Deteksi Masker Wajah dengan Metode Convolutional Neural Network

Ivan Hartono, Agustinus Noertjahyana, Leo Willyanto Santoso
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
m26416078@john.petra.ac.id, agustinus@petra.ac.id, leow@petra.ac.id

ABSTRAK

Komputer harus mampu mengenali area yang merupakan objek wajah pada citra agar dapat mempermudah deteksi masker wajah yang digunakan manusia. Deep learning adalah kecerdasan buatan dengan representasi sederhana yang memiliki lapisan tersembunyi untuk memproses data yang dapat membangun konsep yang kompleks. Deep learning dapat dilatih untuk mendeteksi suatu objek serta klasifikasi objek. Terdapat banyak algoritma deep learning yang dapat digunakan untuk proses pengenalan model misalnya untuk klasifikasi objek menggunakan MobileNet, VGGNet, DenseNet, GoogLeNet, AlexNet, dan lain-lain sedangkan untuk deteksi objek dapat menggunakan You Only Look Once, SSD Resnet, Multi-task Cascaded Convolutional Neural Network (MTCNN), HyperFace dan lain-lain. Sistem pendeteksi objek dapat menggunakan dua kombinasi algoritma yaitu melakukan dengan metode klasifikasi objek dan metode deteksi objek. Metode untuk pengenalan objek masker pada wajah manusia adalah CNN (Convolutional Neural Network).

Metode CNN adalah pengembangan dari Multilayer Perceptron yang didesain untuk mengolah data dua dimensi. Metode CNN yang sangat baik dalam memproses data spasial dan mengklasifikasi objek [1]. Setelah pelatihan model dengan VGGNet dilakukan maka metode selanjutnya adalah mendeteksi suatu objek dengan menggunakan modul SSD ResNet.

Kata Kunci: *Convolutional Neural Network, SSD Resnet, VGGNet*

ABSTRACT

The computer must be able to recognize the area which is a face object in the image in order to facilitate the detection of face masks used by humans. Deep learning is artificial intelligence with simple representations that have hidden layers to process data that can build complex concepts. Deep learning can be trained to detect an object and classify objects. There are many deep learning algorithms that can be used for the model recognition process, for example for object classification using MobileNet, VGGNet, DenseNet, GoogLeNet, AlexNet, and others while for object detection you can use You Only Look Once, SSD Resnet, Multi-task Cascaded Convolutional Neural Network (MTCNN), HyperFace and others. The object detection system can use two combinations of algorithms, namely the object classification method and the object detection method. The method for recognizing mask objects on human faces is CNN (Convolutional Neural Network).

The CNN method is the development of the Multilayer Perceptron which is designed to process two-dimensional data. CNN method is very good in processing spatial data and classifying objects [1]. After training the model with VGGNet, the next method is to detect an object using the SSD ResNet module.

Keywords: *Convolutional Neural Network. SSD Resnet, VGGNet*

1. PENDAHULUAN

1.1 Latar Belakang Masalah

Tahun 2019 di negara China telah ditemukan kasus awal terdeteksinya COVID-19, penyakit saluran pernapasan dan spektrum infeksi virus. Tedros (2020) menjelaskan COVID-19 yaitu singkatan dari 'Co' yang artinya 'corona', 'Vi' untuk 'Virus', dan "D" untuk 'Penyakit (disease)'. Virus corona ini perlahan menyebar ke seluruh dunia dan *World Health Organization* (WHO) yang merupakan badan khusus Perserikatan Bangsa-Bangsa (PBB) dalam bidang kesehatan menetapkan wabah COVID-19 sebagai pandemi. Artinya COVID-19 sudah menyebar secara global di seluruh dunia. Angka kasus COVID-19 terus meningkat dan masyarakat dihimbau untuk tetap berada di dalam rumah untuk memutus rantai penyebaran COVID-19. Namun, pada kondisi tertentu kita tetap harus keluar rumah untuk melakukan aktivitas tertentu. Agar tetap aman saat harus pergi beraktivitas keluar rumah, Kementerian Kesehatan membuat sebuah protokol kesehatan sebagai solusinya yang tertulis pada Keputusan Menteri Kesehatan Republik Indonesia Nomor HK.01.07/MENKES/382/2020 tentang Protokol Kesehatan bagi Masyarakat di Tempat dan Fasilitas Umum dalam Rangka Pencegahan dan Pengendalian Corona Virus Disease 2019 (COVID-19). Aturan tersebut perlu dilakukan di tempat atau fasilitas umum seperti pasar, pusat perbelanjaan, penginapan, rumah makan, sarana kegiatan olahraga, transportasi umum, lokasi daya tarik wisata, jasa perawatan kecantikan, jasa ekonomi kreatif, tempat ibadah, dan jasa penyelenggaraan pertemuan. Pada setiap lokasi tersebut, orang harus memastikan diri dalam kondisi sehat sebelum keluar rumah, menerapkan prinsip jaga jarak 2 meter dari orang lain, membawa alat pribadi, dan wajib menggunakan masker.

