PROPOSAL TUGAS AKHIR

PERANCANGAN APLIKASI ANDROID PENDETEKSI MASKER MUKA DENGAN TEKNIK *DEEP LEARNING*

Diajukan untuk memenuhi prasyarat memperoleh gelar Sarjana Teknik
di Jurusan Teknik Elektro Universitas Jenderal Soedirman

Disusun oleh:

Muhammad Fakhrurrozi Sutisna
H1A017091

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS JENDERAL SOEDIRMAN
FAKULTAS TEKNIK
JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO
PURBALINGGA

2020

PROPOSAL TUGAS AKHIR

PERANCANGAN APLIKASI ANDROID PENDETEKSI MASKER MUKA DENGAN TEKNIK *DEEP LEARNING*

Diajukan untuk memenuhi prasyarat memperoleh gelar Sarjana Teknik
di Jurusan Teknik Elektro Universitas Jenderal Soedirman

Disusun oleh:

Muhammad Fakhrurrozi Sutisna
H1A017091

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS JENDERAL SOEDIRMAN
FAKULTAS TEKNIK
JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO
PURBALINGGA

2020

HALAMAN PENGESAHAN

Proposal Tugas Akhir dengan Judul:

PERANCANGAN APLIKASI ANDROID PENDETEKSI MASKER MUKA DENGAN TEKNIK *DEEP LEARNING*

Disusun oleh:
Muhammad Fakhrurrozi Sutisna
H1A017091

Diajukan untuk memenuhi salah satu persyaratan
memperoleh gelar Sarjana Teknik pada
Jurusan/Program Studi Teknik Elektro
Fakultas Teknik
Universitas Jenderal Soedirman

Disetujui dan disahkan
Pada Tanggal : _____

Pembimbing I	Pembimbing II
Imron Rosyadi, S.T., M. Sc.	Muhammad Syaiful Aliim, S.T., M.T.
NIP. 197909242003121003	NIP. 199009052019031021

Mengetahui:
Dekan Fakultas Teknik

Prof. Dr. Eng. Suroso, S. T., M. Eng.
NIP. 197812242001121002

HALAMAN PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Proposal Tugas Akhir dengan judul **"PERANCANGAN APLIKASI ANDROID PENDETEKSI MASKER MUKA DENGAN TEKNIK *DEEP LEARNING*"** ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Purbalingga, 10 November 2020

[materai sesuai ketentuan uu]
Ttd.

Muhammad Fakhrurrozi Sutisna
NIM. H1A017091

HALAMAN MOTTO DAN PERSEMBAHAN

MOTTO

Selalu berusaha melakukan yang terbaik, karena tidak akan pernah ada usaha yang sia-sia

PERSEMBAHAN

Proposal penelitian ini ditulis dan dipersembahkan untuk:

1. orang tua dan keluarga,
2. seluruh mahasiswa Teknik Unsoed, dan
3. siapapun yang membutuhkan hasil penelitian ini.

RINGKASAN

PERANCANGAN APLIKASI ANDROID PENDETEKSI MASKER MUKA DENGAN TEKNIK *DEEP LEARNING*

Muhammad Fakhrurrozi Sutisna

Coronavirus Disease 2019 (Covid-19) merupakan penyakit menular yang disebabkan oleh virus corona jenis baru, penyakit ini dapat menular dengan cepat secara *droplet*. Sehingga semua orang harus menjaga dan melindungi diri dengan berbagai macam protokol kesehatan, salah satunya adalah memakai masker ketika keluar rumah. Dalam kebiasaan baru penggunaan masker muka merupakan suatu kewajiban. Maka dari itu diperlukan sebuah aplikasi untuk mendeteksi penggunaan masker muka agar orang-orang lebih disiplin dan peduli terhadap penggunaan masker muka.

Pendeteksi masker muka dibangun dengan menggunakan teknik *deep learning*. Dengan arsitektur yang digunakan yaitu jenis *MobileNetV2* dan *VGG16Net* serta menggunakan deteksi muka dengan *SSD ResNet* dan *MTCNN*. Pelatihan dan pengujian model dilakukan pada infrastruktur *Google Colaboratory*. Hasil dari pelatihan model tersebut kemudian disimpan dan dikonversi ke dalam bentuk *file TensorFlow Lite* yang selanjutnya diimpor ke dalam *project* pada *Android Studio* agar dapat diimplementasikan pada aplikasi Android.

Kata kunci : deteksi masker muka, *deep learning*, *MobileNetV2*, *VGG16Net*, *Google Colaboratory*, Andorid

SUMMARY

DESIGNING AN ANDROID APPLICATION FOR FACE MASK DETECTION WITH DEEP LEARNING TECHNIQUES

Muhammad Fakhrurrozi Sutisna

Coronavirus Disease 2019 (Covid-19) is an infectious disease caused by a newly of coronavirus, this disease can be transmitted quickly in droplets. So that everyone must maintain and protect themselves with various kinds of health protocols, one of which is wearing a face mask when leaving the house. In the new normal, using face masks is also an obligation. Therefore an application is needed to detect the use of face masks so that people are more disciplined and care about using face masks.

Face mask detection is built using deep learning techniques. The architecture used is the type of MobileNetV2 and VGG16Net also using face detection with SSD ResNet and MTCNN. Model training and testing on the Google Colaboratory infrastructure. The results of the model training are then stored and converted into a TensorFlow Lite file which is then imported into the project in Android Studio, so that it can be implemented in the Android application.

Keywords : face mask detection, deep learning, *MobileNetV2*, *VGG16Net*, *Google Colaboratory*, *Android*

PRAKATA

Puji syukur kehadirat Allah S.W.T. yang telah melimpahkan keberkahan dan rahmat-Nya sehingga penulis mampu menyelesaikan proposal penelitian tugas akhir dengan judul, "**Perancangan Aplikasi Android Pendeteksi Masker Muka dengan Teknik *Deep Learning***".

Proposal penelitian tugas akhir ini disusun untuk memenuhi salah satu persyaratan memperoleh gelar sarjana Teknik di Jurusan Teknik Elektro Universitas Jenderal Soedirman. Dengan adanya tugas akhir, penulis mendapatkan banyak wawasan dan pengetahuan serta pengalaman baru serta dapat menggali ilmu lebih banyak dari pada bangku perkuliahan.

Segala kendala yang dialami penulis dalam proses penyusunan proposal penelitian ini dapat teratasi berkat bantuan dan dukungan dari berbagai pihak. Maka penulis mengucapkan terimakasih kepada:

1. Allah SWT yang selalu memberikan rahmat dan karunia-Nya sehingga selalu memberikan jalan yang terbaik dalam menjalani kehidupan.
2. Kedua orangtua dan keluarga yang selalu mendoakan hal baik serta memberikan segala dukungan baik moral dan juga material.
3. Ibu Farida Asriani, S.Si., M.T. selaku ketua jurusan Teknik Elektro Universitas Jenderal Soedirman.
4. Bapak Imron Rosyadi, S.T., M.Sc. selaku dosen pembimbing I.
5. Bapak Muhammad Syaiful Aliim, S.T., M.T. selaku dosen pembimbing II.
6. Teman-teman jurusan Teknik Elektro Universitas Jenderal Soedirman yang telah memberikan dukungan.
7. Pihak-pihak yang tidak disebutkan satu persatu kami ucapkan terimakasih.

Penulis menyadari bahwa proposal tugas akhir ini masih jauh dari kata sempurna, oleh sebab itu penulis membutuhkan saran dan masukan yang membangun agar dapat memperbaiki proposal ini.

Purbalingga, 10 November 2020

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

DAFTAR ISTILAH DAN SINGKATAN

CNN : *Convolutional Nerual Network*, merupakan salah satu metode dari deep learning

DNN : Deep Neural Network

MCTNN : Multi-task Cascaded Convolutional Neural Network

Covid-19 : *Coronavirus Disease* 2019, merupakan penyakit menular yang disebabkan oleh Coronavirus jenis baru

MERS : *Middle East Respiratory Syndrome*

SARS : *Severe Acute Respiratory Syndrome*

Droplet : Partikel berisi air dengan diameter >5-10 µm

Dataset : Obyek yang merepresentasikan data dan relasinya di memori

Python : Bahasa pemrograman tingkat tinggi yang bersifat open source

CPU : Central Processing Unit

GPU : Graphical Processing Unit

API : Application Programming Interface

DAFTAR SIMBOL

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Coronavirus Disease 2019 (Covid-19) merupakan penyakit menular yang disebabkan oleh virus corona jenis baru, setidaknya ada dua jenis virus corona yang diketahui menyebabkan penyakit yang dapat menimbulkan gejala berat seperti *Middle East Respiratory Syndrome* (MERS) dan *Severe Acute Respiratory Syndrome* (SARS) [1]. Sehingga diketahui bahwa virus ini dapat menular begitu cepat. Ketika seseorang terkena infeksi Covid-19 maka akan mengalami beberapa gejala umum antara lain gejala gangguan pernapasan akut seperti demam, batuk, dan sesak napas. Kasus ini diduga berhubungan dengan Pasar Ikan di Wuhan, China pada akhir Desember 2019 [2].

Berdasarkan studi epidemiologi dan virologi, membuktikan bahwa Covid-19 utamanya ditularkan dari orang yang bergejala (simptomatik) ke orang lain yang berada jarak dekat melalui *droplet* [2]. Penularan *droplet* terjadi ketika seseorang berada pada jarak yang dekat yaitu sekitar 1 (satu) meter dengan seseorang yang memiliki gejala pernapasan seperti batuk atau bersin. Sehingga *droplet* berisiko akan mengenai mulut dan hidung. Oleh karena itu, diperlukan pencegahan Covid-19 dengan beberapa cara seperti memakai masker, menjaga jarak, rajin mencuci tangan, dan selalu menerapkan protokol kesehatan.

Salah satu keharusan menerapkan protokol kesehatan yaitu dengan menggunakan masker ketika keluar dari rumah. Penggunaan masker merupakan cara perlindungan diri dan untuk saat ini diwajibkan untuk semua orang, baik orang

sehat maupun sakit. Orang sehat dapat menggunakan masker kain saat hendak keluar rumah, sedangkan bagi orang yang memiliki gejala infeksi pernapasan seperti batuk atau bersin, dicurigai memiliki gejala Covid-19, dan petugas kesehatan menggunakan masker bedah [3]. Namun, banyak sekali orang-orang yang masih mengabaikan penggunaan masker ketika memasuki sebuah tempat yang ramai. Maka diperlukan sebuah terobosan untuk deteksi masker pada suatu tempat seperti ketika memasuki lingkungan kampus, atau pun tempat-tempat umum lainnya agar penggunaan masker dapat lebih terkontrol dengan baik.

Machine learning adalah sebuah cabang ilmu kecerdasan buatan yang bertujuan untuk memungkinkan sebuah mesin dapat melakukan pekerjaannya dengan terampil dengan menggunakan perangkat lunak yang cerdas [4]. *Machine learning* nantinya dapat melakukan tugas-tugas tertentu dengan cara mempelajari algoritma dan model statistika yang ada. Pada *machine learning* terdapat tiga tipe yaitu *Supervised Learning*, *Unsupervised Learning* dan *Deep Learning.*

Deep learning merupakan sebuah kecerdasan buatan yang memiliki banyak lapisan tersembunyi untuk memproses data yang masih mentah [5]. Dasarnya *deep learning* menggunakan representasi yang sederhana tetapi dapat membangun konsep yang kompleks. Sehingga *deep learning* dapat dilatih untuk mendeteksi suatu objek serta klasifikasi objek. Terdapat beberapa algoritma yang dapat digunakan untuk melakukan pelatihan model sebuah *deep learning* misalnya untuk klasifikasi objek dengan *MobileNet*, *VGGNet*, dan sebagainya sedangkan untuk deteksi objek dapat menggunakan *YOLO*, *SSD ResNet*, *MTCNN*, dan sebagainya.

