membuat *region proposal* serta membuat *bounding box* sebagian yang dianggap memiliki kemungkinan terdapat objek.
3. *R-CNN* melakukan klasifikasi terhadap proposal yang telah dibuat *RPN* dan menentukan apakah objek pada proposal merupakan objek yang ada pada model yang telah dilatih dan memberikan pelabelan pada objek.

Kelebihan Faster R-CNN :

1. Proses training data yang lebih cepat karena *proposal region* yang dibuat tidak sebesar pada *R-CNN* yang membuat proses pengerjaan lebih ringan.
2. Dapat diimplementasikan untuk klasifikasi *real time* karena membutuhkan waktu pemrosesan yang relatif singkat yaitu sekitar 1 – 2 detik.

Faster R-CNN:



Gambar 2.12. Arsitektur Umum dari Faster R-CNN (Gavrilescu., 2018)

Penjelasan arsitektur umum dari *Faster R-CNN*:

- a) *Convolutional Layer* : Pada tahap ini convolutional layer mempelajari bagian bagian penting yang dapat menjadi ciri khas dari objek tersebut serta membuat *feature map* dari objek yang telah ditangkap kamera *smartphone*. *Convolutional* adalah *layer* yang terdiri dari *neuron* yang tersusun sedemikian rupa sehingga membentuk sebuah *filter* dua dimensi dengan panjang dan tinggi (*pixels*).
- b) *Feature Map* : *Feature Map* adalah sebuah map yang dibuat oleh *convolutional layer* yang berisi informasi tentang representasi *vector* dari *image* yang ditangkap.
- c) *RPN (Region Proposal Network)* : *RPN* merupakan sebuah modul yang bekerja untuk mengolah *feature map* yang telah dibuat pada *convolution layer* untuk memprediksi bagian yang dianggap sebagai objek dan melakukan prediksi *bounding box* dari objek tersebut, *RPN* dibagi menjadi 2 *convolution layer* dimana 1 *layer* bertanggung jawab untuk mendeteksi letak objek dan 1 *layer* berfungsi memprediksi *bounding box*.
- d) *ROI Pooling* : *ROI* merupakan *layer* yang bertanggung jawab untuk

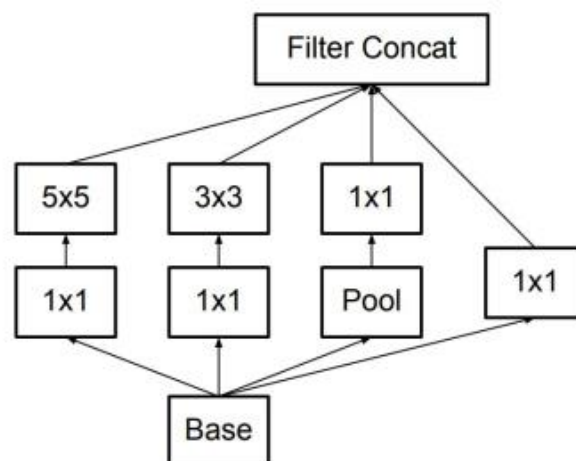
mengeksrak informasi feature map yang telah diproses oleh *RPN* untuk dikasifikasi pada *classification layer*.

- e) *Classification Layer* : *Classification layer* merupakan *layer* yang berfungsi untuk mengelompokkan objek yang telah dideteksi pada *RPN* dan melakukan pelabelan terhadap objek tersebut serta memberikan *bounding box* pada objek tersebut.

2.12. Googlenet Inception-V2

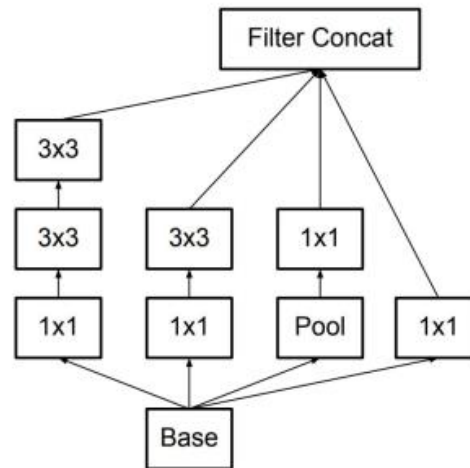
Google Inception merupakan sebuah pengembangan dari *convolutional neural network* yang dikembangkan oleh *Google* yang digunakan untuk proses pengekstraksian ciri *image*. Pengembangan ini dibuat karena *CNN* memiliki keterbatasan seperti rentan terhadap *overfitting*, penggunaan memori yang besar dan juga kompleksitas dari pengerjaan. *googlenet inception* kemudian dikembangkan kembali mejadi *inception-v2* dikarenakan *googlenet inception* dianggap masih memiliki banyak kekurangan seperti perubahan dimensi *input* secara drastis dan juga masih terdapatnya kompleksitas dari komputasi. Kelebihan dari *googlenet inception-v2*

1. Mengurangi representasi *bottleneck*, dengan cara tidak merubah ukuran *input* secara drastis karena akan mengurangi kinerja dari *neural network*.
2. Menggunakan faktorisasi yang lebih efisien sehingga mengurangi kompleksitas dari komputasi dan juga mempercepat proses pengekstraksian ciri citra.



Gambar 2.13. Faktorisasi Konvolusi *Google Inception* (White., 2018)

Gambar 2.13. merupakan faktorisasi yang digunakan pada *Googlenet inception* dimana *input* akan dimasukkan ke *filter* 1x1, 3x3 dan 5x5 dan *max pooling* sebelum dimasukkan kedalam *inception* lainnya.



Gambar 2.14. Faktorisasi Konvolusi *Google Inception-v2* (Bharth., 2018)

Gambar 2.14. merupakan faktorisasi yang digunakan pada *Googlenet inception-v2* dimana *input* akan dimasukkan ke *filter* 1x1, 3x3 sebelum dimasukkan kedalam *inception* lainnya. Perbedaan *googlenet inception* dengan *inception-v2* adalah penggunaan *filter* 5x5 yang diganti dengan dengan dua *layer filter* 3x3 karena penggunaan *filter* 5x5 dianggap memakan banyak *resource* dan bila *layer* ini digantikan dua *layer* 3x3 akan meningkatkan daya komputasi sebesar 2,78 kali.

2.13. Penelitian Terdahulu

Penelitian yang dilakukan oleh Bin Liu et al. pada tahun 2017, membuat *Study of Object Detection Based on Faster R-CNN*, tujuan yang ingin dicapai pada penelitian ini adalah mendeteksi beberapa objek seperti kucing, manusia, mobil dan kuda dan objek-objek lainnya, dimana data gambar yang digunakan penulis pada penelitian ini berasal dari *ImageNet*.

Penelitian yang dilakukan oleh Shih-Chung Hsu et al. pada tahun 2018, membuat *Vehicle Detection using Simplified Fast R-CNN*, tujuan yang ingin dicapai pada penelitian ini adalah mendeteksi kendaraan yang tertangkap kamera pada *dashboard*, pada penelitian ini persentasi keberhasilan pedeteksi kendaraan adalah 90.3%.

Penelitian yang dilakukan oleh Beibei Zhu et al. pada tahun 2016, membuat *Automatic Detection of Books Based on Faster R-CNN*, tujuan yang ingin dicapai pada penelitian ini adalah mendeteksi buku berdasarkan bentuk dari objeknya, persentase keberhasilan deteksi penelitian ini mencapai 98.6 %.

Penelitian yang dilakukan oleh Xiaochun Mai et al. pada tahun 2018, *Faster R-CNN Classifier Fusion for Small Fruit Detection*, tujuan yang ingin dicapai pada penelitian ini adalah mendeteksi buah kecil yang ada pada pohonnya pada penelitian ini buah yang akan dideteksi adalah buah almond yang masih berada pada pohonnya.

Penelitian yang dilakukan oleh Wei Zhang et al. pada tahun 2018, *Deconv R-CNN For Small Object Detection On Remote Sensing Images*, tujuan yang ingin dicapai pada penelitian ini adalah pesawat yang terlihat kecil karena diambil dari ketinggian. Pada penelitian ini penulis berhasil mendeteksi objek pesawat yang terlihat sangat kecil karena diambil dari ketinggian. persentase keberhasilan deteksi penelitian ini mencapai 80.5 %.