Penggunaan masker merupakan bagian dari rangkaian komprehensif langkah pencegahan dan pengendalian yang dapat membatasi penyebaran penyakit – penyakit virus saluran pernapasan tertentu, termasuk COVID-19. Masker dapat digunakan baik untuk melindungi orang yang sehat (dipakai untuk melindungi diri sendiri saat berkontak dengan orang yang terinfeksi) atau untuk mengendalikan sumber (dipakai oleh orang yang terinfeksi untuk mencegah penularan lebih lanjut).

Adanya protokol kesehatan yang ditetapkan oleh pemerintah membuat tugas baru untuk petugas keamanan fasilitas umum dalam memeriksa apakah masyarakat memakai masker dengan benar. Permasalahan yang didapat adalah pengenalan objek pada video yang digunakan untuk pengenalan masker. Deteksi objek pada gambar bergerak dapat digunakan pula pada kamera pengintai di tempat umum oleh karena itu penelitian ini lebih diarahkan pada gambar bergerak atau video. Penelitian ini juga diarahkan pada multi deteksi yang dilakukan pada orang yang berjalan di fasilitas umum.

Berkaitan dengan sistem keamanan untuk orang yang tidak memakai masker, fasilitas umum biasanya membutuhkan bantuan nasihat dari petugas keamanan fasilitas umum. Otomatisasi sangat diinginkan untuk memantau orang yang tidak memakai masker dan juga mengurangi jumlah manusia sebagai sumber daya yang dibutuhkan. Solusi yang tepat untuk menghadapi masalah ini adalah menggunakan *object detection* pada video kamera pengintai di tempat umum.

Komputer harus mampu mengenali area yang merupakan objek wajah pada citra agar dapat mempermudah deteksi masker wajah yang digunakan manusia. *Deep learning* adalah kecerdasan buatan dengan representasi sederhana yang memiliki lapisan tersembunyi untuk memproses data yang dapat membangun konsep yang kompleks. *Deep learning* dapat dilatih untuk mendeteksi suatu objek serta klasifikasi objek. Terdapat banyak algoritma *deep learning* yang dapat digunakan untuk proses pengenalan model misalnya untuk klasifikasi objek menggunakan *MobileNet, VGGNet, DenseNet, GoogLeNet, AlexNet*, dan lain-lain sedangkan untuk deteksi objek dapat menggunakan *You Only Look Once, SSD Resnet*, Multi-task Cascaded Convolutional Neural Network (MTCNN), HyperFace dan lain-lain. Sistem pendeteksi objek dapat menggunakan dua kombinasi algoritma yaitu melakukan dengan metode klasifikasi objek dan metode deteksi objek. Metode untuk pengenalan objek masker pada wajah manusia adalah CNN (*Convolutional Neural Network*). Metode CNN adalah pengembangan dari *Multilayer Perceptron* yang didesain untuk mengolah data dua dimensi. Metode CNN yang sangat baik dalam memproses data spasial dan mengklasifikasi objek [1]. Setelah pelatihan model dengan *VGGNet* dilakukan maka metode selanjutnya adalah mendeteksi suatu objek dengan menggunakan modul *SSD ResNet*.