Membentuk suatu sistem pendeteksi objek dapat menggunakan dua kombinasi algoritma yaitu dengan melakukan klasifikasi kemudian mendeteksi objek tersebut. Maka perlu dilakukan teknik pelatihan untuk klasifikasi objek dengan menggunakan *Convolutional Neural Network* (CNN), yaitu sebuah pengembangan dari *Multilayer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi [6]. Setelah dilakukan pelatihan model dengan CNN maka selanjutnya model dikombinasikan dengan model untuk mendeteksi suatu objek. Ada banyak model yang dapat digunakan misalnya *Multi-task Cascaded Convolutional Neural Network* (MTCNN) dan modul *SSD ResNet*.

Perkembangan teknologi tentunya sangat cepat, ponsel cerdas menjadi bagian besar dari kehidupan sehari-hari bagi banyak orang di seluruh dunia. Berdasarkan survei [7] jumlah pengguna ponsel cerdas yaitu mencapai 3 miliar dan angka tersebut diperkirakan akan meningkat dalam beberapa tahun mendatang. Dengan menerapkan *deep learning* pada ponsel cerdas *android*, deteksi masker muka akan menjadi lebih mudah dan fleksibel yang mana dapat dilakukan oleh sebagaian besar pengguna karena hanya dengan menggunakan sebuah perangkat *android*.

Sehingga berdasarkan yang telah dipaparkan sebelumnya, penulis akan melakukan penelitian tugas akhir dengan judul "PERANCANGAN APLIKASI ANDROID PENDETEKSI MASKER MUKA DENGAN TEKNIK *DEEP LEARNING*".

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah dalam penelitian adalah sebagai berikut.

1. Bagaimana perancangan arsitektur *deep learning* untuk pendeteksi masker muka pada perangkat *Android*?
2. Bagaimana pengujian arsitektur *deep learning* untuk deteksi masker muka pada perangkat *Android*?

1.3 Batasan Masalah

Agar penelitian tugas akhir mendapatkan hasil yang optimal maka permasalahan dibatasi sebagai berikut.

1. Metode *deep learning* yang digunakan untuk pendeteksi masker muka terdiri dari dua komponen yaitu algoritma pendeteksi muka dengan *SSD ResNet* dan MTCNN, serta algoritma klasifikasi wajah dengan CCN (*MobileNetV2* dan *VGG16Net*).
2. Program dibuat untuk membedakan pengguna memakai masker atau tidak memakai masker secara *realtime* menggunakan *Android*.
3. Program pendeteksi masker muka menggunakan bahasa pemrograman *Python* dengan antarmuka dan infrastruktur *Google Colaboratory*.
4. Pembuatan model *deep learning* pada deteksi masker muka dengan citra menggunakan *Framework Keras* dan *Tensorflow*.
5. Aplikasi *Android* untuk pendeteksi masker muka dibuat menggunakan *Android Studio* dengan bahasa pemrograman *Java*.

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

Dalam penelitian tugas akhir ini terdapat beberapa tujuan yaitu sebagai berikut.

1. Merancang arsitektur *deep learning* untuk pendeteksi masker muka dan penerapannya pada aplikasi *Android*.
2. Menguji arsitektur *deep learning* untuk deteksi masker muka dengan perangkat *Android*.

1.4.2 Manfaat

Manfaat yang diharapkan dalam pelaksanaan penelitian tugas akhir ini adalah sebagai berikut.

1. Mampu menerapkan ilmu yang didapat pada bangku perkuliahan untuk menyelesaikan tugas akhir.
2. Dapat membantu mencegah penyebaran Covid-19.
3. Mempermudah petugas dalam melakukan pemeriksaan penggunaan masker.

1.5 Sistematik Penulisan

Adapun sistematika penulisan yang digunakan dalam penelitian ini, sebagai berikut.

BAB I Pendahuluan

Bab pendahuluan berisi tentang judul penelitian, latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, dan manfaat penelitian.

BAB II Tinjauan Pustaka

Bab Tinjauan Pustaka berisi tentang dasar kajian pustaka yang mendasari berbagai gagasan tentang penelitian terdahulu serta teori-teori pendukung

untuk penelitian seperti penjelasan tentang Covid-19, *deep learning*, *Convolutional Neural Network* (CNN), *Google Colaboratory*, *framework Keras* dan *TensorFlow*, sistem pengenalan wajah, *TensorFlow Lite*, dan *Android Studio*.

BAB III Metode Penelitian

Bab Metode Penelitian berisi tentang urutan langkah atau metode penelitian yang digunakan, meliputi waktu dan tempat penelitian, alat dan bahan yang digunakan, metode penelitian, sumber data, alur penelitian dan jadwal penelitian.

BAB IV Pembahasan

Bab Pembahasan berisi tentang hasil dan analisa dari penelitian perancangan aplikasi *Android* pendeteksi masker muka dengan teknik *deep learning*. Pada bab ini juga akan membahas hasil kajian dan analisa yang disesuaikan dengan rumusan masalah yang dibuat.

BAB V Penutup

Bab Penutup berisi tentang kesimpulan dan saran dari hasil penelitian yang telah dilaksanakan.

BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Dalam penyusunan penelitian tugas akhir "Deteksi Penggunaan Masker dengan Metode *Convolutional Neural Network* (CNN) Berbasis *Android* sebagai Bentuk Pencegahan Covid-19" terdapat beberapa penelitian terdahulu yang berhubungan yaitu membahas klasifikasi obyek dengan citra menggunakan *machine learning*. Penelitian tersebut dapat dilihat sebagai berikut.

1. Penelitian I Wayan Suartika E. P., Arya Yudhi Wijaya, dan Rully Soelaiman dengan judul "Klasifikasi Citra Menggunakan *Convolutional Neural Network* (CNN) pada *Caltech* 101" [6]. Penelitian ini melakukan klasifikasi 5 kategori unggas yaitu *Emu, Flamingo, Ibis, Pigeon dan Rooster* dengan 390 citra dari data *Caltech* 101 dan menggunakan 3 lapisan CNN, antara lain *convolutional layer*, *subsampling layer*, dan *fully connected layer*.
2. Penelitian Musakkarul Mu'minim Lambacing dan Ferdiansyah dengan judul "Rancang Bangun *New Normal* Covid-19 Masker Detektor dengan Notifikasi Telegram Berbasis *Internet of Things*" [8]. Penelitian ini melakukan pelatihan model dengan data dibagi menjadi 2 variabel yaitu *with* masker sebanyak 1915 citra dan *without* masker sebanyak 1918 citra. Kemudian menggunakan Raspberry Pi sebagai otak utamanya serta modul kamera juga PIR sensor sebagai deteksi orang dan maskernya. Selanjutnya data yang didapat akan dikirim melalui notifikasi Telegram.

3. Penelitian Mohamed Loey, Gunnasekaran Manogaran, Mohamed Hamed N. Taha dan Nour Eldeen M. Khalifa dengan judul "*A Hybrid Deep Transfer Learning Model with Machine Learning Methods For Face Mask Detection In The Era of The Covid-19 Pandemic*" [9]. Pada penelitian ini menggunakan dua model yaitu pertama dengan *transfer learning* arsitektur *ResNet50* yang digunakan untuk fase ekstraksi fitur. Kemudian menggunakan pembelajaran mesin tradisional seperti pohon keputusan, *Support Vector Machine* (SVM) dan algoritma ansambel yang digunakan untuk fase pelatihan, validasi dan pengujian. Dataset yang digunakan menggunakan tiga jenis yaitu wajah yang bermasker nyata, bermasker simulasi dan ketika berada di alam liar.
4. Penelitian Vidi Fitriansyah Hidarlan dengan judul "Rancang Bangun Klasifikasi Varietas Beras Berdasarkan Citra Menggunakan Metode *Convolutional Neural Network* (CNN) Berbasis *Android*" [10]. Pada penelitian ini membahas cara mengklasifikasikan 3 varietas beras yaitu Bashmati, IR64, dan Ketan dengan menggunakan arsitektur *VGG-16Net* dan *MobileNet* yang selanjutnya diaplikasikan pada *smartphone android*.

2.2 *Coronavirus Disease 2019* (Covid-19)

Coronavirus Disease 2019 (Covid-19) adalah penyakit menular yang disebabkan oleh *Severe Acute Respiratory Syndrome Coronavirus 2* (SARS-CoV-2) [1]. Ini merupakan virus corona jenis baru yang belum pernah diidentifikasi sebelumnya pada manusia. Tanda dan gejala ketika terinfeksi antara lain gejala pernapasan akut seperti demam, batuk, dan sesak napas. Dengan kasus yang berat

akan menyebabkan pneumonia, sindrom pernapasan akut, gagal ginjal dan bahkan kematian.

Pertama kali mendapatkan laporan Covid-19 berasal dari Kota Wuhan, Provinsi Hubei, China pada 31 Desember 2019. Selanjutnya tanggal 30 Januari 2020, *World Health Organization* (WHO) menetapkan kejadian tersebut sebagai Kedaruratan Kesehatan Masyarakat yang Meresahkan Dunia (KKMMD) dan pada tanggal 11 Maret 2020 ditetapkan Covid-19 sebagai pandemi. Sebab virus ini dapat menular sangat cepat dan sangat berbahaya.

2.2.1 Pencegahan dan Pengendalian Covid-19

Berdasarkan hal tersebut maka Pemerintah Indonesia mengeluarkan pedoman untuk melakukan pencegahan dan pengendalian penularan. Mengingat cara penularannya berdasarkan infeksi droplet dari individu ke individu, maka dapat terjadi dimana saja dan kapan saja masuk ke dalam tubuh melalui hidung, mulut dan mata. Untuk itu pencegahan penularan Covid-19 dapat dilakukan dengan beberapa tindakan berikut [2].

a. Membersihkan tangan secara teratur dengan cuci tangan pakai sabun dan air mengalir selama 40 – 60 detik atau menggunakan cairan antiseptik berbasis alkohol (*handsanitizer*) minimal 20 – 30 detik. Hindari menyentuh mata, hidung dan mulut dengan tangan yang tidak bersih.
b. Menggunakan alat pelindung diri berupa masker yang menutupi hidung dan mulut jika harus keluar rumah atau berinteraksi dengan orang lain yang tidak diketahui status kesehatannya.

c. Menjaga jarak minimal 1 meter dengan orang lain untuk menghindari terkena droplet dari orang yang batuk atau bersin.

d. Membatasi diri terhadap interaksi/kontak dengan orang lain yang tidak diketahui status kesehatannya.

e. Saat tiba di rumah setelah berpergian, segera mandi dan berganti pakaian sebelum kontak dengan anggota keluarga di rumah.

f. Meningkatkan daya tahan tubuh dengan menerapkan pola hidup bersih dan sehat (PHBS) seperti konsumsi gizi seimbang, aktivitas fisik minimal 30 menit sehari, istirahat yang cukup termasuk pemanfaatan kesehatan tradisional.

g. Mengelola kesehatan jiwa dan psikososial.

h. Apabila sakit menerapkan etika batuk dan bersin. Jika berlanjut segera berkonsultasi dengan dokter/tenaga kesehatan.

i. Menerapkan adaptasi kebiasaan baru dengan melaksanakan protokol kesehatan dalam setiap aktivitas.