Tabel 2.1. Penelitian Terdahulu

NO	Peneliti	Judul	Keterangan
1	Bin Liu et al.	<i>Study of Object Detection Based on Faster R-CNN</i>	Pada penelitian ini penulis menggunakan algoritma <i>faster R-CNN</i> untuk mendeteksi beberapa objek seperti kucing, manusia, mobil dan kuda dan objek-objek lainnya. Data gambar yang digunakan penulis berasal dari ImageNet.
2	Shih-Chung Hsu et al.	<i>Vehicle Detection using Simplified Fast R-CNN</i>	Pada penelitian ini penulis menggunakan algoritma <i>fast R-CNN</i> untuk mendeteksi kendaraan dimana <i>device</i> ini dipasang pada <i>dashboard</i>

			<p>kamera pada mobil, pada penelitian ini persentasi keberhasilan pedeteksian kendaraan adalah 90.3%.</p>
3	Beibei Zhu et al.	<i>Automatic Detection of Books Based on Faster RCNN</i>	<p>Pada penelitian ini penulis menggunakan algoritma <i>faster R-CNN</i> untuk mendeteksi buku berdasarkan bentuk dari objeknya, persentase keberhasilan deteksi penilitian ini mencapai 98.6 %.</p>
4	Xiaochun Mai et al.	<i>Faster R-CNN with Classifier Fusion for Small Fruit Detection</i>	<p>Pada penelitian ini penulis menggunakan algoritma <i>faster R-CNN</i> menggunakan 5 layar konvolusi untuk mendeteksi buah almond yang masih berada pada pohonnya pada penelitian ini penulis.</p>
5	Wei Zhang et al.	<i>Deconv R-CNN For Small Object Detection On Remote Sensing Images</i>	<p>Pada penelitian ini penulis menggunakan algoritma <i>R-CNN</i> dan berhasil mendeteksi objek pesawat yang terlihat sangat kecil karena diambil dari ketinggian persentase keberhasilan deteksi penilitian ini mencapai 80.5 %.</p>

BAB 3

ANALISIS DAN PERANCANGAN

Bab ini akan membahas tentang analisis dan perancangan dalam aplikasi pendeteksian objek buah - buahan menggunakan *Faster R-CNN*. Pada tahapan selanjutnya yaitu dilakukan perancangan tampilan antarmuka sistem yang akan dibangun.

3.1. Data Yang Digunakan

Data yang digunakan pada penelitian ini yaitu data *image* dengan ekstensi jpg yang diperoleh dari hasil foto penulis dan *image* yang *download* dari <https://pixabay.com> dan juga <https://www.kaggle.com/moltean/fruits>. Data *image* yang diperoleh dibagi menjadi dua *dataset* yaitu data *training* dan data *validasi* dimana data *training* berjumlah 1018 *image* dan data *validasi* sebanyak 129 *image*.

3.2. Analisis Sistem

Metode yang diajukan penulis untuk pendeteksian objek ini memiliki tiga tahapan yaitu input, proses dan output:

3.2.1 Input

Input yang digunakan adalah citra RGB dari buah - buahan yang diambil dari kamera penulis juga *image* yang diambil dari <https://pixabay.com> dan juga <https://www.kaggle.com/moltean/fruits> dimana dari semua kategori objek, 90% data dibuat sebagai data latih dan 10% data lainnya dibuat sebagai data *validasi*.

3.2.2 Proses

Pada tahap ini merupakan proses pengolahan citra buah - buahan yang telah diperoleh. Proses yang dilakukan terdiri dari *resize*, pelabelan objek, *generate data*, *training*, *export inference*, *testing*.

3.2.2.1 Resize

Pada tahap awal proses *preprocessing* yaitu dilakukan *resizing* untuk mengubah ukuran citra dengan memperkecil ukuran citra secara *horizontal* dan/atau *vertical*, gambar akan di-*resize* agar ukuran *file* tidak lebih 200 KB dan dimensi *image* tidak lebih dari 720 x 1280. Hal ini dilakukan untuk membuat proses *training* lebih ringan dan cepat.

3.2.2.2 Pelabelan Objek

Pada tahap ini dilakukan pelabelan dan pemberian *bounding box* terhadap objek pada setiap *image* yang telah diambil hal ini bertujuan untuk memberikan ciri dari masing-masing objek dan melatih sistem untuk mengenali dan memberikan *bounding box* kepada setiap objek. Hasil dari pelabelan objek ini adalah *file xml* yang berisi informasi tentang kelas dan *bounding box* objek dimana setiap *image* akan memiliki satu *file xml*.

3.2.2.3 Generate Data

Pada tahap ini dilakukan proses *record* data yang akan dijadikan data *training* dan data *testing* dimana semua *image* yang telah memiliki *file xml* akan dibuat menjadi *file list* dengan ekstensi *csv*, dan setelah *file csv* jadi maka *file* tersebut akan di-generate menjadi *file* “*train.record*” dan “*test.record*” menggunakan *library tensorflow*. Data “*train.record*” dan “*test.record*” ini telah berisi semua informasi mengenai *image* dan letak *bounding box* objek pada masing masing *image* dan kedua *file* inilah yang akan dibuat menjadi data yang akan dilatihkan ke sistem.

3.2.2.4 Training

Tahap *training* adalah tahap melatihkan data yang telah digenerate untuk dipelajari oleh sistem. Algoritma yang digunakan adalah *Faster R-CNN Googlenet Inception V2*. Pada penelitian ini penulis menggunakan 1147 *images*, dimana 1018 *images* sebagai *training images* dan 129 *images* sebagai data validasi.

3.2.2.5 Export Inference

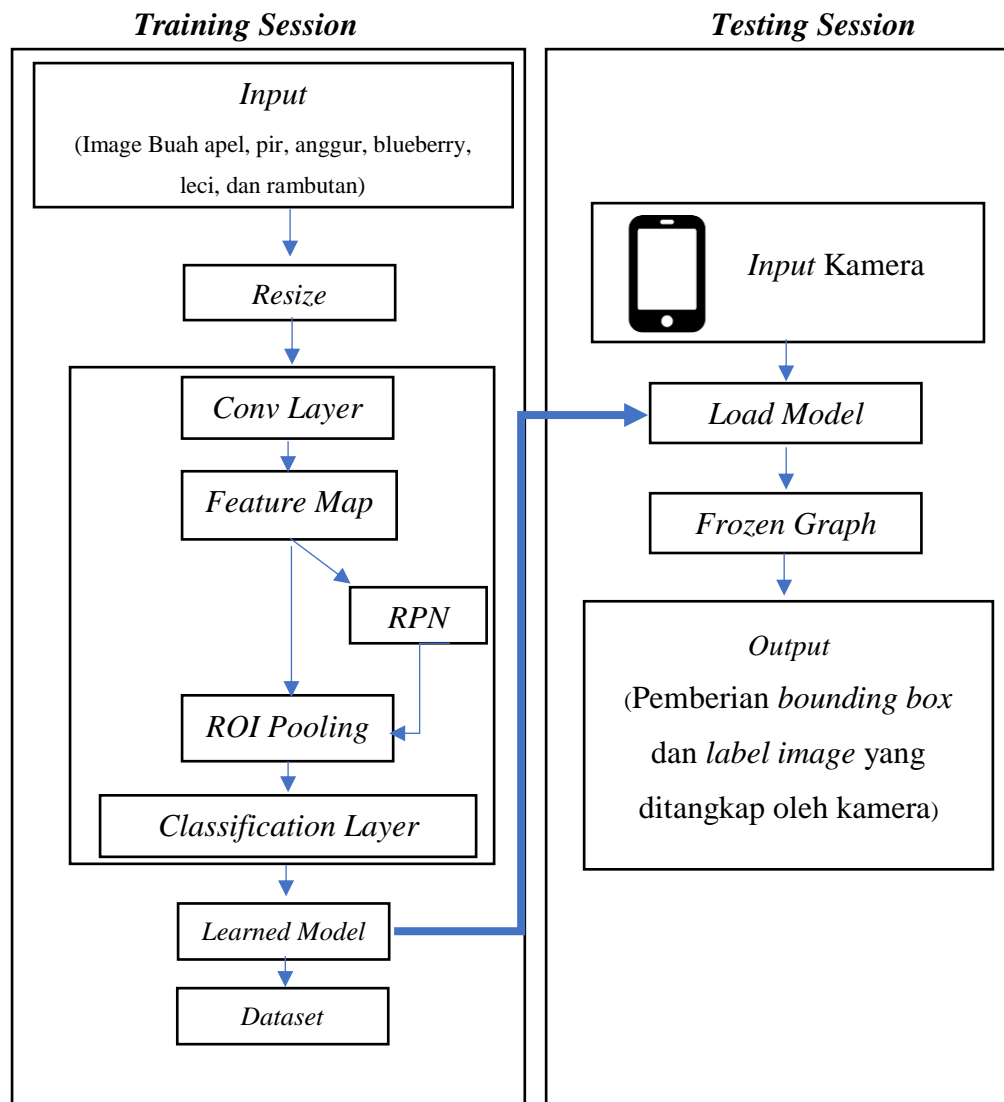
Tahap ini merupakan proses *mengcovert* data hasil *training* menjadi *frozen model* yang dapat dijalankan pada android dimana *frozen model* ini sudah di optimalkan sehingga ukurannya lebih ringan dari *file* hasil *training*.