Berdasarkan latar belakang, rumusan masalah yang dapat dibuat berhubungan dengan penelitian ini antara lain: Seberapa tinggi tingkat akurasi pelatihan mengenali *region* masker wajah menggunakan metode *VGGNet* dan seberapa tinggi tingkat akurasi mendeteksi apakah orang memakai masker wajah dengan benar menggunakan metode *SSD ResNet.*

Dalam penelitian ini terdapat pembatasan terhadap masalah yang akan dibahas, yaitu: input yang digunakan adalah foto dengan format ekstensi .jpg; Metode deep learning yang digunakan untuk pendeteksi masker muka terdiri dari dua komponen yaitu algoritma klasifikasi wajah dengan *Convolutional Neural Network* (*VGG16Net*) dan algoritma pendeteksi muka dengan *SSD Resnet10;* Program pendeteksi masker muka menggunakan bahasa pemrograman Python dengan *interface* dan infrastruktur Google Colaboratory; objek yang menjadi data *training* maupun *testing* tidak terpotong dan bersih dari *noise; D*ataset yang dipakai memiliki objek masker wajah; *Dataset* yang digunakan sebanyak 2.095 data gambar didapatkan dari github/chandrikadeb7 *"Face Mask Detection"*. Dataset tersebut selain digunakan untuk *testing*, juga digunakan untuk *training* pengenalan jenis masker. *Dataset* yang diperoleh akan dibagi menjadi 2. Pertama, *dataset* akan digunakan untuk *training* dan lalu yang kedua *dataset* digunakan untuk *testing*. Jumlah yang digunakan untuk *training* adalah sebesar 75% dari jumlah keseluruhan *dataset*, sedangkan untuk *testing* adalah 25% dari jumlah keseluruhan *dataset; p*rogram dibuat untuk membedakan pengguna memakai masker atau tidak memakai masker; Arsitektur sistem membahas permasalahan bagaimana data yang di-*input* diproses sehingga dapat menghasilkan *output* yang sesuai. Sistem akan menerima input berupa gambar lalu akan diterapkan metode *SSD ResNet10* untuk mendeteksi masker wajah, lalu untuk mendapatkan posisi wajah. Setelah itu, akan digunakan metode *VGG16Net* untuk mengklasifikasi pengguna masker.

2. LANDASAN TEORI

Pada bab ini dijelaskan mengenai semua teori yang digunakan untuk membuat sistem untuk mendeteksi masker, mulai dari metode algoritma pendeteksi muka dengan *SSD ResNet10*, dan *CNN (VGG16Net)*, serta tinjauan studi.

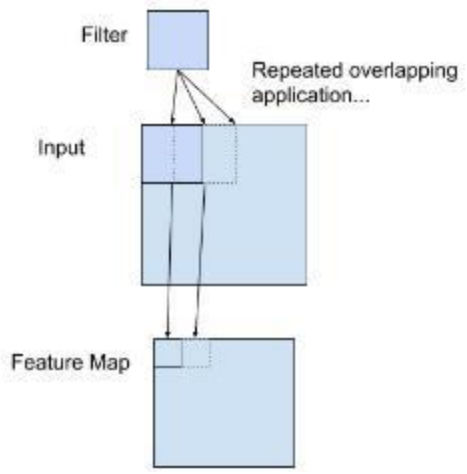
2.1 Single Shot-Multibox Detector (SSD) Resnet

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 [7]. 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.

2.2 Convolutional Neural Network

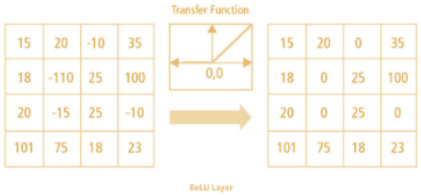
Convolutional Neural Network (CNN) adalah bagian dari *neural network* yang berspesialisasi dalam memproses data yang memiliki topologi *grid*, seperti gambar. Gambar mengandung serangkaian *pixel* yang tersusun dalam bentuk *grid* yang mengandung nilai *pixel* yang menunjukkan seberapa terang dan warna dari *pixel* tersebut.

CNN pada umumnya memiliki tiga *layer*, yaitu *convolutional layer*, *pooling layer*, *fully connected layer*. *Convolutional layer* adalah inti dari arsitektur CNN. Pada layer ini dilakukan proses konvolusi filter berupa matriks berukuran MXN (pada umumnya adalah 3 x 3) untuk mengekstrak fitur yang menonjol pada gambar. Fitur yang diambil ini mengandung hubungan yang khusus pada setiap *pixel*. Konvolusi dari gambar dengan konvolusi filter yang berbeda dapat melakukan performa seperti pendeteksian tepi, *blur*, dan juga penajaman gambar. Dalam *convolutional layer* ini selain memilih *filter*, juga ada *stride*. *Stride* adalah jumlah perpindahan *pixel* pada matriks *input*. Ketika *stride* bernilai 1 maka perpindahan *filter* yang dilakukan juga 1 pixel setiap waktu. Setelah *filter* dan *stride* ada juga *padding*. Kadang *filter* yang diaplikasikan pada *input* tidak sepenuhnya tepat, pada saat inilah *padding* dapat digunakan [5]. Contoh dari *convolutional layer* dapat dilihat pada Gambar 2.