2.2.2 Pentingnya Memakai Masker

Seperti yang diketahui bahwa penularan virus corona dapat melalui droplet atau percikan ketika seseorang berbicara, batuk, bersin, atau sebagainya. Oleh karenanya, masker dibuat untuk melindungi diri dari droplet yang dikeluarkan oleh orang lain agar tidak masuk ke hidung dan mulut kita atau sebaliknya. Terdapat tiga jenis masker yang disarankan kepada masyarakat agar dapat memutus penyebaran virus corona [11], antara lain sebagai berikut.

1. Masker Kain

   Sesuai anjuran Kementerian Kesehatan RI, masyarakat disarankan untuk memakai masker kain ketika harus berpergian ke luar rumah, misalnya saat harus bekerja atau membeli kebutuhan. Masker kain tetap dapat menghalau Sebagian percikan air liur (droplet) yang keluar saat berbicara, menghela napas atau batuk dan bersin.

2. Masker Bedah

   Jenis masker ini yaitu sekali pakai dan sering dijumpai dan digunakan oleh tenaga medis saat bertugas. Jika sedang sakit, lebih baik disarankan menggunakan masker dengan tiga fungsi ini karena lebih efektif dalam mencegah penyebaran penyakit menular seperti infeksi virus corona.

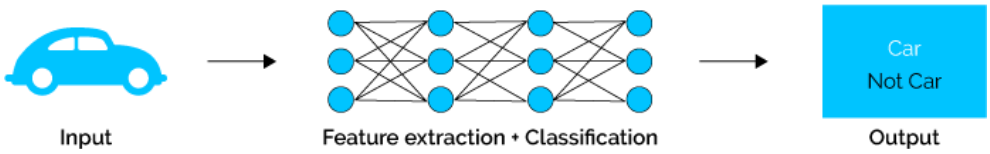
3. Masker N95

   Masker ini cenderung lebih mahal dari masker bedah, karena fungsinya yang sangat efektif untuk mencegah penularan virus corona. Selain dapat menghalau percikan air liur, tapi juga partikel kecil di udara yang mungkin mengandung virus. Namun tidak disarankan untuk penggunaan sehari-hari dan masker ini diutamakan untuk petugas medis yang memang kontak secara langsung dengan penderita Covid-19.

2.3 *Deep Learning*

Deep learning merupakan salah satu cabang dari ilmu pembelajaran mesin (*machine learning*) yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam [12]. Pada pengaplikasiannya *deep learning* bayak digunakan untuk *speech recognition* pada ponsel pintar, analisa video dan citra,

klasifikasi teks dan sebagainya. Teknik dan algoritma dalam *deep learning* dapat digunakan untuk kebutuhan *supervised learning* (pembelajaran terarah), *unsupervised learning* (pembelajaran tak terarah), dan *semi-supervised learning* (semi-terarah). Struktur dan jumlah jaringan saraf pada algoritmanya sangat banyak bisa mencapai ratusan lapisan. Struktur dari pemodelan jaringan pada *deep learning* ditampilkan pada Gambar 2.1 [12].



Gambar 2.1 Struktur arsitektur *deep learning*

Terdapat dua istilah penting dalam pembangunan model yaitu pelatihan dan pengujian [13]. Pelatihan atau *training* adalah proses konstruksi model sedangkan pengujian atau *testing* merupakan proses menguji kinerja dari model hasil pembelajaran. Pelatihan biasanya menggunakan sebuah kumpulan data atau biasa disebut dataset yang berupa sampel data dalam statistika, dapat berupa citra. Umumnya, *dataset* dibagi menjadi tiga jenis yang tidak beririsan antara lain sebagai berikut.

1. *Training set* merupakan sebuah himpunan data yang digunakan untuk melakukan pelatihan atau membangun sebuah model.
2. *Development set* atau biasa disebut dengan *validation set* merupakan sebuah himpunan data yang digunakan untuk mengoptimasi saat melatih model. Jadi ketika sebuah model sudah dibangun oleh *training set*, maka akan dilakukan validasi menggunakan *validation set*.. Sehingga akan memudahkan dalam proses generalisasi dan model mampu mengenali pola secara generik.

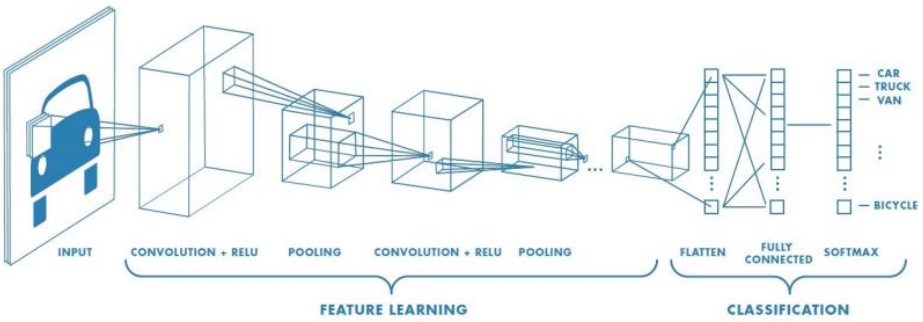
3. *Testing set* merupakan sebuah himpunan data yang digunakan untuk menguji model setelah proses pelatihan selesai.

Pembagian rasio sebuah *dataset* beragam macamnya sesuai dengan kondisi yang ada. Pada umumnya, rasio pembagian *dataset* (*training* : *validation* : *testing*) adalah (80% : 10% : 10%) atau (90% : 5% : 5%). Tetapi ketika *dataset* berukuran kecil maka *validation set* bisa tidak digunakan sehingga hanya dibagi menjadi *training set* dan *testing set* saja. Kasus seperti ini mengakibatkan ketika latihan *dataset* yang digunakan menggunakan *training set* dan dikenal sebagai *closed testing*. Rasio (*training*, *testing*) yang dapat digunakan yaitu (90% : 10%), (80% : 20%), (75% : 25%), atau bahkan (50% : 50%).

2.4 *Convolutional Neural Network* (CNN)

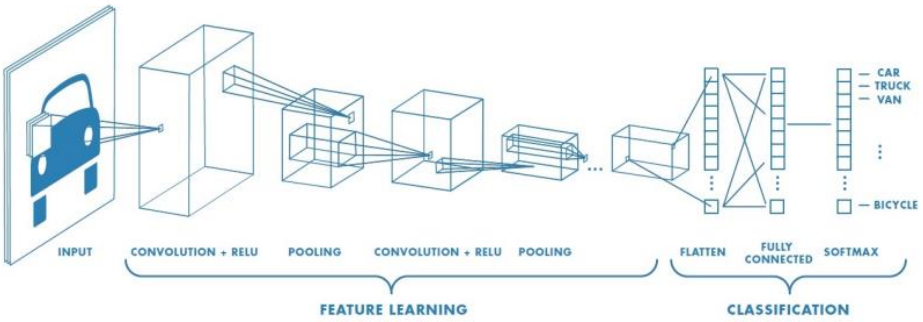
Convolutional Neural Network (CNN) adalah pengembangan dari *Multilayer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi [6]. CNN mampu mengenali informasi suatu objek seperti citra, teks, potongan suaran dan sebagainya berupa data. Namun paling banyak digunakan pada bidang pemrosesan citra.

Pada Gambar 2.2 dapat dilihat sebuah alur dari proses CNN ketika melakukan pemrosesan citra dengan *feature learning* menggunakan *convolution relu* kemudian *pooling* hingga proses klasifikasi citra dengan menggunakan *flatten*, *fully connected* dan *softmax*. sehingga citra dapat diklasifikasikan ke kategori tertentu berdasarkan nilai keluarannya.



Gambar 2.2 Alur Proses CNN

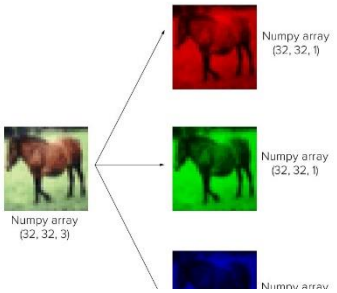
Berdasarkan



Gambar 2.2, maka CNN dapat dikategorikan memiliki lima komponen utama pada *layer* atau lapisannya yaitu sebagai berikut.

1. *Input Layer* (Lapisan Masukkan)

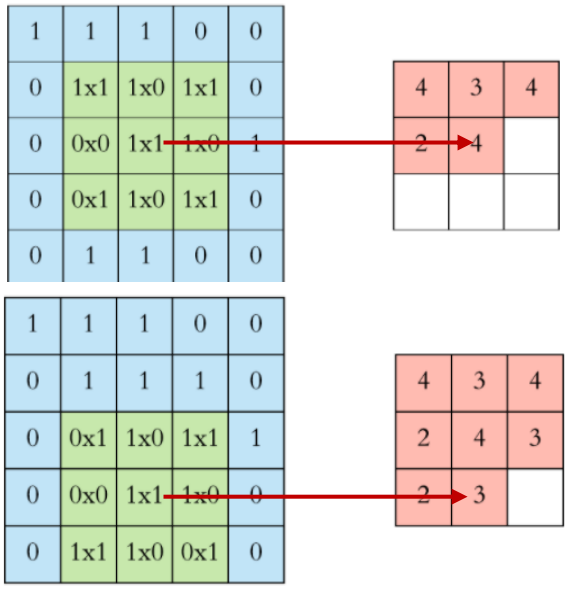
   Lapisan masukkan dapat berupa sebuah citra RGB (Red, Green, Blue) dengan ukuran 32 x 32 piksel yang sebenarnya merupakan sebuah multidimensional array dengan ukuran 32 x 32 x 3. Nilai 3 terakhir merupakan jumlah dari kanal. Contoh dari citra tersebut ditunjukkan pada Gambar 2.3.



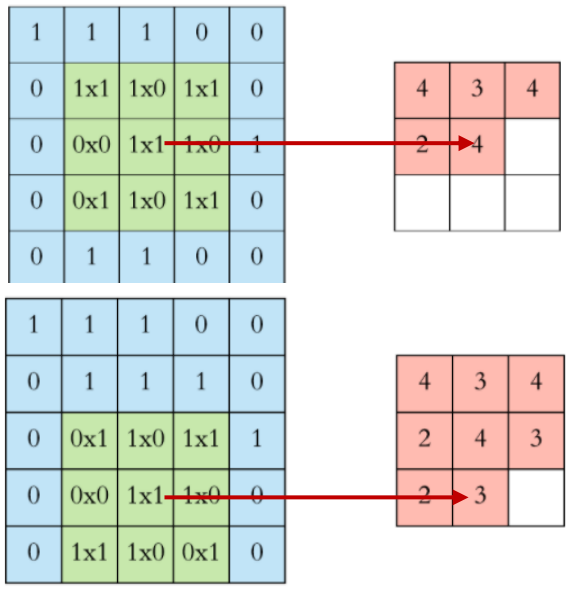
Gambar 2.3 Citra *Input Layer*

2. *Convolutional Layer* (Lapisan Konvolusi)

Lapisan yang pertama kali menerima masukkan citra langsung pada arsitektur. Operasi pada lapisan ini sama dengan operasi konvolusi yaitu melakukan operasi kombinasi linier filter terhadap daerah lokal [14]. Seperti layaknya citra, filter lapisan pada proses konvolusi memiliki ukuran tinggi, lebar dan tebal tertentu. Alur pada lapisan konvolusi dapat digambarkan seperti pada



Gambar 2.4.