3.2.2.6 Testing

Pada tahap ini *user* menjalankan program dari android untuk melakukan pendeteksian objek untuk melihat tingkat akurasi sistem.

3.2.3. Arsitektur Umum

Adapun arsitektur umum dari aplikasi ini dapat dilihat pada Gambar 3.1.



Gambar 3.1. Arsitektur Umum

Keterangan Arsitektur Umum (Gambar 3.1) adalah :

1. Training Session

- Input :** Pada tahap ini, sistem diberikan *input data* berupa citra dari buah apel, pir, anggur, blueberry, leci, dan rambutan.
- Resize :** Pada tahap awal proses *preprocessing* yaitu dilakukan *resizing* untuk mengubah ukuran citra dengan memperkecil ukuran citra secara *horizontal* dan/atau *vertical*, gambar akan di-*resize* agar ukuran *file* tidak lebih 200 KB dan dimensi *image* tidak lebih dari 720 x 1280.

- c) *Convolutional Layer* : Pada tahap ini *convolutional layer* akan mengekstraksi ciri gambar dan mempelajari bagian bagian penting yang dapat menjadi ciri khas dari objek tersebut serta membuat *feature map* dari objek. *Convolutional* adalah *layer* yang terdiri dari *neuron* yang tersusun sedemikian rupa sehingga membentuk sebuah *filter* dua dimensi dengan panjang dan tinggi (*pixels*).
- d) *Feature Map* : *Feature Map* adalah sebuah map yang dibuat oleh *convolutional layer* yang berisi informasi tentang representasi *vector* dari *image* yang ditangkap, *convolution layer* akan menghasilkan dua *feature map* yang sama, *feature map* pertama akan diolah di *RPN* untuk menghasilkan *region proposal* dan *feature map* lainnya akan langsung dikirim ke *pooling layer*.
- e) *RPN (Region Proposal Network)* : *RPN* merupakan sebuah modul yang bekerja untuk mengolah *feature map* yang telah dibuat pada *convolution layer* untuk memprediksi bagian yang dianggap sebagai objek dan melakukan prediksi *bounding box* dari objek tersebut, *RPN* dibagi menjadi 2 *convolution layer* dimana 1 *layer* bertanggung jawab untuk mendeteksi letak objek dan 1 *layer* berfungsi memprediksi *bounding box*, keluaran dari *RPN* adalah *region proposal* dari *image*.
- f) *ROI Pooling* : *ROI* merupakan *layer* yang bertanggung jawab untuk menyamakan ukuran dari *feature map* dan *region proposal* yang telah diolah oleh *RPN* serta mengirim informasi *feature map* dan *proposal* untuk diklasifikasi pada *classification layer*.
- g) *Classification Layer* : *Classification layer* merupakan *layer* yang berfungsi untuk mengelompokkan objek yang telah dideteksi pada *RPN* dan melakukan pelabelan terhadap objek tersebut serta memberikan *bounding box* pada objek tersebut.
- h) *Learned Model* : *Learned model* adalah model yang telah selesai dipelajari oleh sistem.
- i) *Dataset* : *Dataset* adalah kumpulan dari *learned model* lain yang tidak dipakai dalam pengujian yang berisi informasi bobot dataset.

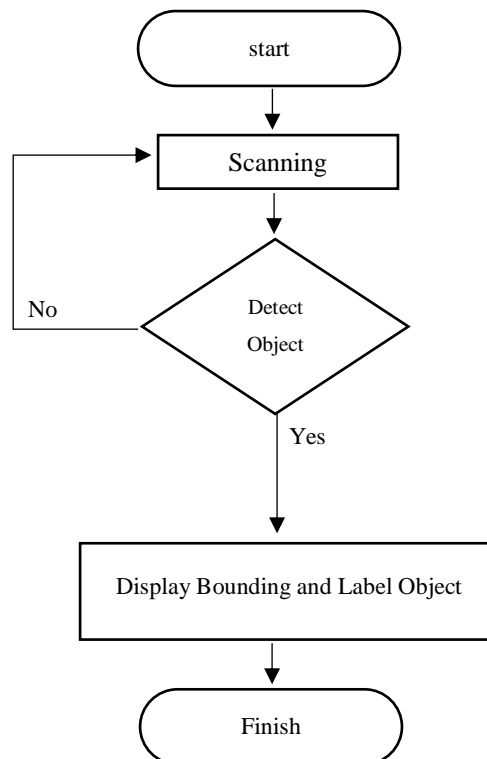
2. Testing Session

- a) *Input* : Pada tahap ini, sistem diberikan *input data* berupa citra dari buah apel, pir, anggur, blueberry, leci, dan rambutan.

- b) *Load Model* : *Load model* adalah tahap dimana sistem akan memuat kembali model yang telah disimpan pada masa *training session*.
- c) *Frozen Graph* : Pada tahap ini data *inputan* yang diterima melalui kamera akan diproses pada *graph* yang tersimpan pada *frozen model* untuk dilakukan identifikasi dan pemberian *bounding box* berdasarkan *weight* yang telah tersimpan pada model yang telah dilatih.
- d) *Output* : *Output* merupakan hasil dari identifikasi yang telah dilakukan, dimana objek yang telah diklasifikasi diberikan *bounding box* dan label.

3.3. Flowchart Sistem

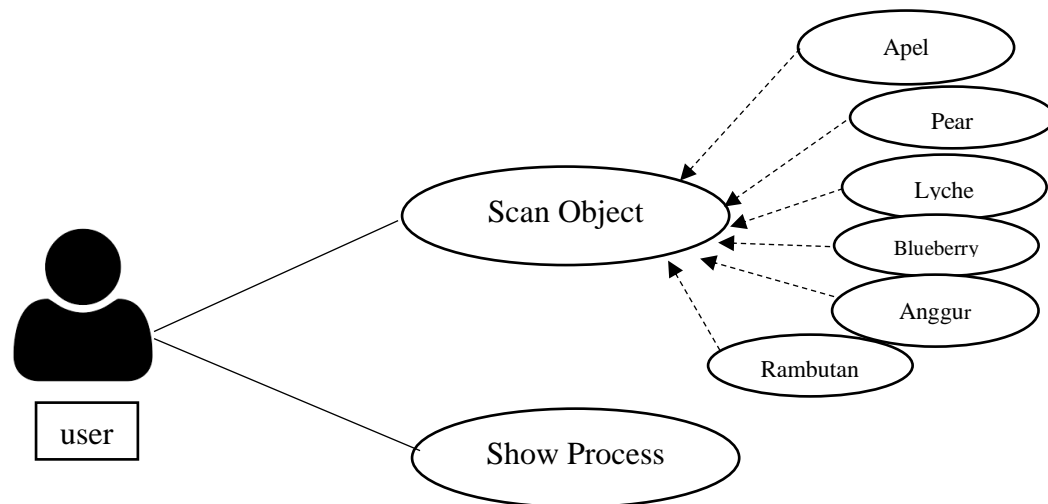
Pada sub-bab ini dijelaskan alur kerja sistem, serta aspek-aspek teknik yang merupakan solusi perancangan sistem. Adapun cara kerja sistem secara menyeluruh dapat dilihat pada Gambar 3.2.



Gambar 3.2. Flowchart Sistem

3.3.1. Use Case Diagram

Use case diagram bertujuan untuk mendeskripsikan sebuah interaksi antara aktor dengan sistem yang dibuat, dan digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem maupun siapa yang berhak menggunakan fungsi tersebut. Adapun *use case* dari aplikasi dapat dilihat pada Gambar 3.3



Gambar 3.3. Use Case Diagram

Keterangan dari Gambar 3.4 adalah :

1. Aktor

Fungsi aktor yaitu untuk mendeskripsikan peranan aktor pada sistem. Defenisi tersebut dapat dilihat pada Tabel 3.1.