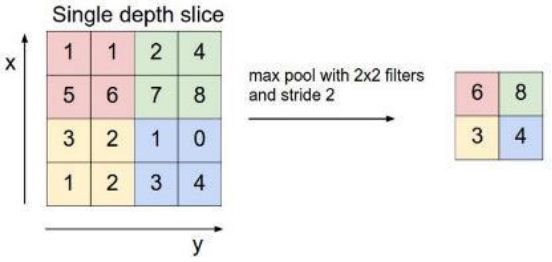
Gambar 1. *Convolutional Layer*

Pada *convolutional layer* ini juga dilakukan *activation function* pada setiap nilai dari *feature map*. *ReLU function* adalah *activation function* yang banyak digunakan di *neural network.* Tujuannya adalah mengubah semua *input* negatif menjadi nol [6]. Untuk ReLU *Layer* dapat dilihat pada Gambar 3.



Gambar 2. ReLU *Layer*

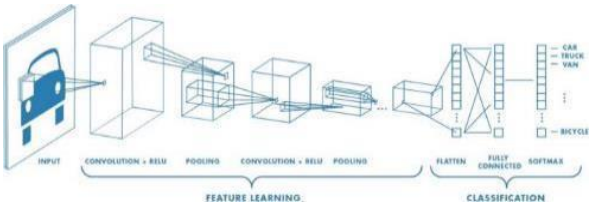
Setelah itu, pada *pooling layer* akan dilakukan pengecilan matriks, beberapa orang menyebutnya *reduce*. Dari setiap fitur yang dihasilkan dari *convolutional layer*, akan diambil nilai tertinggi pada setiap *window* untuk mengecilkan matriks (Ciaburro & Venkateswaran, 2017). Contoh dari pooling layer dapat dilihat pada Gambar 4.



Gambar 3. *Max P66ooling Layer*

Layer terakhir adalah *fully connected layer*, dimana matriks di-*flatten* menjadi vector. *Layer* ini menentukan klasifikasi dari *input* dengan menggunakan fungsi aktivasi seperti *softmax* atau *sigmoid. Layer* terakhir ini menentukan klasifikasi berdasarkan nilai kategori yang paling tinggi.

Secara garis besar, proses CNN dapat dilihat pada Gambar 5.



Gambar 4. *Convolutional Neural Network Process*

2.2.1 Arsitektur Jaringan VGG16Net

VGG16Net 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*" [2]. 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 6 menunjukan arsitektur dari VGG16.

Gambar 5. 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.

2.3 Tinjauan Studi

Berikut merupakan penelitian yang telah dilakukan sebelumnya:

2.5.1 *Detection of Motor Vehicles License Plateon Streaming Media with Convoutional Neural Network Algorithm Using Tensorflow*

Arsitektur jaringan yang digunakan untuk mendeteksi Tanda Nomor Kendaraan Bermotor pada media *streaming* terbagi menjadi beberapa *layer* yaitu *layer input, layer convolution, layer activation, layer pooling dan fully connected layer*. Dengan menggunakan 25.000 *step* dan *batch* sebanyak 8 pada proses pelatihan, akan menghasilkan model pelatihan deteksi Tanda Nomor Kendaraan Bermotor dengan tingkat akurasi yang tinggi. Tingkat akurasi pendeteksian Tanda Nomor Kendaraan bermotor pada media *streaming* menggunakan algoritma *Convolutional Neural Network* berkisar antara 70-100%. Hasil dari pendeteksian Tanda Nomor Kendaraan Bermotor pada media *streaming* menggunakan algoritma *Convolutional Neural Network* dapat dinilai dengan bekerja dengan baik. Banyaknya dataset dan beragamnya sudut pandang pada gambar pada proses pelatihan berpengaruh pada kecepatan dan nilai akurasi hasil model [4].

2.5.2 *A Hybrid Deep Transfer Learning Model with Machine Learning Methods For Face Mask Detection In The Era of The Covid-19 Pandemic*

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 [3].