Gambar 2.4 Alur *Convolutional Layer*

Lapisan konvolusi memiliki tiga *hyperparameter*, dimana *hyperparameter* pada lapisan ini menjadi sebuah acuan untuk menentukan dan ukuran hasil ekstraksi lapisan. Tiga *hyperparameter* ini antara lain yaitu *depth* (kedalaman atau jumlah lapisan konvolusi), *stride* (langkah atau jumlah pergeseran filter pada proses konvolusi), dan *zero-padding* yang merupakan jumlah penambahan nilai intensitas nol di daerah sekitar input gambar.

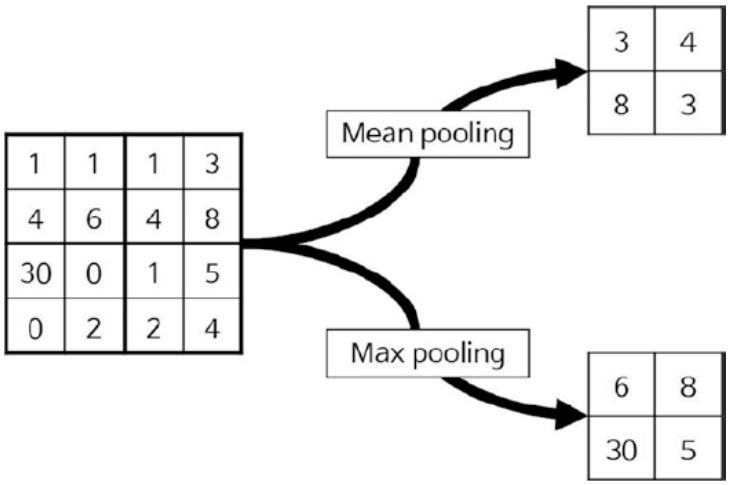
3. *Activation Layer* (Lapisan Aktivasi)

   Lapisan aktivasi adalah lapisan dimana *feature map* dimasukan ke dalam fungsi aktivasi [15]. Fungsi aktivasi digunakan untuk mengubah nilai-nilai pada *feature map* pada *range* tertentu sesuai dnegan fungsi aktivasi yang digunakan. Hal ini bertujuan untuk meneruskan nilai yang menampilkan fitur dominan dari citra yang masuk ke lapisan berikutnya. Terdapat beberapa fungsi aktivasi yang

umum untuk digunakan, namun dalam penelitian hanya menggunakan aktivasi *ReLu* dan *Softmax*.

4. *Pooling Layer*

Selanjutnya masukan dari lapisan aktivasi akan menuju *pooling layer*, kemudian lapisan ini akan mengurangi parameternya. *Pooling* juga bisa disebut sebagai *subsampling* atau *downsampling* yang mengurangi dimensi dari *feature map* tanpa menghilangkan informasi penting di dalamnya [15]. Proses dalam lapisan ini cukup sederhana, pertama akan ditentukan ukuran downsampling yang akan digunakan pada *feature map*, misalnya 2 x 2. Kemudian akan dilakukan proses *pooling* pada *feature map*. Proses dari *pooling* ada beberapa macam seperti *max pooling*, *mean pooling*, dan *sum pooling*. Contoh proses *pooling* ditunjukan pada Gambar 2.5.



Gambar 2.5 Matriks *feature map* 4x4 dengan proses *pooling* 2x2

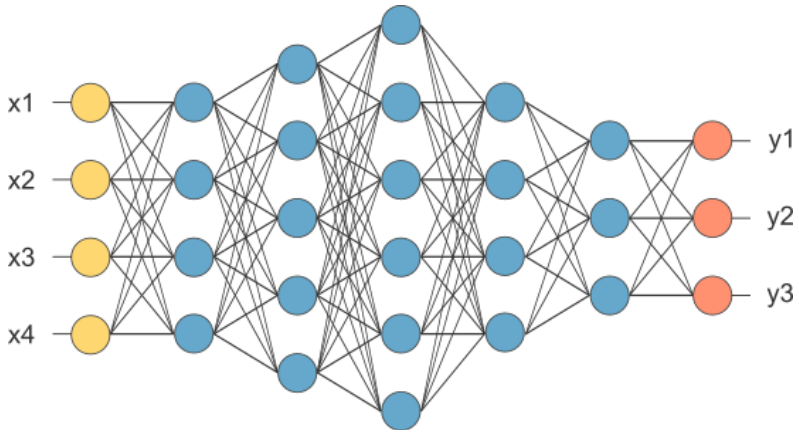
Setelah dilakukan *pooling layer*, maka dapat diketahui tujuan dari *pooling layer* tersebut yaitu untuk mengurangi dimensi dari *feature map*. Sehingga proses ini

akan mempercepat komputasi karena parameter yang harus diperbaharui semakin sedikit dan mengatasi *overfitting*.

5. *Fully Connected Layer*

   Setelah melewati proses lapisan-lapisan diatas, hasil dari *pooling layer* digunakan menjadi masukan untuk *fully connected layer*. Lapisan ini memiliki kesamaan struktur dengan *Artificial Neural Network* (ANN) yang pada umumnya memiliki lapisan masukkan, lapisan tersembunyi, dan lapisan keluaran yang masing-masing memiliki neuron-neuron yang saling berhubungan dengan neuron-neuron di lapisan tetangganya.

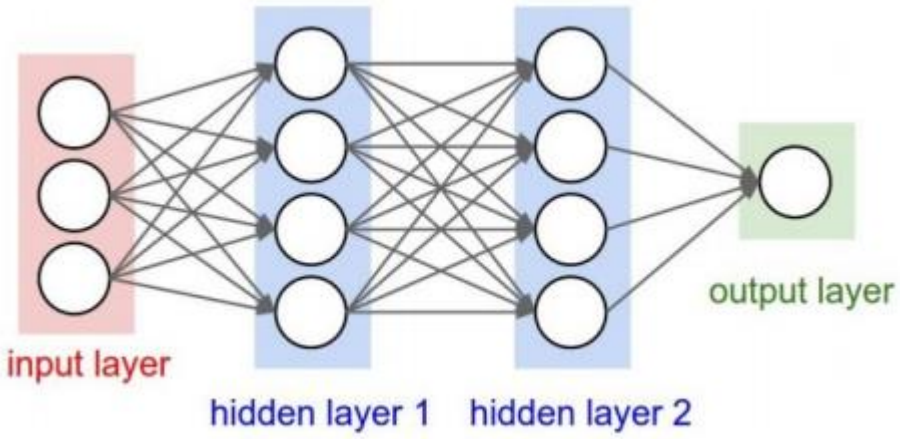
   Pada konsepnya sebelum pooling digunakan sebagai masukan, hasil *pooling* terlebih dahulu diubah menjadi vektor (x1, x2, x3, x4, dan seterusnya). Selanjutnya dari sini akan diproses ke dalam *fully connected layer*. Sehingga pada lapisan terakhir pada *fully connected layer* akan digunakan fungsi *ReLu* atau *softmax* untuk menentukan klasifikasi dari citra masukan yang dari lapisan masukan CNN. Untuk lebih jelas dapat perhatikan contoh *fully connected layer* pada Gambar 2.6.



Gambar 2.6 Contoh *fully connected layer*

2.4.1 Konsep *Convolutional Neural Network* (CNN)

Cara kerja pada CNN memiliki kesamaan dengan MLP, namun dalam CNN setiap neuron dipresentasikan dalam bentuk dua dimensi, tidak seperti MLP yang setiap neuron hanya berukuran satu dimensi [6]. Gambar 2.7 akan menunjukan sebuah arsitektur sederhana dari MLP.



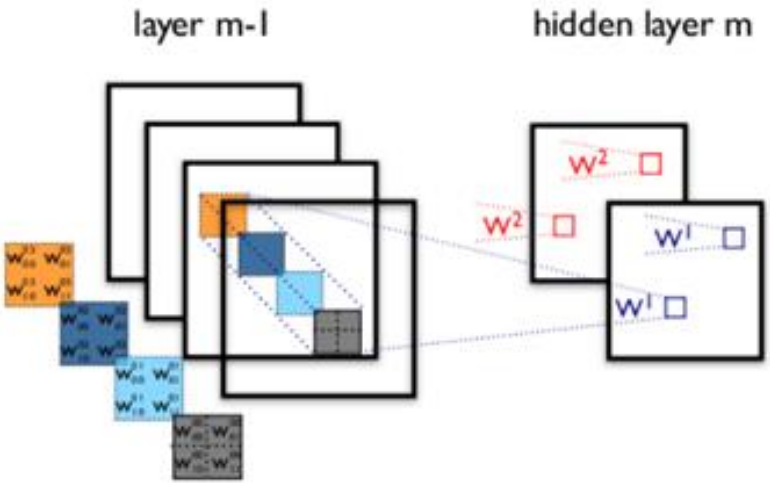
Gambar 2.7 Arsitektur Sederhana MLP

Seperti pada Gambar 2.7 MLP memiliki *i* lapisan (kotak merah dan biru) dengan masing-masing lapisan berisi j_i neuron (lingkaran putih). Ketika MLP menerima masukan data satu dimensi dan mempropagasikan data tersebut pada jaringan hingga menghasilkan keluaran. Setiap hubungan antar neuron pada dua lapisan yang bersebelahan memiliki parameter bobot satu dimensi yang menentukan kualitas mode. Di setiap data masukan pada lapisan dilakukan operasi linear dengan nilai bobot yang ada, kemudian hasil komputasi akan ditransformasikan menggunakan operasi non-linear yang disebut sebagai fungsi aktivasi.

Data yang dipropagasikan pada CNN adalah data dua dimensi sehingga operasi linear dan parameter bobot pada CNN berbeda yaitu menggunakan operasi

linear konvolusi dengan bobot yang tidak lagi satu dimensi. Namun berbentuk empat dimensi yang merupakan kumpulan kernel konvolusi seperti pada Gambar 2.8. Dimensi bobot pada CNN adalah:

neuron input x neuron output x tinggi x lebar



Gambar 2.8 Proses Konvolusi pada CNN

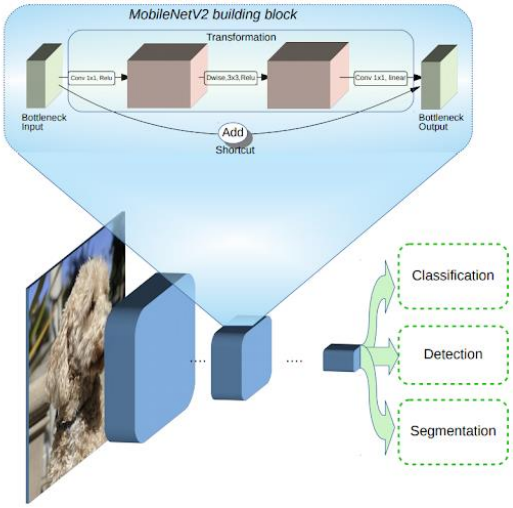
Oleh karena sifat proses konvolusi, maka CNN hanya dapat digunakan pada data yang memiliki struktur dua dimensi seperti citra dan suara.