Tabel 3.1. Defenisi Aktor

Aktor	Deskripsi
<i>User</i>	Adalah aktor yang berperan sebagai pengguna aplikasi yang akan melihat hasil deteksi objek beserta label objek .

2. Definisi *Use Case*

Fungsi *use case* yaitu untuk menjelaskan proses yang terjadi pada diagram *use case*. Defenisi tersebut dapat dilihat pada Tabel 3.2.

Tabel 3.2. Defenisi *Use Case*

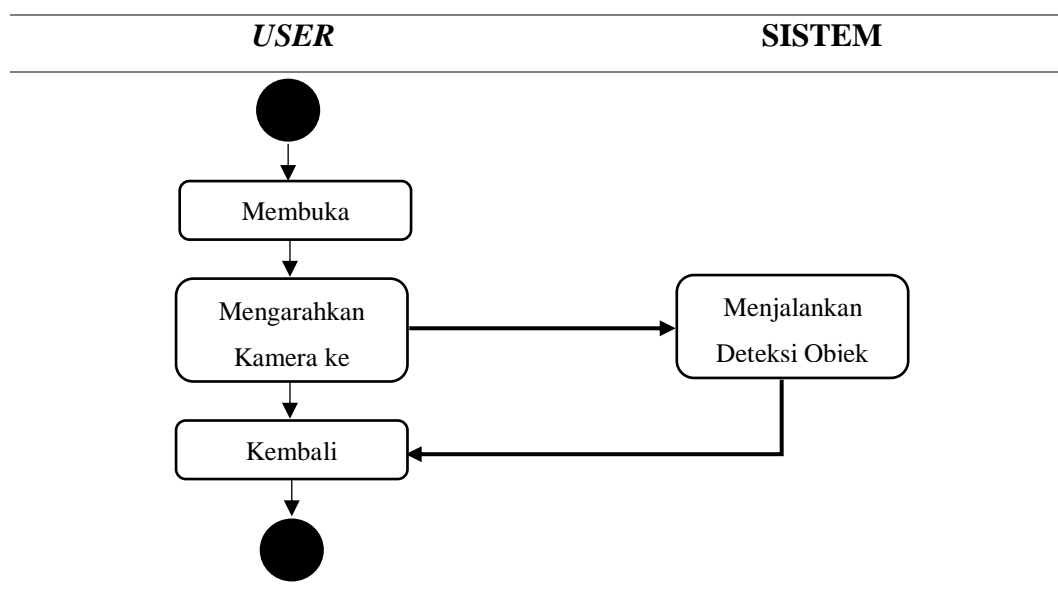
Nama Use Case	Deskripsi
Scan Objek	Adalah fungsi untuk melakukan identifikasi gambar yang ditangkap oleh kamera dan meberikan pelabelan berupa apel, <i>blueberry</i> , <i>pear</i> , anggur, <i>lyche</i> , rambutan serta <i>bonding box object</i> .
Show Process	Adalah fungsi untuk melihat proses yang sedang dikerjakan oleh sistem saat kamera diarahkan ke objek.

3.3.2. Diagram Aktivitas

Diagram aktivitas atau *activity diagram* adalah diagram yang menjelaskan alur kerja selama sitem dijalankan. Diagram aktivitas mendeskripsikan aktivitas di dalam sistem yang telah dirancang. Diagram aktivitas medeskripsikan awal proses aplikasi dimulai, keputusan tindakan yang dilakukan oleh user dan bagaimana proses sistem berjalan serta hasil akhir dari proses tersebut.

1. Diagram Aktivitas Cara Penggunaan

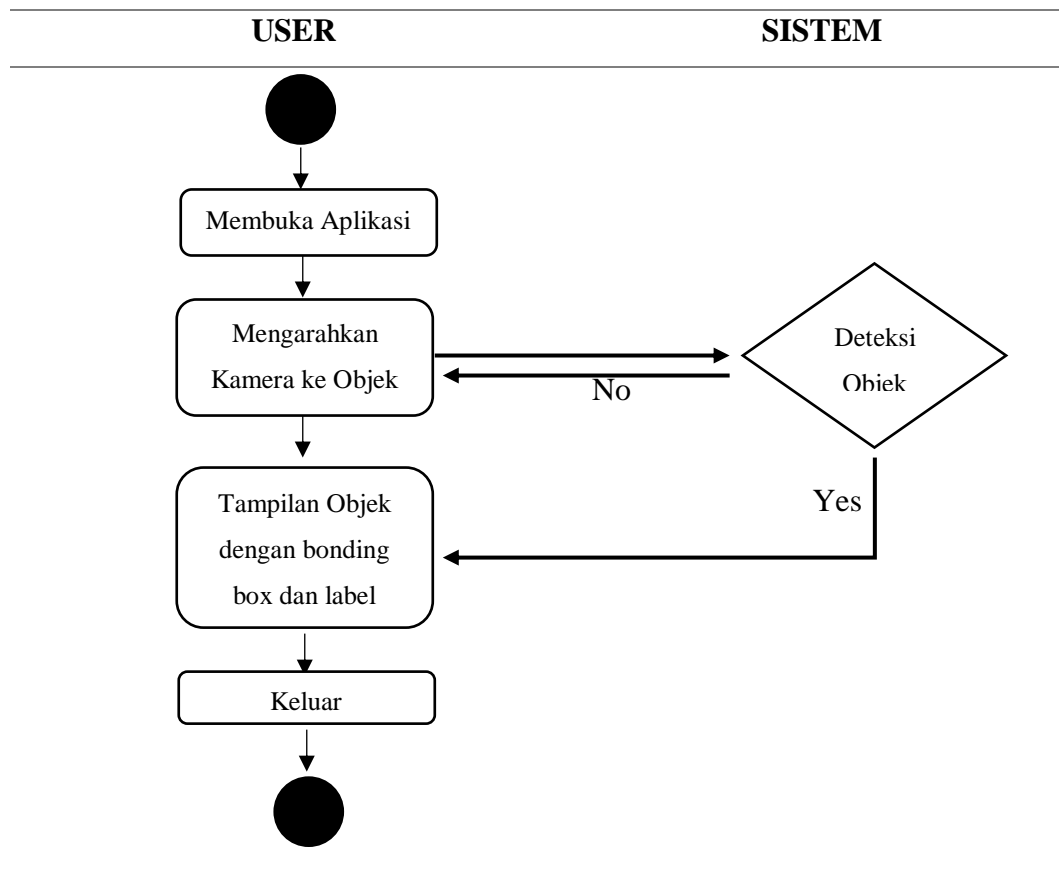
Menjelaskan aktivitas sistem untuk menjelaskan cara penggunaan aplikasi, beserta deskripsi objek. Diagram aktivitas cara penggunaan aplikasi dapat dilihat pada Tabel 3.3.

Tabel 3.3 Tabel Diagram Aktivitas Penggunaan

2. Diagram Aktivitas Deteksi Objek

Menjelaskan aktivitas sistem untuk melihat semua objek dan label, beserta deskripsi objek. Diagram aktivitas mulai program dapat dilihat pada Tabel 3.4.

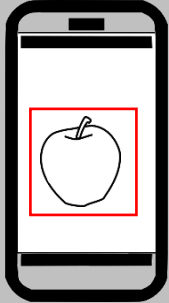
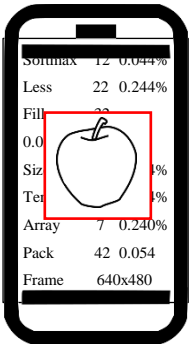
Tabel 3.4 Aktivitas Deteksi Objek



3.4. Perancangan Antarmuka

Pada bagian ini akan dibahas rancangan antarmuka aplikasi ini. Pada penelitian ini terdapat 2 *scene* perancangan antarmuka yaitu *Object Detection* dan *Show Process*. Adapun rancangan antarmuka pada penelitian ini yaitu dapat dilihat pada Tabel 3.5.

Tabel 3.5. Perancangan Antarmuka Aplikasi

Rancangan Antarmuka	Keterangan
<p>1. Halaman Utama</p> 	<p>Tampilan ini merupakan tampilan dari proses <i>object detection</i>, menu yang tersedia, yaitu :</p> <ol style="list-style-type: none"> Show Process, dapat menunjukkan process yang sedang dilakukan oleh sistem.
<p>2. <i>Show Process</i></p> 	<p>Tampilan ini merupakan dimana user dapat melihat <i>process</i> yang berjalan di background dengan cara menekan tombol volume pada android.</p>

BAB 4

IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi implementasi dan pengujian aplikasi berdasarkan analisis dan perancangan aplikasi yang telah dibahas pada bab sebelumnya. Pada tahap ini bertujuan untuk menampilkan hasil perancangan aplikasi yang telah dibangun dan proses pengujian aplikasi dalam melakukan pendeteksian pada objek buah - buahan.