2.5.3 Klasifikasi Citra Menggunakan *Convolutional Neural Network (CNN) pada Caltech 101*

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* [8]

3. ANALISA DAN DESAIN SISTEM

Pada bab ini dibahas mengenai tahap analisis data yang mencakup pengumpulan data dan pembagian data. Tahap Analisis Sistem mencakup Pengolahan Data dan Sistem Neural Network. Tahap Desain Sistem dimana mencakup Desain Web untuk Program.

3.1 Analisis Data

Analisis Data meninjau permasalahan yang berorientasi pada data, salat satunya adalah Pengumpulan Data. Data adalah bahan utama dalam penelitian ini. Data yang digunakan adalah data gambar berwarna (RGB) berformat JPG.

3.1.1 Pengumpulan Data

Dataset yang digunakan sebagai training sebanyak 2.095 data gambar didapatkan dari github/chandrikadeb7 "Face Mask Detection". Dataset tersebut selain digunakan untuk testing, juga digunakan untuk training pengenalan jenis masker. Dataset yang diperoleh akan dibagi menjadi 2. Pertama, dataset akan digunakan untuk training dan lalu yang kedua dataset digunakan untuk testing. Jumlah yang digunakan untuk training adalah sebesar 75% dari jumlah keseluruhan dataset, sedangkan untuk testing adalah 25% dari jumlah keseluruhan dataset. Gambar 8 adalah contoh dari data yang akan digunakan.

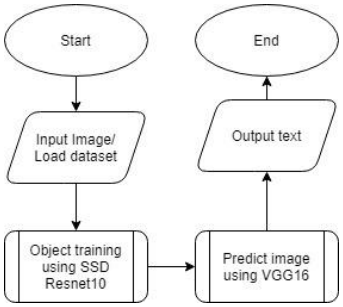
Gambar 6. Contoh Data yang Digunakan

3.1.2 Pembagian Data

Dataset yang diperoleh akan dibagi menjadi 2. Pertama, *dataset* akan digunakan untuk *training* dan lalu yang kedua *dataset* digunakan untuk *testing*. Jumlah yang digunakan untuk training adalah sebesar 75% dari jumlah keseluruhan *dataset*, sedangkan untuk *testing* adalah 25% dari jumlah keseluruhan dataset.

3.2 Analisa Sistem

Analisis Sistem membahas bagaimana data akan diproses. Sistem akan mengolah data berupa gambar menjadi sebuah *output* berupa teks hasil prediksi deteksi masker. Secara garis besar, sistem mencakup proses pengolahan data dan sistem *neural network.*

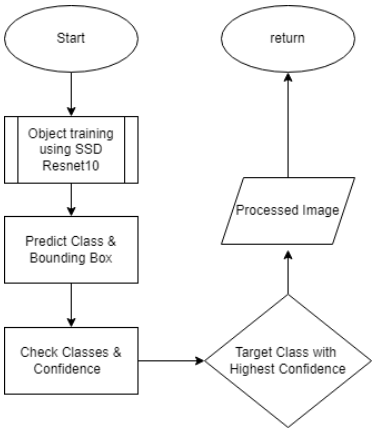


Gambar 7. Arsitektur Sistem

Sistem pertama-tama akan menerima *input* data dari *dataset* berupa *image* wajah manusia memakai masker dan tidak memakai masker dalam format .jpg. Pada *image* tersebut kemudian akan diterapkan *sistem* untuk mendeteksi lokasi wajah yang ada di dalamnya. Sistem untuk mendeteksi lokasi wajah ini dilakakukan menggunakan metode SSD Resnet10.

Setelah didapatkan gambar yang objeknya sudah diprediksi menggunakan SSD Resnet10, gambar tersebut akan dimasukkan ke dalam proses selanjutnya yaitu proses menggunakan CNN (*Convolutional Neural Network*) yang telah di-*train* sebelumnya agar dapat menghasilkan kondisi wajah dalam bentuk tipe data *string*. Gambar 9 adalah *flowchart* dari keseluruhan sistem.