2.4.2 Arsitektur Jaringan *MobileNetV2*

Salah satu implementasi dari *deep learning* yaitu digunakan dalam bidang *object detection* atau pendeteksian objek misalnya pada berbagai produk riset hingga produk komersil seperti *self driving car*, *cctv* cerdas, dan sebagainya. Namun penggunaan *deep learning* untuk *object detection* membutuhkan perangkat yang cukup mahal seperti GPU atau spesifikasi PC yang cukup tinggi. Salah satu arsitektur jaringan CNN yang dapat digunakan yaitu *MobileNetV2*.

Arsitektur *MobileNetV2* merupakan sebuah arsitektur pengembangan dari arsitektur *MobileNetV1* yang rilis pada April 2017 [16]. Sama seperti *MobileNetV1*,

MobileNetV2 masih menggunakan *depthwise* dan *pointwise convolution*. Dengan menambahkan dua fitur baru yaitu *linear bottleneck* dan *shortcut connections* antar *bottlenecks*. Jika digambarkan maka struktur dasar dari arsitektur *MobileNetV2* ditunjukkan pada Gambar 2.9.



Gambar 2.9 Arsitektur *MobileNetV2*

Berdasarkan Gambar 2.9, kotak biru menunjukkan blok pembentukan konvolusi *linear bottleneck*. Pada bagian tersebut terdapat masukan dan keluaran antara model sedangkan lapisan atau *layer* bagian dalam meng-enkapsulasi kemampuan model untuk mengubah input dari konsep tingkat yang lebih rendah misalnya piksel ke deskriptor tingkat yang lebih tinggi seperti kategori gambar. Sehingga pada akhirnya, seperti halnya koneksi residual pada CNN tradisional, *shortcut* antara *bottlenecks* memungkinkan *training* atau pelatihan yang lebih cepat dan akurasi yang lebih baik. Jika diuraikan maka arsitektur jaringan *MobileNetV2* didefinisikan dalam Tabel 2.1 [17].

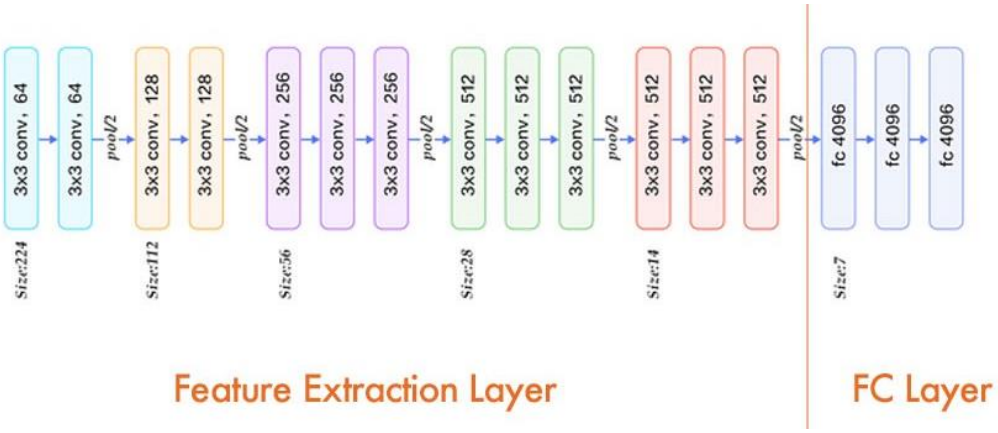
Tabel 2.1 Arsitektur *MobileNetV2*

Input	Operator	*t*	*c*	*n*	*s*
224^2 x 3	conv2d	-	32	1	2
112^2 x 32	bottleneck	1	16	1	1
112^2 x 16	bottleneck	6	24	2	2
56^2 x 24	bottleneck	6	32	3	2
28^2 x 32	bottleneck	6	64	4	2
14^2 x 64	bottleneck	6	96	3	1
14^2 x 96	bottleneck	6	160	3	2
7^2 x 160	bottleneck	6	320	1	1
7^2 x 320	conv2d 1 x 1	-	1280	1	1
7^2 x 1280	avgpool 7 x 7	-	-	1	-
1 x 1 x 1280	conv2d 1 x 1	-	k	-	

Setiap baris mendeskripsikan urutan dari satu atau lebih lapisan identik (*modulo stride*), diulang sebanyak *n* kali. Semua lapisan dalam urutan yang sama memiliki nomor saluran keluaran *c* yang sama. Lapisan pertama dari setiap urutan memiliki *stride s* dan yang lainnya menggunakan *stride* 1. Semua konvolusi spesial menggunkan kernel 3x3. Faktor ekspansi *t* selalu diterapkan ke ukuran masukan.

2.4.3 Arsitektur Jaringan VGG16Net

VGG16 merupakan model jaringan saraf konvolusi yang diusulkan oleh K. Simonyan dan A. Zisserman dari Universitas Oxford dalam tulisan "*Very Deep Convolutional Networks for Large-Scale Image Recognition*" [18]. Model *VGG16* mencapai akurasi pengujian ke 5 yaitu 92,7% di *ImageNet* yang merupakan *dataset* lebih dari 14 juta gambar dengan 1000 kelas. Salah satu model yang terkenal diajukan ke ILSVRC-2014. *Gambar 2.10* menunjukan arsitektur dari *VGG16*.



Gambar 2.10 Arsitektur *VGG16*

Pada *VGG16* memiliki 16 lapisan seperti namanya, dengan 4096-4096-4096 neuron pada *hidden layer* dan 1000 neuron pada *output layer*. Pada tiga lapisan *Fully-Connected* (FC) mengikuti tumpukan lapisan konvolusional (yang memiliki kedalaman berbeda dalam arsitektur yang berbeda) yaitu dua yang pertama memiliki ukuran masing-masing 4096 neuron kanal, yang ketiga melakukan klasifikasi ILSVRC 1000 keluaran dan dengan demikian memuat 1000 saluran (satu untuk setiap kelas). Lapisan terakhir adalah lapisan *soft-max*. Konfigurasi *fully connected layer* adalah sama di semua jaringan. Sedangkan pada *hidden layer* dilengkapi dengan fungsi aktivasi *ReLU*.

2.5 Sistem Pengenalan Wajah

Pengenalan wajah adalah suatu metode yang berorientasi pada wajah. Pengenalan ini dibagi menjadi dua bagian yaitu dikenali atau tidak dikenali setelah itu kemudian dilakukan perbandingan dengan pola yang sebelumnya disimpan di dalam database [19]. Dalam proses pengenalan wajah, citra dapat diambil dari jarak jauh tanpa menyentuh orang yang sedang diidentifikasi. Pengenalan wajah

memiliki tiga tahap a) lokasi deteksi wajah, b) ekstraksi fitur, c) klasifikasi citra wajah [20].

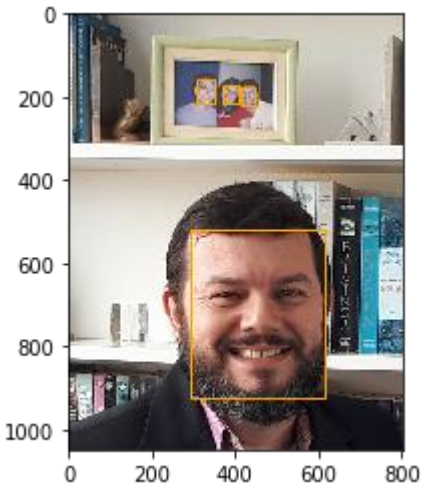
Ada banyak model yang dapat digunakan untuk pengenalan wajah. Dalam penelitian ini akan digunakan *Multi-task Cascaded Convolutional Neural Network* (MTCNN) dan modul DNN OpenCV.

2.5.1 *Multi-task Cascaded Convolutional Neural Network* (MTCNN)

Multi-task Cascaded Convolutional Neural Network (MTCNN) merupakan sebuah model pengenalan wajah yang diperkenalkan oleh Kaipeng Zhang dkk pada tahun 2016 dalam tulisan "*Joint Face Detection and Alignment Using Multi-task Cascaded Convolutional Ntworks*" [21]. Dalam model ini tidak hanya untuk mendeteksi wajah tetapi dapat juga mendeteksi objek lainnya.

Pada susunan strukturnya, MTCNN ini menggunakan struktur *cascade* dengan tiga tahap CNN yaitu pertama menggunakan jaringan konvolusional penuh untuk mendapatkan kotak dan vektor regresi untuk kotak pembatasnya. Untuk kotak yang sangat tumpang tindih akan ditumpang menggunakan *On-Maximum Suppression* (NMS). Selanjutnya kotak akan diteruskan ke CNN lain yang menolak sejumlah besar positif palsu dan melakukan kalibrasi kotak pembatas. Kemudian pada tahapan terakhir penandaan hasil deteksi dilakukan pada bagian wajah.

Berikut merupakan contoh deteksi dengan menggunakan MTCNN ditampilkan pada Gambar 2.11.

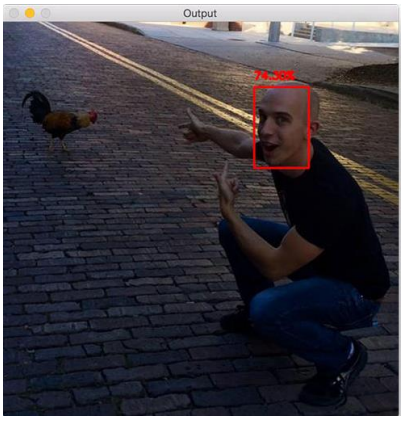


Gambar 2.11 Deteksi dengan MTCNN

2.5.2 Modul DNN OpenCV

Dalam modul *Deep Neural Network* (DNN) OpenCV ini adalah *model caffe* yang didasarkan pada *Single Shot-Multibox Detector* (SSD) dan menggunakan arsitektur *ResNet-10* sebagai tulang punggungnya [22]. Itu diperkenalkan pasca OpenCV 3.3 dalam modul jaringan saraf dalam. Dengan OpenCV dapat dilakukan deteksi pengenalan wajah dengan menggunakan model pendeteksi wajah *deep learning* yang terlatih. Untuk menggunakan modul DNN OpenCV dengan *model caffe* maka dibutuhkan *prototxt* yang mendefinisikan arsitektur model *ResNet10* dan *caffemodel* yang merupakan file yang berisi bobot untuk lapisan sebenarnya.