4.1. Implementasi Sistem

4.1.1. Spesifikasi perangkat keras yang digunakan

Berikut adalah spesifikasi perangkat keras yang digunakan untuk membangun sistem ini :

Tabel 4.1. Spesifikasi perangkat keras

Perangkat Keras	Komponen
Aspire E5-475G	Intel Core i5 7200U Processor (@ 2.50 GHz)
	RAM (8 GB)
	Harddisk (500 GB)
	Display 14" HD (1366×768)
	Graphics Discrete graphics Nvidia GeForce 940MX 2GB
Google Cloud Computing	RAM (15 GB)
	Harddisk (50GB)
	Graphics Discrete graphics Nvidia K80
ASUS ZENPHONE 4 MAX	Performance Quad-core
	Display 5.5" (13.97 cm), Resolusi Layar 720 x 1280 pixels
	Internal 32 GB, 3 GB RAM

Camera 13 MP
Battery 4100 mAh
OS Android 7 (Nougat); ZenUI 4

4.1.2. Spesifikasi perangkat lunak yang digunakan

Berikut adalah spesifikasi perangkat lunak yang digunakan untuk membangun sistem ini :

Tabel 4.2. Spesifikasi perangkat lunak

Perangkat Keras	Komponen
Aspire E5-475G	Sistem Operasi (Windows 10)
	Miniconda 2
	Python 3.6.9
	Android Studio v3.5
	Tensorflow 1.13.1
	LabelImg
	SDK 26.1.1
Google Cloud Computing	Sistem Operasi Ubuntu 16.04
	Miniconda 2
	Python 3.6.9
	Tensorflow 1.13.1
	LabelImg
ASUS ZENFONE 4 MAX	OS Android 7 (Nougat)

4.1.3. Implementasi Data

Pada Lampiran 1 kita dapat melihat contoh dari data yang digunakan dalam proses *training* yang berjumlah 1147 data dimana 90 % data ini digunakan untuk *training* data dan 10 % untuk validasi data.

4.1.4. Pelabelan Objek

Pada tahap ini dilakukan pelabelan setiap objek yang akan dilatihkan ke sistem, pelabelan dilakukan menggunakan `labelImg` yang akan dimana setiap objek yang ada pada *image* akan diberikan *bounding box* dan juga kelas masing-masing objek. Tahap ini dapat kita lihat pada Lampiran 2. Proses pelabelan ini menghasilkan sebuah *file* xml untuk setiap objek pada *image* yang dilabeli, contoh *file* xml dari pelabelan dapat dilihat pada Gambar 4.1.

```
<annotation>
  <folder>train</folder>
  <filename>3 (1).jpg</filename>
  <path>E:\tensorflow2\models\research\object_detection\images\train
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>187</width>
    <height>187</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>anggur</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>33</xmin>
```

Gambar 4.1 Contoh Hasil Pelabelan

Gambar 4.1 merupakan contoh dari *file* xml yang dibuat dalam proses pelabelan dimana *file* ini berisi informasi seperti lebar, panjang, dan kelas dari objek yang kita labeli, setiap objek tersebut memiliki informasi yang berbeda-beda, apabila terdapat dua objek pada satu *image* maka kemungkinan akan ada dua objek dengan informasi yang berbeda yang akan dimasukkan ke *file* xml.



Gambar 4.2 Kumpulan *Image* dan *File XML* dari *Image*

Gambar 4.2 merupakan kumpulan dari *image* yang digunakan untuk proses *training* beserta file xml dari hasil pelabelannya. *File* ini nantinya akan dibuat menjadi file “*train.record*” yang akan dimasukkan kedalam sistem untuk dijadikan data *training* sistem.

4.1.5. Proses Training

Setelah masing masing *image* dilabeli dan memiliki *file xml* maka *file xml* ini akan di-*generate* menjadi dua *file csv* yaitu “*train_labels.csv*” dan “*test_label.csv*” yang berisi informasi dari semua *image* yang akan dijadikan data *training* dan *validasi*, contoh *file csv* ini dapat dilihat pada Gambar 4.3 dan 4.5.

1	filename	width	height	class	xmin	ymin	xmax	ymax
2	000477.jpg	375	500	Lyche	42	66	324	347
3	000479.jpg	500	375	Lyche	145	52	330	237
4	000480.jpg	500	375	Lyche	88	25	426	329
5	000481.jpg	375	500	Lyche	15	36	370	409
6	000482.jpg	500	333	Lyche	58	46	239	270
7	000482.jpg	500	333	Lyche	241	46	454	286
8	000483.jpg	374	500	Lyche	8	38	370	438
9	000484.jpg	374	500	Lyche	32	163	285	414
10	000485.jpg	500	500	Lyche	58	71	455	470
11	000486.jpg	500	375	Lyche	11	7	419	302
12	000487.jpg	500	464	Lyche	23	27	488	422
13	resizedP_201910	384	384	anggur	77	132	306	264
14	resizedP_201910	384	384	anggur	62	144	308	277
15	resizedP_201910	384	384	anggur	66	176	294	316
16	resizedP_201910	384	384	anggur	72	151	317	297
17	resizedP_201910	384	384	anggur	122	121	309	300
18	resizedP_201910	384	384	anggur	72	166	323	312
19	resizedP_201910	384	384	anggur	65	160	341	287
20	resizedP_201910	384	384	anggur	37	134	315	283
21	resizedP_201910	384	384	anggur	66	151	280	312
22	resizedP_201910	384	384	anggur	104	119	275	269

Gambar 4.3 Contoh isi File “*train_label.csv*”

Gambar 4.3 merupakan contoh dari file “train_label.csv” yang dibuat dari file xml yang dibuat pada saat pelabelan objek. File ini kemudian di-generate menjadi file “train.record” yang berisi informasi dari semua file xml setiap objek yang ada didalam image yang kita labeli. Pada penelitian ini jumlah dari data objek yang dibuat adalah 1771 data.

1	filename	width	height	class	xmin	ymin	xmax	ymax
2	000935.jpg	180	320	Rambutan	1	110	177	260
3	000936.jpg	180	320	Rambutan	34	121	152	270
4	000937.jpg	180	320	Rambutan	27	96	145	245
5	000938.jpg	180	320	Rambutan	17	106	158	266
6	000939.jpg	180	320	Rambutan	18	112	159	272
7	000940.jpg	180	320	Rambutan	16	151	158	271
8	000941.jpg	180	320	Rambutan	22	85	164	238
9	tlyche (5).jpg	510	340	Lyche	82	217	206	324
10	tlyche (5).jpg	510	340	Lyche	113	27	234	124
11	tlyche (5).jpg	510	340	Lyche	239	56	353	169
12	tlyche (6).jpg	356	340	Lyche	56	43	288	300
13	tlyche (7).jpg	509	340	Lyche	123	145	261	281
14	tlyche (7).jpg	509	340	Lyche	190	59	308	170
15	tlyche (7).jpg	509	340	Lyche	263	153	381	270
16	tlyche (8).jpg	680	340	Lyche	121	75	272	201
17	tlyche (8).jpg	680	340	Lyche	261	127	419	250
18	tlyche (8).jpg	680	340	Lyche	442	89	588	212
19	tlyche (8).jpg	680	340	Lyche	311	74	429	161
20	tlyche (9).jpg	510	340	Lyche	130	60	381	266

Gambar 4.4 Contoh isi “File test_label.csv”

Gambar 4.4 merupakan contoh dari file “test_label.csv” yang dibuat dari file xml yang dibuat pada saat pelabelan objek. File ini kemudian di-generate menjadi file “test.record” yang berisi informasi dari semua file xml setiap objek yang ada didalam image yang kita labeli. Pada penelitian ini jumlah dari data objek yang dibuat adalah 360 data.