3.2.1 *SSD Resnet*



Gambar 8. Flowchart Sub-proses *SSD Resnet*

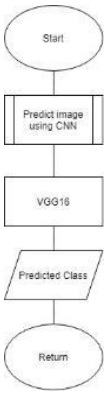
Sub-proses *SSD Resnet* digunakan untuk mendeteksi objek pada gambar. Pada sub-proses ini, objek yang akan dideteksi adalah wajah yang memakai masker dan tidak memakai masker yang terdapat pada gambar. Alur dari sub-proses ini dapat dilihat pada Gambar 10. Langkah pertama untuk proses ini adalah memprediksi semua kelas dan *bounding box* pada gambar, dimana pada proses ini semua objek akan dideteksi. Setelah itu akan dilakukan proses pengecekan *confidence*, dimana pada gambar dicari objek yang terdeteksi dengan *confidence* tertinggi. Hasil

proses prediksi semua kelas dan *bounding box* beserta dengan *confidence*-nya dapat dilihat pada gambar 11.

Gambar 9. Gambar Hasil *Predict Class* & *Bounding Box* & *Confidence*

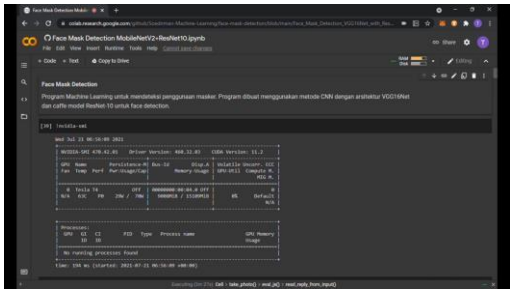
3.2.2 *Convolutional Neural Network*

Setelah melalui proses SSD Resnet, *image* akan diproses dalam CNN. Langkah pertama *image* akan dioleh menggunakan CNN lalu akan menghasilkan berupa kelas yang diprediksi. *Flowchart* dari sub-proses ini dapat dilihat pada Gambar 12.



Gambar 10. *Flowchart* Sub-proses CNN

3.3 Desain Sistem



Gambar 11. Tampilan Program Google Colaboratory

Pada bagian ini dijelaskan tentang user interface dari program. Terdapat tempat agar user dapat menjalankan program coding pelatihan dan pengujian deteksi masker menggunakan SSD-Resnet dan VGG16. Tampilan user interface Google Colaboratory terlihat seperti Gambar 13.

4. IMPLEMENTASI SISTEM

Pada bab ini dibahas tentang implementasi sistem sesuai dengan Analisa dan desain sistem. Implementasi sistem meliputi implementasi aplikasi, pengolahan data, implementasi *model neural network* untuk *training* dan *testing*.

4.1 Implementasi Aplikasi yang Digunakan

Skripsi ini dibuat menggunakan bahasa pemrograman *Python* sebagai bahasa utama yang digunakan. Alasan digunakannya bahasa pemrograman *Python* adalah untuk mempermudah penggunaan berbagai *library* yang dibutuhkan dalam skripsi ini. Untuk proses *image processing* dilakukan menggunakan *library OpenCV*. Selain *library OpenCV* juga digunakan *library-library* lain sebagai pendukung *numpy*, scikit-learn, dan pandas yang hanya tersedia dalam bahasa pemrograman Python saja. Langkah pertama untuk implementasi *neural network* adalah dengan menggunakan *library tensorflow*.

Instalasi dan konfigurasi *tensorflow* dilakukan pada laptop OS Windows 10 dengan spesifikasi GPU NVIDIA GeForce 930MX. Tensorflow yang digunakan adalah tensorflow-gpu. Tensorflow diakses menggunakan *Google Colaboratory*.

4.2 Pengolahan Data

Proses ini merupakan proses yang dijalankan pertama kali ketika program berjalan. Data *input* berupa gambar wajah yang memakai masker dan tidak memakai masker yang akan dideteksi posisinya dan akan dipilih berdasarkan tingkat keyakinan. Setelah itu gambar akan dipotong, yaitu dari keseluruhan gambar akan diambil bagian gambar yang merupakan letak wajah pada gambar. Proses pengolahan data ini dilakukan baik pada proses *training* maupun proses *testing.*

Pada tahap ini data *image* yang didapatkan dalam 2 folder yang berbeda dimana masing-masing *folder* menandakan data *image* yang berada di dalamnya akan digunakan untuk *training* atau *testing*. Data dibagi menjadi susunan seperti ini untuk mempermudah pembagian data sehingga setiap kategori dapat disesuaikan jumlahnya pada *folder training* dan *testing*. Penamaan *file* untuk gambar adalah # .jpg (# adalah nomor urut gambar).