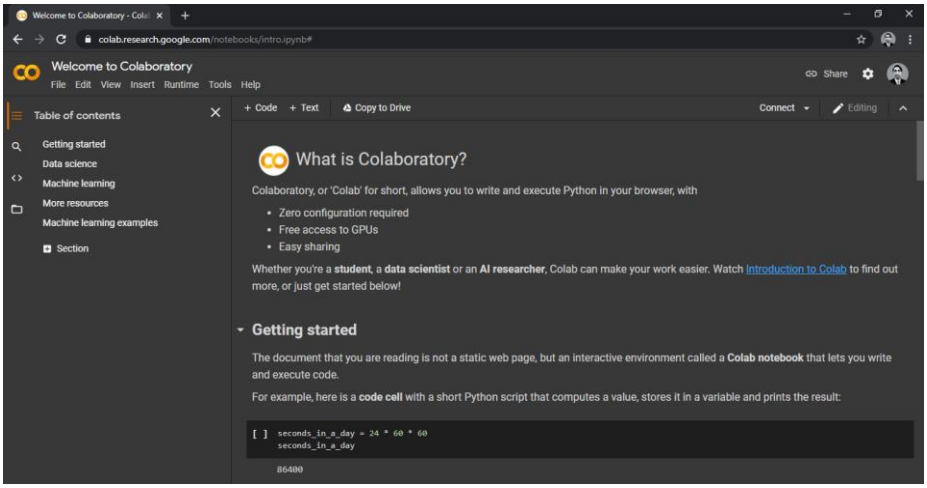
Berikut merupakan contoh deteksi dengan menggunakan MTCNN ditampilkan pada Gambar 2.12



Gambar 2.12 Deteksi dengan DNN OpenCV

2.6 *Google Colaboratory*

Google Colaboratory atau biasa disebut *Google Colab* merupakan salah satu produk *Google* berbasis *cloud* yang dapat digunakan secara gratis. Perbedaanya dengan produk *Google* lainnya yaitu pada *Google Colab* biasa diaplikasikan sebagai *coding environment* bahasa pemrograman *Python* dengan format "*notebook*" yang mirip dengan *Jupyter Notebook* [23]. Dengan kata lain, *Google* seakan memberikan fasilitas untuk para programmer atau *researcher* untuk menggunakan komputer gratis dengan spesifikasi yang tinggi. Tampilan antarmuka dari *Google Colab* ditunjukkan pada Gambar 2.13.



Gambar 2.13 Tampilan awal *Google Colaboratory*

Selain sangat praktis digunakan, *Google Colab* memiliki banyak manfaat. Berikut beberapa kelebihan yang dimiliki oleh *Google Colab*.

1. GPU Gratis. *Google Colab* memberikan komputer dengan spesifikasi tinggi berupa *GPU Tesla*, *Ram* ~12GB, *Disk* ~300GB yang masih bisa disambungkan dengan *Google Drive*, dan akses internet cepat untuk mengunduh data besar.
2. Kolaborasi. *Google Colab* memudahkan pengguna untuk melakukan kolaborasi dengan orang lain karena dapat membagi kodingan secara daring seperti *Google Doc*.
3. Mudah berintegrasi. *Google Colab* sangatl fleksibel dalam hal ini karena dapat menghubungkan dengan *Jupyter Notebook*, *Google Drive*, atau dengan *Github*.
4. Fleksibel. Salah satu yang paling menarik yaitu dapat mudah diakses menggunakan ponsel karena esensinya *Google Colab* hanya perlu dijalankan pada *browser*.

2.7 *Framework TensorFlow* dan *Keras*

Saat ini cukup banyak *framework* yang dapat digunakan untuk pengguna *deep learning*, antara lain ada *framework TensorFlow* dan *Keras*. Kedua *framework* tersebut merupakan yang paling banyak digunakan dalam dunia *deep learning*.

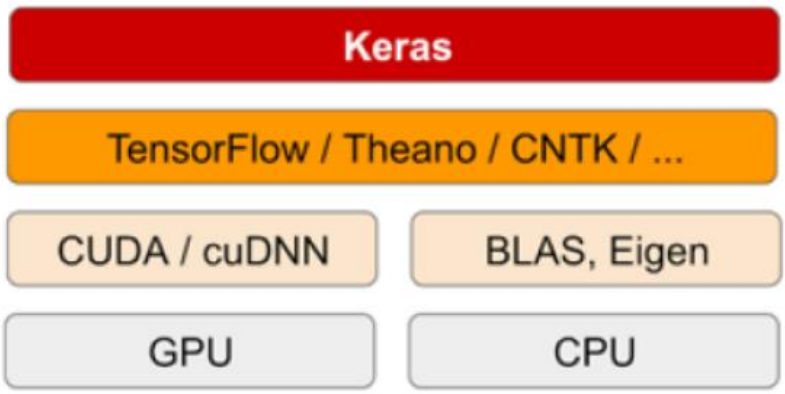
TensorFlow merupakan sebuah *platform* sumber terbuka untuk *deep learning* [24]. Ini adalah sebuah *library* dengan API level tinggi. Dengan begitu *TensorFlow* dapat membantu dalam pembuatan *neural network* dalam skala yang besar. Sedangkan *Keras* adalah sebuah *framework deep learning* untuk *python* yang menyediakan cara mudah untuk mendefinisikan dan melatih hampir semua jenis model *deep learning* [25]. *Keras* awalnya dikembangkan dengan tujuan untuk

mempercepat kemungkinan bereksperimen. Ada beberapa fitur utama pada *Keras* antara lain sebagai berikut.

a. *Keras* memungkinkan kode yang berjalan mulu di CPU atau GPU.
b. Memiliki API yang *user-friendy* yang membuat *Keras* menjadi mudah untuk cepat prototipe model *deep learning.*
c. Memiliki dukungan bawaan untuk *convolutional networks*, *recurrent network*, dan kombinasi keduanya.
d. Mendukung arsitektur jaringan yang diinginkan: model *multi-input* atau *multi-output*, berbagai lapisan, berbagai model dan sebagainya.

Susunan dari perangkat lunak dan perangkat keras dari *deep learning*, dapat dilihat

Gambar *2.14* [25].



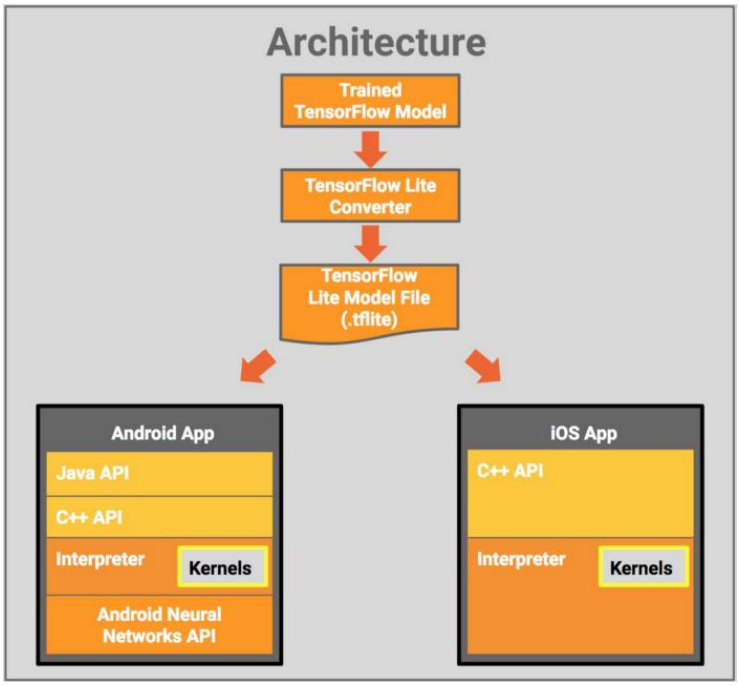
Gambar 2.14 Susunan Perangkat Lunak dan Perangkat Keras *Deep Learning*

Melalui *TensorFlow*, *Keras* dapat berjalan dengan mulus di CPU dan GPU. Ketika berjalan pada CPU, *TensorFlow* sendiri akan melibatkan *library* tingkat rendah untuk operasi *tensor* yang disebut dengan *Eigen*. Sedangkan pada GPU, *TensorFlow* akan melibatkan *library* operasi *deep learning* yang dioptimalkan

dengan baik menggunakan cuDNN atau biasa disebut dengan *library NVIDIA CUDA Deep Neural Network*.

2.8 *TensorFlow Lite*

TensorFlow Lite adalah seperangkat alat untuk membantu *developer* menjalankan *TensorFlow* di perangkat seluler, tersemat, dan *Internet of Things* (IoT) [26]. Ini memungkinkan inferensi *machine learning* di perangkat dengan latensi rendah dan ukuran biner. Selain itu juga ketika menggunakan *TensorFlow Lite* dan diaplikasi pada suatu perangkat maka dapat sangat membantu dalam konektivitas karena tidak lagi diperlukan koneksi internet. Sehingga akan memudah untuk dimanfaatkan melakukan klasifikasi, regresi, deteksi, atau apa pun yang diinginkan tanpa harus melakukan akses pengiriman dan penerimaan dari atau ke server. Arsitektur *TensorFlow Lite* itu sendiri ditunjukkan pada Gambar 2.15 [27].



Gambar 2.15 Arsitektur *TensorFlow Lite*

Saat ini *TensorFlow Lite* didukung pada *Android* dan *iOS* melalui C++ *Application Program Interface* (API) seta memiliki *Java Wrapper* untuk pengembangan *Android*. Selain itu, pada fitur utamanya terdapat penerjemah yang disesuaikan untuk machine learning di perangkat *Android* maupun *iOS*.

TensorFlow Lite bukan dirancang untuk model pelatihan, melainkan untuk melatih modelnya dilakukan pada mesin spesifikasi lebih tinggi dan kemudian dilakukan penyimpanan dengan melakukan konversi ke dalam format ".tflite". Selanjutnya model dengan format tersebut digunakan atau dimuat ke dalam penerjemah seluler (*mobile interpreter*).

2.9 *Android Studio*

Android Studio adalah lingkungan pengembangan terpadu (*Integrated Development Environment/IDE*) resmi untuk pengembangan aplikasi *Android* yang didasarkan pada *IntelliJ IDEA* [28]. Selain sebagai editor kode dan fitur *developer IntelliJ* yang andal, Android Studio juga menawarkan banyak fitur yang meningkatkan produktivitas dalam membuat aplikasi *Android*, seperti:

a. sistem *build* berbasis *Gradle* yang fleksibel,
b. emulator yang cepat dan kaya fitur,
c. lingkungan yang terpadu dimana bisa digunakan untuk mengembangkan aplikasi pada semua perangkat android,
d. menerapkan perubahan untuk melakukan *push* pada perubahan kode dan *resource* ke aplikasi yang sedang berjalan tanpa memulai ulang aplikasi,
e. *code templates* dan integrasi *GitHub* untuk membuat fitur aplikasi umum dan mengimpor kode sampel,

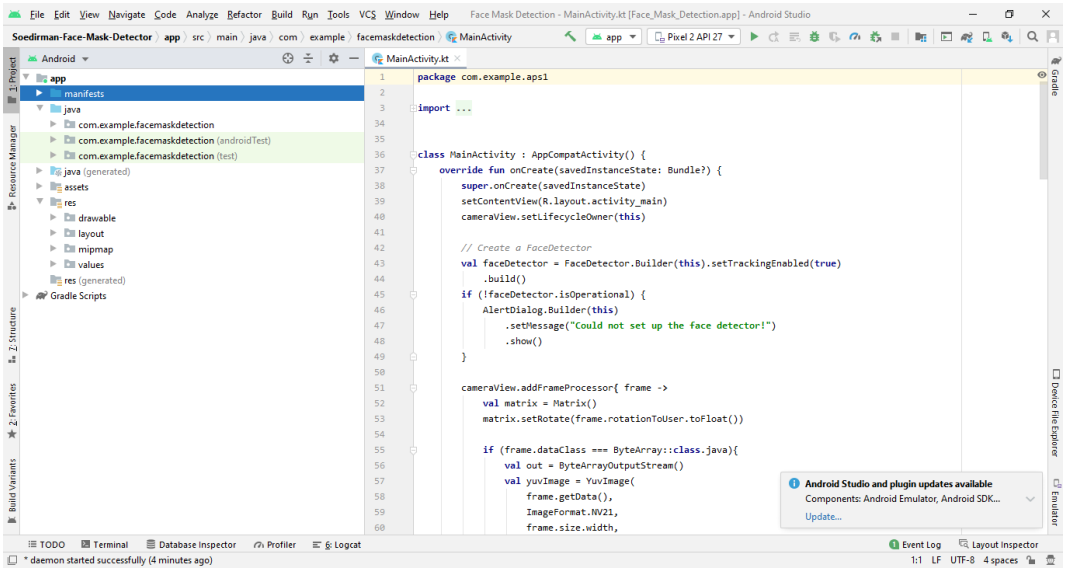
f. *framework* dan fitur pengujiannya lengkap,

g. *lint tools* untuk menangkap kinerja, kegunaan, kompabilitas versi dan masalah lainnya,

h. dukungan C++ dan NDK, serta

i. dukungan bawaan untuk *Google Cloud Platform*, membuatnya mudah untuk mengintegrasikan *Google Cloud Messaging* dan *App Engine*.