Setelah file train.record dan test.record selesai dibuat, proses selanjutnya adalah membuat file “labelmap.pbtxt” yang berisi informasi kelas yang ada. Contoh isi labelmap.pbtxt dapat dilihat pada Gambar 4.5.

```

item {
  id: 1
  name: 'apel'
}

item {
  id: 2
  name: 'pear'
}

item {
  id: 3
  name: 'Rambutan'
}

item {
  id: 4
  name: 'Blueberry'
}

```

Gambar 4.5 Contoh isi File “*labelmap.pbtxt*”

Gambar 4.5 merupakan contoh isi file “*labelmap.pbtxt*”, file ini berfungsi untuk memberitahukan pada sistem bahwa kita memiliki 6 kelas yaitu apel, pir, rambutan, blueberry, anggur, leci. Proses *training* dilakukan menggunakan *pretrained-weight* dari *Faster R-CNN Googlenet Inception-v2* menggunakan tensorflow-gpu 1.13.1 selama 36 jam pada Aspire E5-475G dan mencapai lebih dari seratus ribu iterasi serta mencapai *loss function* < 0.005 . *Loss function* dihitung dengan Persamaan 4.1.

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (4.1)$$

i = indeks *anchor*

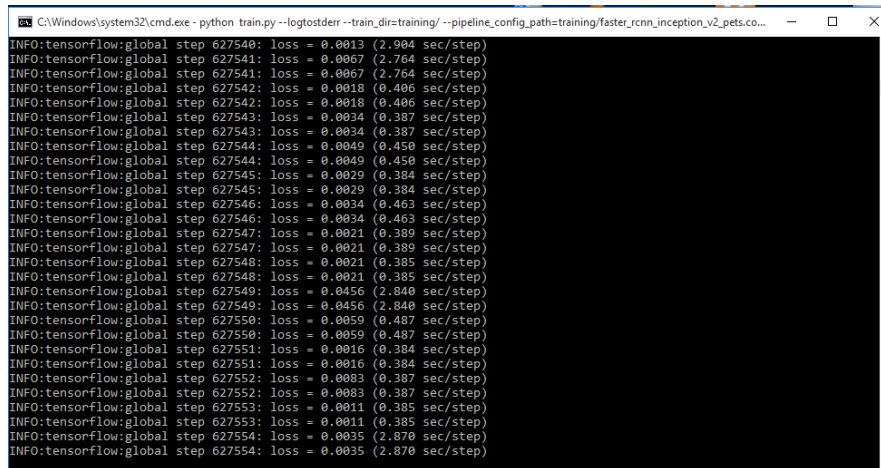
p_i = probabilitas indeks i sebagai objek

t_i = *ground truth box*

L_{cls} = *log loss over two classes*

L_{reg} = *loss over regression*

Proses *training* dan *function loss* dapat dilihat di Gambar 4.6 dan 4.7



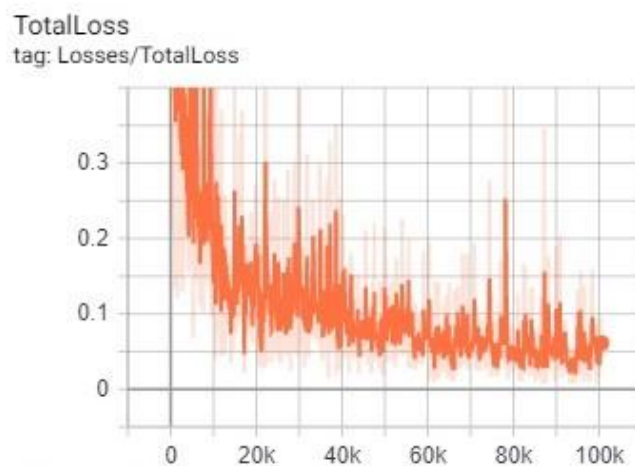
```

C:\Windows\system32\cmd.exe - python train.py --logtostderr --train_dir=training/ --pipeline_config_path=training/faster_rcnn_inception_v2_pets.co...
INFO:tensorflow:global step 627540: loss = 0.0013 (2.984 sec/step)
INFO:tensorflow:global step 627541: loss = 0.0067 (2.764 sec/step)
INFO:tensorflow:global step 627541: loss = 0.0067 (2.764 sec/step)
INFO:tensorflow:global step 627542: loss = 0.0018 (0.406 sec/step)
INFO:tensorflow:global step 627542: loss = 0.0018 (0.406 sec/step)
INFO:tensorflow:global step 627543: loss = 0.0034 (0.387 sec/step)
INFO:tensorflow:global step 627543: loss = 0.0034 (0.387 sec/step)
INFO:tensorflow:global step 627544: loss = 0.0049 (0.450 sec/step)
INFO:tensorflow:global step 627544: loss = 0.0049 (0.450 sec/step)
INFO:tensorflow:global step 627545: loss = 0.0029 (0.384 sec/step)
INFO:tensorflow:global step 627545: loss = 0.0029 (0.384 sec/step)
INFO:tensorflow:global step 627546: loss = 0.0034 (0.463 sec/step)
INFO:tensorflow:global step 627546: loss = 0.0034 (0.463 sec/step)
INFO:tensorflow:global step 627547: loss = 0.0021 (0.389 sec/step)
INFO:tensorflow:global step 627547: loss = 0.0021 (0.389 sec/step)
INFO:tensorflow:global step 627548: loss = 0.0021 (0.385 sec/step)
INFO:tensorflow:global step 627548: loss = 0.0021 (0.385 sec/step)
INFO:tensorflow:global step 627549: loss = 0.0456 (2.840 sec/step)
INFO:tensorflow:global step 627549: loss = 0.0456 (2.840 sec/step)
INFO:tensorflow:global step 627550: loss = 0.0059 (0.487 sec/step)
INFO:tensorflow:global step 627550: loss = 0.0059 (0.487 sec/step)
INFO:tensorflow:global step 627551: loss = 0.0016 (0.384 sec/step)
INFO:tensorflow:global step 627551: loss = 0.0016 (0.384 sec/step)
INFO:tensorflow:global step 627552: loss = 0.0083 (0.387 sec/step)
INFO:tensorflow:global step 627552: loss = 0.0083 (0.387 sec/step)
INFO:tensorflow:global step 627553: loss = 0.0011 (0.385 sec/step)
INFO:tensorflow:global step 627553: loss = 0.0011 (0.385 sec/step)
INFO:tensorflow:global step 627554: loss = 0.0035 (2.870 sec/step)
INFO:tensorflow:global step 627554: loss = 0.0035 (2.870 sec/step)

```

Gambar 4.6 Iterasi Training Data

Gambar 4.6 menunjukkan proses *training* data telah mencapai lebih dari ratusan ribu iterasi hal ini dilakukan agar tingkat *loss function* pada model semakin kecil, proses ini menunjukkan tingkat *loss* pada pelatihan kecil 0.005 dengan kecepatan rata-rata 0.5 detik per iterasi.



Gambar 4.7 Loss Function

Gambar 4.7 menunjukkan proses turunnya *loss function* yang terjadi selama *training data* dapat kita lihat semakin banyak iterasi yang dijalankan sistem maka semakin kecil jumlah *loss function* dari sistem. Parameter *Faster R-CNN* yang digunakan pada tahap pelatihan dapat dilihat pada Table 4.3.

Tabel 4.3 Parameter *Training*

No	Parameter	Keterangan
1	Input Node	1
2	Output Node	4
3	Iteration	200000
4	Learning Rate	0.0002
5	Loss function	0.005
6	IOU	0.75

Proses training dilakukan pada acer aspire E5-475G selama 36 jam dengan proses iterasi sebanyak 200000 dan learning rate 0.0002 dan menghasilkan *loss function* < 0.005. Proses *training* akan menghasilkan empat *file* yaitu “model.ckpt.index”, “model.ckpt.meta”, “model.ckpt.data” dan “graph.pbtxt”, keempat *file* ini merupakan informasi dari hasil *training* yang telah selesai dilakukan. Agar dapat digunakan pada android hasil *training* ini perlu diubah menjadi *frozen inference graph* yang merupakan model jadi yang tidak bisa di-*train* lagi hanya dapat digunakan sebagai model untuk pendeteksian objek, pada saat pembuatan *frozen model* ini juga akan mengurangi ukuran *file* dari hasil training dengan cara menghapus node yang hanya digunakan pada proses training pada graph, pada penelitian ini model *training* yang awalnya berukuran 100MB dapat dikurangi menjadi 50MB dengan cara menghapus seluruh *node-node* yang hanya digunakan selama fase *training*.