4.2.1 *Object Training dengan VGG16Net*

Proses *object detection* ini dilakukan menggunakan metode *VGG16Net* dimana pada proses ini setiap wajah pada gambar akan dideteksi. Proses ini pertama akan membaca *file weights* yang sudah di-*train* sebelumnya, file konfigurasi serta *file* nama yang berisikan nama objek yang dapat dideteksi.

Setelah mengecek runtime siap untuk digunakan, tahap awal yang dilakukan adalah melakukan *cloning dataset* yang berada di *github* untuk memproses pelathan menggunakan metode VGG16Net. Pastikan *cloning dataset* lengkap supaya hasil proses *training* dapat berjalan dengan maksimal. File yang selanjutnya dimasukkan adalah file untuk melakukan pengujian model dengan menggunakan *res10_300x300_ssd_iter_140000.caffemodel* dan *deploy.prototxt* untuk mendeteksi bagian wajah. Tahap selanjutnya adalah mengimpor *libraries* yang dibutuhkan. *Autotime* adalah *code* opsional untuk menghitung waktu lamanya eksekusi tiap sel di *Google Colab*. Tahap selanjutnya adalah *preprocessing dataset. Dataset directory* diambil gambarnya digunakan untuk inisialisasi data dan *class* gambar. Pada *class* gambar dilakukan pengulangan untuk mengekstrak *class* label dari *filename*, memuat input gambar dan melakukan proses yang menghasilkan data dan label yang sudah di-*update* secara berurutan. Data dan label tersebut dikonversi ke dalam *NumPy Arrays* dan dilakukan *one-hot encoding on the labels*. Jika gambar

palet dengan transparansi yang dinyatakan dalam *byte* harus dikonversi ke gambar RGBA.

4.2.2 Membuat objek *ImageDataGenerator* dan Data Augmentation

Proses ini melakukan partisi data ke dalam *training* dan *testing*. Rasio *training* dibuat 75% dan *testing* 25%. Selanjutnya membentuk *training image generator* yang digunakan untuk *data augmentation*. Augmentasi data bertujuan untuk meningkatkan jumlah sampel, secara artifisial. Proses ini dilakukan untuk mencegah *overfitting* dalam proses *training*. Mengubah ukuran gambar, orientasinya, kondisi pencahayaan adalah contoh operasi untuk augmentasi data. Data buatan yang dihasilkan juga termasuk dalam *dataset training*.

4.2.3 *Pre-Trained Convnets*

Pada tahap ini dilakukan membuat model jaringan CNN yang sudah dipelajari sebelumnya. Model jaringan CNN yang sudah dibuat adalah arsitektur jaringan *VGG16Net*. Tahap ini dapat mengambil jaringan klasifikasi gambar yang telah dilatih sebelumnya yang telah *learning* untuk mengekstrak fitur yang kuat dan informatif dari gambar alami dan menggunakannya sebagai titik awal untuk mempelajari tugas baru. Menggunakan jaringan yang telah dilatih dengan pembelajaran transfer biasanya jauh lebih cepat dan lebih mudah daripada melatih jaringan dari awal. Setelah itu dilakukan ekstraksi fitur untuk menjalankan arsitektur jaringan *VGG16Net*.

4.3 Tahap Pembuatan Model

Pada tahap ini membentuk bagian *head* dari model yang akan ditempatkan pada *base* model dan dilakukan perulangan *layer* per *layer*.

4.4 Confusion Matrix

Proses ini akan dilakukan pada saat *training* untuk mengukur performa dari model yang selesai di-*train*. *Co6nfusion matrix* digunakan untuk menghitung *accuracy*, *recall*, *precision* dan juga F1-*Score*. *Confusion matrix* yang ditampilkan berdasarkan model yang sudah di-*train* lalu akan memprediksi data *test* yang sudah disediakan bersama labelnya. *Confusion matrix* yang dibuat akan terdiri dari 5 kelas yaitu jumlah jenis golongan kendaraan yang ada.

4.5 TensorFlow Lite

Tahap selanjutnya adalah menyimpan model menggunakan *tf.saved_model/save* dan kemudian mengkonversi model tersimpan ke format yang kompatibel *tensorflow lite.*

4.6 SSD Resnet

Model diujikan pada gambar dan secara real-time dengan menggunakan *res10_300x300_ssd_iter_14000.caffemodel* dan *deploy.prototxt* yang digunakan untuk mendeteksi wajah.