Android Studio sangat membantu *developer* dalam membangun sebuah aplikasi *android*. Ketika melakukan koding, *android studio* menawarkan beberapa *live hints* atau petunjuk langsung sehingga ini akan sangat membantu, selain itu mereka juga akan memberikan *auto correct*. Tampilan antarmuka dari *android studio* dapat dilihat pada Gambar 2.16.



Gambar 2.16 Tampilan Utama *Android Studio*

BAB 3
METODE PENELITIAN

3.1 Waktu dan Tempat

Penelitian dilaksanakan dalam waktu 4 bulan pada bulan Oktober 2020 sampai bulan Januari 2021 di Kampus Fakultas Teknik Universitas Jenderal Soedirman yang terletak di Km 5, Jalan Mayor Jenderal Sungkono, Desa Blater, Kabupaten Purbalingga, Jawa Tengah.

3.2 Alat dan Bahan

Dalam penelitian ini, daftar alat dan bahan yang akan digunakan selama penelitian sebagai berikut.

1. Perangkat keras:
    a. Laptop *Asus Vivobook* A407MA dengan spesifikasi *Intel Celeron* N4000 dan RAM 4 GB.
    b. Komputer virtual *Google Colaboratory* dengan spesifikasi GPU Nvidia Tesla T4, RAM 13 GB, Disk: ~38 GB.
    c. *Smartphone Android* Mi A1 dengan spesifikasi prosesor *Octa-core Snapdragon* 625, maks 2,0 GHz, memori internal 64 GB dan RAM 4 GB.
2. Perangkat lunak
    a. Sistem operasi Windows 10 64 bit
    b. Peramban Internet *Google Chrome* versi 82.0.4240.111 64 bit
    c. *Google Colaboratory* (*Jupyter Notebook* versi *cloud*)
    d. *Andorid Studio* versi 4.1

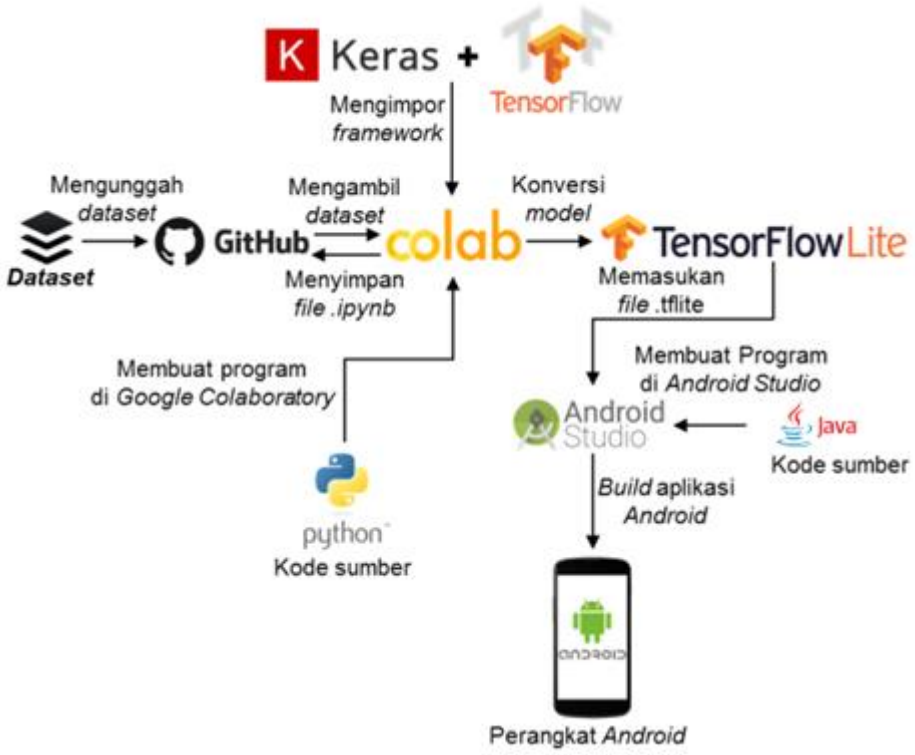
e. Sistem Operasi *Android Pie* 9.0

f. Layanan Repositori *Web Development* pada Platform *Github*

3. *Dataset* pelatihan dan pengujian berupa gambar orang menggunakan masker dan tidak menggunakan masker didapatkan dari github/chandrikadeb7 "*Face Mask Detection*".

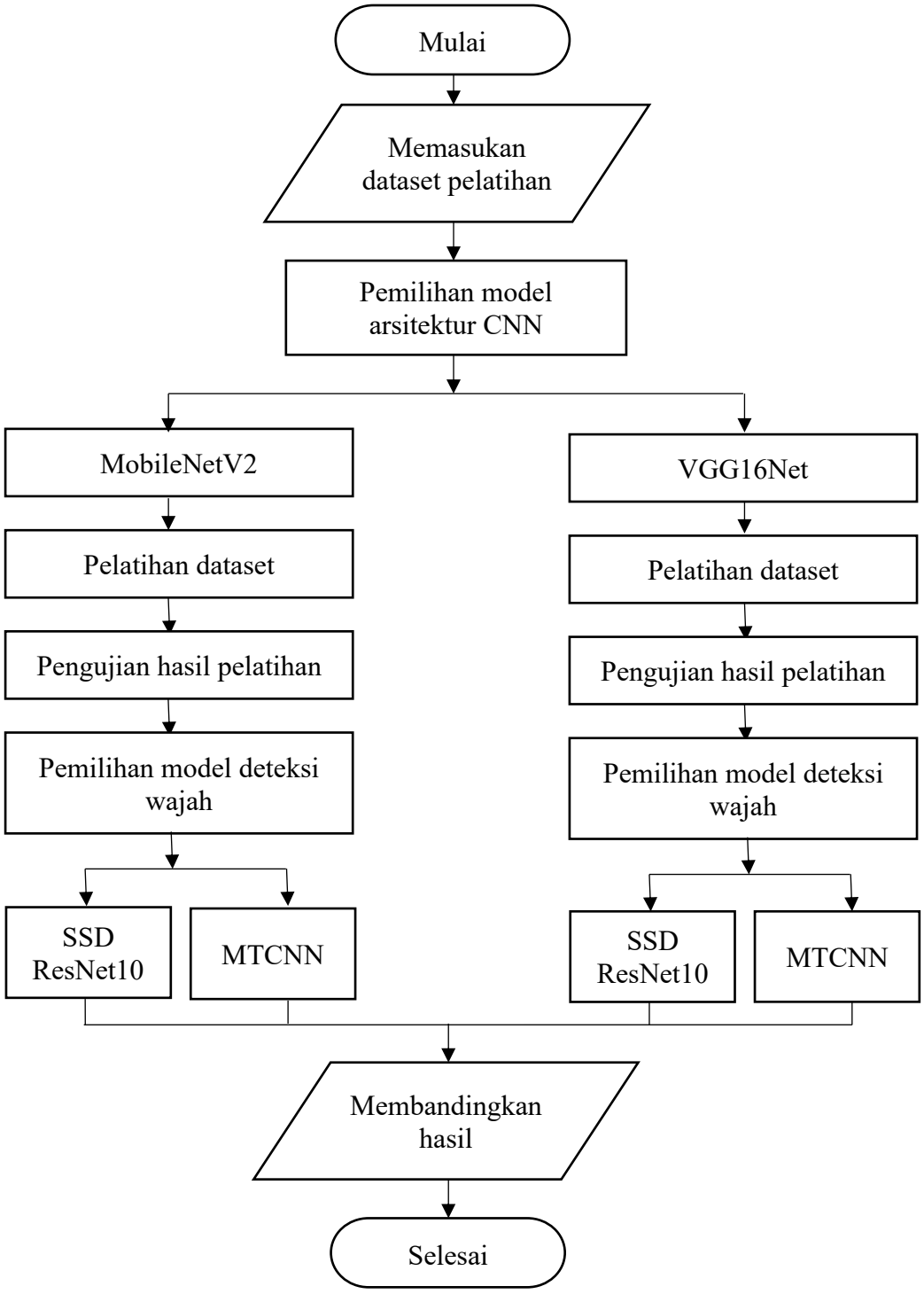
3.3 Alur dan Tahap Penelitian

Penelitian dilaksanakan melalui beberapa tahapan yaitu tahap persiapan dengan studi litelatur, tahap persiapan data dan *pre*-proses *dataset*, tahap desain arsitektur, tahap pengujian dan tahap evaluasi. Arsitektur sistem yang akan dibuat disajikan pada Gambar 3.1 berikut.



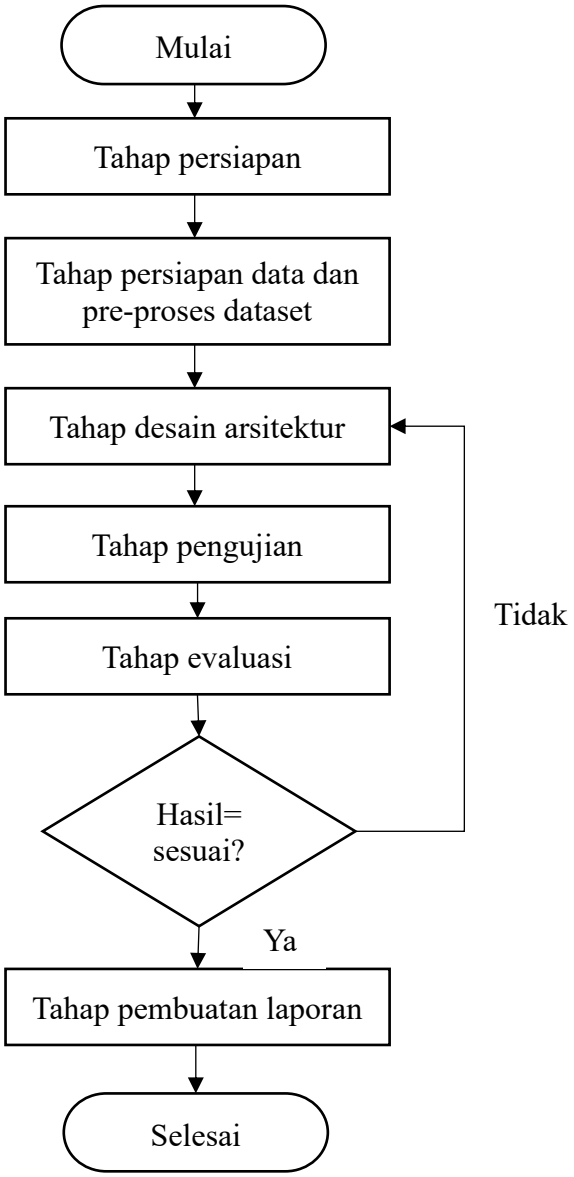
Gambar 3.1 Desain Arsitektur Sistem

Sedangkan untuk diagram alir sistem penelitian yang akan dilakukan yaitu seperti yang disajikan pada *Gambar 3.2*.