Setelah *frozen model* dibuat, maka *frozen model* ini akan dimasukkan kedalam aplikasi android dan akan dipanggil ketika program android dijalankan dan melakukan proses pendeteksian. Pada android, kelas-kelas yang terdapat pada model juga perlu dideklarasikan kembali karena didalam *frozen graph* kelas objek hanya dideklarasikan sebagai *array* bukan nama dari kelas objek tersebut, *file* pendeklarasian kelas-kelas didalam model ini dapat dilihat pada Gambar 4.8

```

|???
apel
pear
Rambutan
Blueberry
anggur
Lyche

```

Gambar 4.8 Pendeklarasian Kelas-Kelas Objek Pada Android

Gambar 4.8 merupakan proses pendeklarasian kelas-kelas yang terdapat pada model dengan membuat sebuah *file* “label.txt” yang akan dibaca oleh sistem, dimana kita membuat sebuah kelas “???” yang merupakan kelas dari *background* dan enam kelas dari model yang telah kita latih. Penulisan dari kelas ini harus sesuai dengan yang telah kita deklarasikan pada saat *training* dikarenakan sistem akan membaca kelas ini sesuai dengan nilai *array* yang tersimpan pada model.

4.1.6. Implementasi Perancangan Antarmuka

Implementasi perancangan antarmuka berdasarkan rancangan sistem yang telah dibahas pada Bab 3 adalah sebagai berikut:

1. Logo Aplikasi



Gambar 4.9 Logo Aplikasi

Gambar 4.9 merupakan tampilan dari logo aplikasi pada android. Aplikasi diberi nama *fruit detection* pada android.

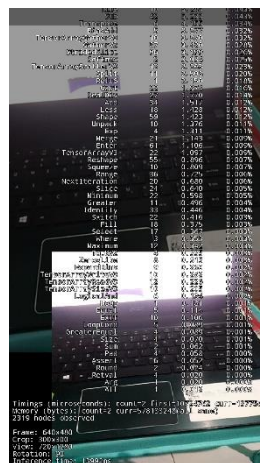
2. Tampilan Halaman Utama



Gambar 4.10 Halaman Utama

Halaman utama merupakan tampilan depan dari aplikasi, dimana aplikasi akan meminta izin akses untuk menggunakan kamera, dan setelah diberi akses aplikasi akan membuka kamera dan menjalankan pendeteksian objek terhadap objek yang ditangkap melalui kamera.

3. Tampilan *Show Process*



Gambar 4.11 Tampilan *Show Process*

Show process merupakan halaman dimana proses yang sedang dijalankan/dideteksi oleh sistem, objek yang sedang dideteksi dapat dilihat di sudut kiri dari layar android.

4.2. Prosedur Penggunaan

Tampilan awal aplikasi merupakan proses pendeteksian objek dimana aplikasi akan membuka kamera pada android seperti Gambar 4.10. Pada halaman ini pengguna hanya perlu mengarahkan kamera ke objek yang akan dideteksi dan sistem akan menjalankan proses pendeteksian objek, dimana proses pendeteksiaan ini akan berjalan sekitar 9 detik saat di uji di coba pada ASUS ZENPHONE 4 MAX dan memakan waktu selama 3 detik pada saat di uji pada Samsung A8+. Pada saat membuka aplikasi jika pengguna

menekan tombol *volume up* atau *down* maka akan muncul proses yang sedang berjalan di aplikasi dan *user* juga dapat melihat objek apa yang sedang dideteksi oleh kamera, tampilan ini dapat dilihat pada Gambar 4.11.

4.3. Pengujian Sistem

Pada tahap ini akan dilakukan pengujian terhadap data dan sistem untuk mengetahui kemampuan sistem yang dibangun dalam mendeteksi objek buah-buahan. Pengujian data dilakukan terhadap 224 gambar buah - buahan. Contoh hasil pelatihan terhadap pendeteksian objek ini dapat dilihat pada Lampiran 3 dan Lampiran 4. Pengujian juga dilakukan pada berbagai intensitas cahaya dan juga jarak kamera dari objek yang dapat kita lihat pada Lampiran 5 dan Lampiran 6.

Tabel 4.4 Confusion Matrix

	Apel	Pear	Rambutan	Blueberry	Anggur	Lyche	Total
Apel	45	0	0	1	0	0	46
Pear	0	37	0	0	0	0	37
Rambutan	0	0	31	0	0	2	33
Blueberry	0	0	0	31	2	0	33
Anggur	0	0	0	1	38	0	39
Lyche	0	0	0	3	0	33	36
Total	45	37	31	36	40	35	224

Untuk menghitung akurasi dari pengujian, persamaan yang digunakan adalah Persamaan 4.2.

$$\text{Akurasi} = \frac{\text{Jumlah data yang benar} \times 100 \%}{\text{jumlah seluruh data}} \quad (4.2)$$

Berdasarkan hasil pada Tabel 4.4, maka akurasi keseluruhan dapat dihitung. Akurasi keseluruhan yang diperoleh dengan Persamaan 4.2. digunakan sebagai berikut

$$\text{Akurasi} = \frac{215 \times 100\%}{224} = 95,98\%$$

Dari perhitungan di atas dapat diketahui bahwa tingkat akurasi dari metode *Faster R-CNN* dalam mengidentifikasi objek buah-buahan mencapai 95,98%. Namun terdapat kekurangan yang ada pada sistem yaitu tidak tertangkapnya seluruh objek yang ada pada *image* yang telah ditangkap kamera android.

BAB 5

PENUTUP

Bab ini membahas tentang kesimpulan dari penggunaan metode *Faster R-CNN* dalam proses pendeteksian objek buah - buahan serta saran – saran untuk pengembangan pada penelitian berikutnya.

5.1. Kesimpulan

Setelah melakukan tahap-tahap pada implementasi dan pengujian, maka akan diperoleh beberapa kesimpulan yang terdapat pada penelitian ini, antara lain yaitu:

1. Metode *Faster R-CNN* mampu melakukan pendeteksian objek buah-buahan dan membedakan objek buah-buahan yang memiliki kemiripan dengan akurasi yang baik sebesar 95,98%.
2. Berdasarkan hasil percobaan untuk pemilihan parameter yang digunakan, *learning rate* 0.0002 merupakan nilai yang baik dalam pendeteksian objek pada tahap pelatihan.
3. Kegagalan pendeteksian objek pada data uji disebabkan oleh jarak kamera dan intensitas cahaya sehingga kamera tidak dapat menangkap objek secara baik karena objek menjadi terlihat kecil ataupun tidak terlihat jelas sehingga tidak dapat dikenali sistem.
4. Pada proses pengujian sistem dapat mendeteksi objek tidak hanya berupa gambar dari buah-buahan tetapi juga buah-buahan asli yang ditangkap dari kamera android.

5.2. Saran

Saran yang dapat diberikan penulis untuk pengembangan penelitian selanjutnya adalah sebagai berikut:

1. Diharapkan pada penelitian selanjutnya untuk dapat mempercepat proses pengidentifikasin objek pada smartphone dalam penggunaan *Faster R-CNN*.
2. Diharapkan pada penelitian selanjutnya dapat menambahkan objek buah atau objek lainnya agar pengenalan objek lebih banyak.

3. Diharapkan pada penelitian selanjutnya dapat digabungkan dengan robot sehingga dapat menjadi alat penyortiran otomatis dimana aplikasi ini dapat dibuat menjadi mata dari mesin.

DAFTAR PUSTAKA

- Android. (2019). *Meet Android Studio*. (Online)
<https://developer.android.com/studio/intro> (8 Oktober 2019).
- Beibei Zhu, Xiaoyu Wu, Lei Yang, Yinghua Shen and Linglin Wu, Automatic detection of books based on Faster R-CNN," 2016 Third International Conference on Digital Information Processing, Data Mining, and Wireless Communications (DIPDMWC), Moscow, 2016, pp. 8-12. doi: 10.1109/DIPDMWC.2016.7529355.
- Ben-david, S. (2014). *Understanding Machine Learning : From Theory to Algorithm*.
- B. Liu, W. Zhao and Q. Sun, "Study of object detection based on Faster R-CNN," 2017 Chinese Automation Congress (CAC), Jinan, 2017, pp. 6233-6236. doi: 10.1109/CAC.2017.8243900.
- Fausett, Laurene. 1994. Fundamentals of Neural Networks Architectures, Algorithms, and Applications. London: Prentice Hall, Inc.
- Gonzales, R.C. & Woods, R.E. 2008. *Digital Image Processing 3rd Edition*. Prentice Hall: New Jersey.
- Google (2019). *Apel, Pear, Blueberry, Anggur, Lyche, Rambutan* (Online)
<https://www.google.com/?hl=in> (5 Agustus 2019).
- Gandi. R. (2018). R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms. (Online)
<https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e> (8 Oktober 2019).
- Devikar, P. 2016. Transfer Learning for Image Classification of Various Dog Breeds. International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), Vol.5: 2707-2715.
- Karpathy, A. (2018). Introduction to Convolutional Neural Networks.