Gambar 3.2 Diagram Alir Sistem

Dalam penelitian ada beberapa tahapan-tahapan yang akan dilakukan yaitu seperti yang disajikan pada Gambar 3.3.



Gambar 3.3 Alur Penelitian

3.3.1 Tahap Persiapan

Tahap pertama yaitu persiapan, dilakukan pembuatan pra-proposal, merumuskan masalah, dan mengidentifikasi masalah serta memahami permasalahan yang akan diangkat menjadi topik. Untuk mendukung kegiatan

persiapan maka harus dilakukan sebuah studi pustaka seperti mengumpulkan referensi baik berupa jurnal, buku, artikel maupun referensi lain yang berkaitan dengan penelitian terdahulu.

3.3.2 Tahap Persiapan Data dan *Pre*-Proses *Dataset*

Pada tahap ini menyiapkan data yang dibutuhkan dan melakukan *pre*-proses *dataset* dengan cara mengumpulkan gambar orang yang menggunakan masker dan tidak menggunakan masker dan mempersiapkan model untuk deteksi wajah seperti *SSD ResNet10* dan *MTCNN*. *Dataset* pelatihan menggunakan *dataset* proyek chandrikadeb7 dengan judul "*Face Mask Detection*" pada *Github*. Selanjutnya *dataset* tersebut dimasukkan ke *Google Colaboratory*.

3.3.3 Tahap Desain Arsitektur

Proses awal desain arsitektur dimulai dari membuat kode sumber untuk program *Convolutional Neural Network* (CNN) dan mengimpor *Framework Keras* dan *TensorFlow* yang dibuat menggunakan infrastruktur *Google Colaboratory* dengan bahasa pemrograman *Python* dan disimpan dalam bentuk *file Jupyter Notebooks* ".ipynb" dan kemudian akan disimpan ke *Github*. Selanjutnya *dataset* pelatihan akan diambil dan disimpan ke dalam tempat penyimpanan sementara pada *Google Colaboratory* dari *Github*.

Sistem yang dirancang menggunakan arsitektur jaringan CNN yang bernama *MobileNetV2* dan *VGG16Net* dengan cara membuat model jaringan CNN yang sudah dipelajari sebelumnya (*pre-trained convnets*) pada *base model* dan *file* tersebut akan dilakukan *feature extraction*. Selanjutnya akan ditambahkan model arsitektur tambahan pada *head model*.

3.3.4 Tahap Pengujian

Ketika pembuatan arsitektur sudah terlaksanakan, maka dilakukan tahap pengujian. Seluruh dataset pelatihan dilatih terhadap masing-masing arsitektur jaringan CNN yang dibuat dengan panjang *epochs* atau iterasi yang ditentukan, misalnya 25 iterasi atau bahkan lebih. Setelah mencapai hasil prediksi yang memuaskan, akan dilakukan pengujian pada *Google Colabolatory* dan aplikasi *Android*. Pengujian yang dilakukan pada *Google Colabolatory* dengan menggunakan bantuan pendeteksi wajah seperti *SSD ResNet10* dan *MTCNN* sedangkan pada *Android* dengan mengkonversi hasil model pelatihan ke dalam format *TensorFlow Lite ".tflite"* untuk selanjutnya digunakan pada *Android Studio*. File *".tflite"* tersebut akan dimasukan dalam folder "*assets*" pada direktori *project Android Studio*.

Ukuran pengujian yaitu dilakukan pada citra berupa akurasi, presisi, *recall*, nilai F1 atau *harmonic mean* dari presisi dan *recall*, serta pengujian mAP (*mean Average Precision*). Sedangkan untuk video berupa pengujian *Frame Per Second* (FPS) atau berapa banyak gambar yang dihasilkan setiap detiknya. Selain itu terdapat beberapa variasi pengujian yang akan dilakukan seperti perbedaan jarak, tingkat pencahayaan, dan jenis masker muka yang digunakan.

3.3.5 Tahap Evaluasi

Tahap evaluasi dilakukan untuk memperbaiki sistem jika terjadi kesalahan pada sistem aplikasi tersebut. Sehingga akan dihasilkan aplikasi yang baik dan layak digunakan untuk banyak orang. Jika tidak terjadi kesalahan dan aplikasi mudah digunakan oleh banyak orang, maka dapat disimpulkan aplikasi layak untuk digunakan.

3.4 Jadwal Penelitian

Penelitian dilaksanakan dalam waktu 4 bulan pada bulan Oktober 2020 sampai bulan Januari 2021 dengan rincian jadwal kegiatan sebagai berikut.

Tabel 3.1 Rincian Jadwal Penelitian

No.	Kegiatan	Oktober				November				Desember				Januari			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1.	Studi pustaka																
2.	Persiapan dan pre-processing																
3.	Desain asitektur																
4.	Pengujian sistem																
5.	Evaluasi sistem																
6.	Pembuatan laporan																

DAFTAR PUSTAKA

[1] Kemenkes RI dan Direktorat Jenderal P2P, *Pedoman Pencegahan dan Pengendalian Coronavirus Disease (COVID-19) Rev Ke - 4*, vol. 4. 2020.

[2] Kemenkes RI, *Pedoman Pencegahan dan Pengendalian Coronavirus Disease (COVID- 19) Rev Ke - 5*, vol. 5. 2020.

[3] Kemenkes RI, "QnA : Pertanyaan dan Jawaban Terkait COVID-19," 2020. [Online]. Available: https://covid19.kemkes.go.id/qna-pertanyaan-dan-jawaban-terkait-covid-19/#.X6O9pYgzbIU. [Accessed: 05-Nov-2020].

[4] M. Mohammed, M. B. Khan, and E. B. M. Bashie, *Machine learning: Algorithms and Applications*, no. July. 2016.

[5] Learners Evolve Machine, "Mengenal Machine Learning," *medium.com*, 2019. [Online]. Available: https://medium.com/evolve-machine-learners/mengenal-machine-learning-6c4a48db48b0. [Accessed: 06-Nov-2020].

[6] W. S. Eka Putra, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101," *J. Tek. ITS*, vol. 5, no. 1, 2016, doi: 10.12962/j23373539.v5i1.15696.

[7] S. O'Dea, "Number of smartphone users from 2016 to 2021," *statista*, 2020. [Online]. Available: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/. [Accessed: 06-Nov-2020].

[8] M. M. Lambacing and F. Ferdiansyah, "Rancang Bangun New Normal

Covid-19 Masker Detektor Dengan Notifikasi Telegram Berbasis Internet of Things," *Dinamik*, vol. 25, no. 2, pp. 77–84, 2020, doi: 10.35315/dinamik.v25i2.8070.

[9] M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa, "A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic," *Meas. J. Int. Meas. Confed.*, vol. 167, no. May 2020, p. 108288, 2021, doi: 10.1016/j.measurement.2020.108288.

[10] V. F. Hidarlan, "Rancang Bangun Klasifikasi Varietas Beras Berdasarkan Citra Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Android," pp. 1–17, 2020.

[11] Wahyudi AMK, "PENTINGNYA MENGGUNAKAN MASKER DIMASA PANDEMI," *RS Harapan Ibu*, 2020. [Online]. Available: https://www.rsuharapanibu.co.id/pentingnya-menggunakan-masker-dimasa-pandemi/. [Accessed: 10-Nov-2020].

[12] W. Dadang, "Memahami Kecerdasan Buatan berupa Deep Learning dan Machine Learning," *Warung Sains Teknologi*, 2018. [Online]. Available: https://warstek.com/2018/02/06/deepmachinelearning/.

[13] J. W. G. Putra, *Pengenalan Konsep Pembelajaran Mesin dan Deep Learning*, vol. 1.4. 2020.

[14] M. Zufar and B. Setiyono, "Convolutional Neural Networks Untuk Pengenalan Wajah Secara Real-Time," *J. Sains dan Seni ITS*, vol. 5, no. 2, p. 128862, 2016, doi: 10.12962/j23373520.v5i2.18854.

[15] Sofyan, “Pengenalan Convolutional Neural Network – Part 1,” 2019. [Online]. Available: http://sofyantandungan.com/pengenalan-convolutional-neural-network-part-1/. [Accessed: 09-Nov-2020].

[16] R. O. Ekoputris, “MobileNet: Deteksi Objek pada Platform Mobile,” *medium.com*, 2018. [Online]. Available: https://medium.com/nodeflux/mobilenet-deteksi-objek-pada-platform-mobile-bbbf3806e4b3. [Accessed: 09-Nov-2020].

[17] M. Sandler, M. Zhu, A. Zhmoginov, and C. V Mar, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” 2019.

[18] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[19] B. Arifin, “Aplikasi Sensor Passive Infra Red (PIR) Untuk Pendeteksian Makhluk Hidup Dalam Ruang,” *Pros. SNST ke-4*, no. 2011, pp. 39–44, 2013.

[20] Derisma, “Faktor-Faktor yang Mempengaruhi Sistem Pengenalan Wajah Menggunakan Metode Eigenface pada Perangkat Mobile Berbasis Android,” *J. Politek. Caltex Riau*, vol. 2, no. 2, pp. 127–136, 2016.

[21] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks,” *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, 2016, doi: 10.1109/LSP.2016.2603342.

[22] V. Agarwal, “Face Detection Models: Which to Use and Why?,”

medium.com2, 2020. [Online]. Available: https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c. [Accessed: 11-Dec-2020].

[23] R. Adam, "Mengenal Google Colab," *Structimly*, 2019. [Online]. Available: https://structilmy.com/2019/05/mengenal-google-colab/. [Accessed: 09-Nov-2020].

[24] A. Choudhury, "TensorFlow vs Keras: Which One Should You Choose," 2019. [Online]. Available: https://analyticsindiamag.com/tensorflow-vs-keras-which-one-should-you-choose/. [Accessed: 09-Nov-2020].

[25] F. Chollet and J. J. Allaire, *Deep Learning with R*, 1st editio. United States of America: Manning Publications, 2018.

[26] Creative Commons Attribution 4.0, "Panduan TensorFlow Lite," *Google*, 2020. [Online]. Available: https://www.tensorflow.org/lite/guide. [Accessed: 10-Nov-2020].

[27] L. Moroney, "Using TensorFlow Lite on Android," *medium.com*, 2018. [Online]. Available: https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d. [Accessed: 10-Nov-2020].

[28] Developers Android, "Mengenal Android Studio," *Google Developer*, 2020. [Online]. Available: https://developer.android.com/studio/intro?hl=id. [Accessed: 10-Nov-2020].