- Kadir, Abdul & Susanto, Adhi. (2013). Teori dan Aplikasi Pengolahan Citra.
- Lukman, Andi., Marwana (2014) Machine Learning Multi Klasifikasi Citra Digital. Konferensi Nasional Ilmu Komputer.
- Prasetyo, Eko. "Data Mining Konsep dan Aplikasi Menggunakan MATLAB." *Yogyakarta: Andi* (2012).
- R. Gavrilescu, C. Zet, C. Foşalău, M. Skoczylas and D. Cotovanu, "Faster R-CNN:an Approach to Real-Time Object Detection," *2018 International Conference and Exposition on Electrical And Power Engineering (EPE)*, Iasi, 2018, pp. 0165-0168. doi: 10.1109/ICEPE.2018.8559776.
- Raj Bharath, (2018) A Simple Guide to the Versions of the Inception Network. (Online). <https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202> (8 Oktober 2019)
- S. Hsu, C. Huang and C. Chuang, "Vehicle detection using simplified fast R-CNN," *2018 International Workshop on Advanced Image Technology (IWAIT)*, Chiang Mai, 2018, pp. 1-3. doi: 10.1109/IWAIT.2018.8369767.
- S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149, 1 June 2017. doi: 10.1109/TPAMI.2016.2577031.
- Tensorflow. (2019). Panduan Awal TF: *Toolkit*. (Online) <https://developers.google.com/machine-learning/crash-course/first-steps-with-tensorflow/toolkit hl=id> (8 Oktober 2019)
- White David, (2018), Inception Network Overview.
- Wahyuni, F. S., & Mantja, S. N. (2015). Implementasi Artificial Neural Network (ANN) Untuk Optical Character Recognition.
- W. Zhang, S. Wang, S. Thachan, J. Chen and Y. Qian, "Deconv R-CNN for Small Object Detection on Remote Sensing Images," *IGARSS 2018 - 2018 IEEE*

International Geoscience and Remote Sensing Symposium, Valencia, 2018, pp. 2483-2486.doi: 10.1109/IGARSS.2018.8517436.

X. Mai, H. Zhang and M. Q. - Meng, "Faster R-CNN with Classifier Fusion for Small Fruit Detection," 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 7166-7172. doi: 10.1109/ICRA.2018.8461130.

LAMPIRAN I

Contoh Data Yang Digunakan

No	Nama File	Citra	Jenis
1	resizedP_20191004_195449.jpg		Anggur
2	Pear (18).jpg		Pear
3	newapel (36).jpg		Apel
4	blue (9).jpg		Blueberry
5	000998.jpg		Rambutan
6	4 (16).jpg		Lyche

LAMPIRAN II

Contoh Pelabelan Objek

Sebelum Dilabeli

Setelah Dilabeli

Anggur



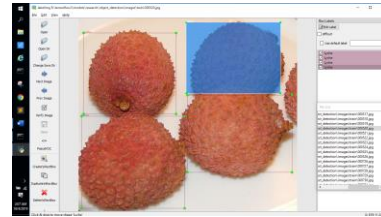
Anggur



Lyche



Lyche



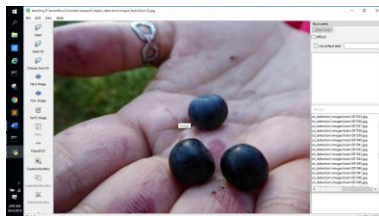
Rambutan



Rambutan



Blueberry



Blueberry



Pear



Pear



Sebelum Dilabeli

Sesudah Dilabeli

Apel



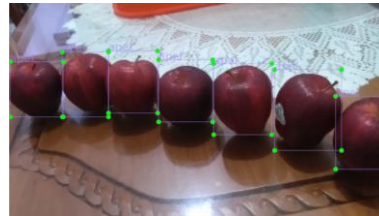
Apel



Apel

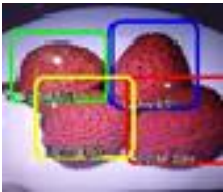
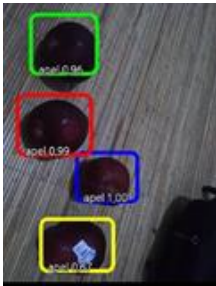









Apel










LAMPIRAN III

Contoh Hasil Pengujian

No	Citra	Desired Output	Actual Output
1		Lyche	Lyche
2		Apel	Apel
3		Apel dan Pear	Apel dan Pear
4		Apel, Pear, Apel	Apel , Pear, Apel




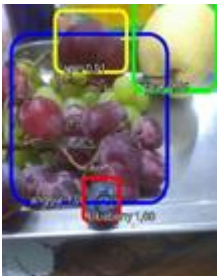
NO	Citra	Desired Output	Actual Output
5		Anggur	Anggur
6		Rambutan	Rambutan
7		Blueberry	Blueberry
8		Apel	Apel
9		Pear	Pear



NO	Citra	Desired Output	Actual Output
10	 Two lychees are shown. The top lychee is enclosed in a blue bounding box with the text 'Lychee 0.99' at the bottom. The bottom lychee is enclosed in a red bounding box with the text 'Lychee 1.00' at the bottom.	Lyche	Lyche
11	 A bunch of purple grapes is shown. A blue bounding box is drawn around the bunch with the text 'anggur 1.00' at the bottom.	Anggur	Anggur
12	 A single blueberry is shown. A blue bounding box is drawn around it with the text 'Blueberry 0.99' at the bottom.	Blueberry	Blueberry
13	 Two rambutans are shown. The left rambutan is enclosed in a red bounding box with the text 'Rambutan 0.84' at the bottom. The right rambutan is enclosed in a blue bounding box with the text 'Rambutan 0.84' at the bottom.	Rambutan	Rambutan
14	 A bunch of grapes with a green leaf is shown. A blue bounding box is drawn around the bunch with the text 'anggur 1.00' at the bottom.	Anggur	Anggur

NO	Citra	Desired Output	Actual Output
15		Rambutan	Rambutan, Leci
16		Leci	Leci

LAMPIRAN IV






Contoh Hasil Pengujian Grup






No	Citra	Desired Output	Actual Output
1		Blueberry, Anggur, Apel	Anggur, Apel
2		Anggur, Pear	Anggur, Pear
3		Apel, Pear, Anggur	Apel, Pear, Anggur
4		Apel, Pear, Anggur, Blueberry	Apel, Pear, Anggur, Blueberry

No	Citra	Desired Output	Actual Output
5		Apel, Pear, Anggur	Apel, Pear, Anggur
6		Apel, Anggur, Apel	Apel, Anggur, Apel

LAMPIRAN V







Contoh Pengujian Terhadap Intensitas Cahaya





Intensitas Cahaya	Citra	Keterangan
10 lux		sistem dapat mendeteksi citra objek
40 lux		sistem dapat mendeteksi citra objek
60 lux		sistem dapat mendeteksi citra objek
100 lux		sistem dapat mendeteksi citra objek
160 lux		sistem dapat mendeteksi citra objek

Intensitas Cahaya	Citra	Keterangan
1234 lux		sistem dapat mendeteksi citra objek
6265 lux		sistem dapat mendeteksi citra objek
7378 lux		sistem dapat mendeteksi citra objek
8175 lux		sistem dapat mendeteksi citra objek
19786 lux		sistem tidak dapat mendeteksi citra objek

LAMPIRAN VI

Contoh Hasil Pengujian Jarak

Jarak	Citra	Keterangan
10 cm		sistem dapat mendeteksi citra objek
20 cm		sistem dapat mendeteksi citra objek
30 cm		sistem dapat mendeteksi citra objek
40 cm		sistem dapat mendeteksi citra objek
50 cm		sistem dapat mendeteksi citra objek
60 cm		sistem dapat mendeteksi citra objek

Jarak	Citra	Keterangan
70 cm		sistem dapat mendeteksi citra objek
80 cm		sistem tidak dapat mendeteksi citra objek
100 cm		sistem dapat mendeteksi citra objek namun mengalami kesalahan dalam identifikasi objek
200 cm		sistem tidak dapat mendeteksi citra objek