5. PENGUJIAN SISTEM

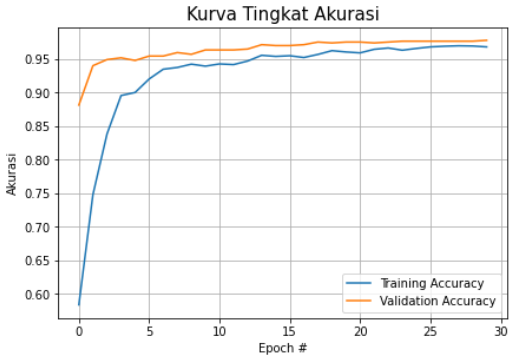
Pada bab ini dijelaskan tentang pengujian terhadap sistem dan aplikasi yang telah dibuat. Pengujian dilakukan dengan menggunakan *dataset* yang sama pada masing-masing percobaan. Jumlah *dataset* yang digunakan untuk pengujian adalah 25% dari total keseluruhan *dataset*, yaitu 523 *images*.

5.1 Pengujian terhadap hasil *training*

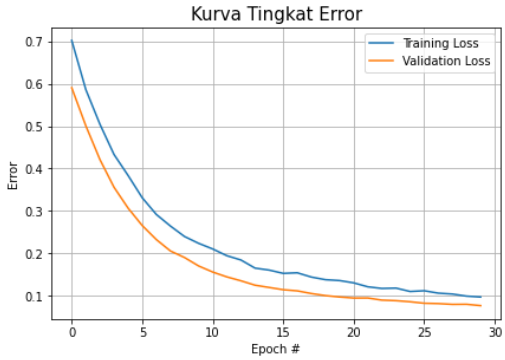
Pengujian terhadap hasil training akan mengevaluasi performa daripada sistem *training* yang telah dibuat. Pencatatan pengujian akan dibedakan berdasarkan orang yang memakai masker dan tidak memakai masker. Performa klasifikasi akan ditampilkan menggunakan *confusion matrix* beserta dengan grafik tiap klasifikasi.

5.1.1 *Pengujan Metode Convolutional Neural Network*

Pengujian dengan menggunakan metode *convolutional neural network* dilakukan berdasarkan nilai *learning rate*, banyaknya *epoch training* serta *batch size*. Pengujian dilakukan dengan nilai *epoch* yaitu 30 untuk mencapai hasil prediksi yang memuaskan. *Learning rate* yang digunakan untuk pengujian adalah 1.e-4. Batch size yang digunakan adalah 32.Grafik tingkat akurasi dan tingkat error VGG dapat dilihatdi gambar 15 dan 16.



Gambar 12. Grafik Akurasi VGG



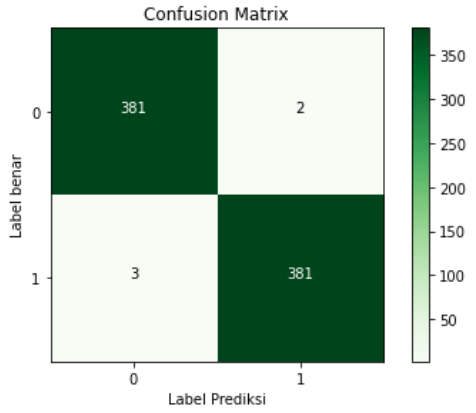
Gambar 13. Grafik *Loss* VGG

5.1.2 *Confusion Matrix*

Confusion Matrix dengan jumlah *true positive* 381 dan *true negative* 381.

Dari *confusion matrix* tersebut dapat menghasilkan *precision*, *recall*, *f1-score*, dan *accuracy*. *Precision* menggambarkan akurasi antara data yang diminta dengan hasil prediksi yang diberikan oleh model. *Precision* pada gambar dataset yang bermasker menunjukkan 0.98 yang artinya 98% akurat. *Recall* atau *sensitivity* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi. F-1 *Score* menggambarkan perbandingan rata-rata *precision* dan *recall* yang dibobotkan. *Accuracy* tepat ketika menggunakan sebagai acuan

performa algoritma jika dataset kita memiliki jumlah data *False Negatif* dan *False Positif* yang sangat mendekati (*symmetric*). Namun jika jumlahnya tidak mendekati, maka sebaiknya menggunakan F1-Score sebagai acuan. Akurasi adalah ukuran kinerja yang paling intuitif dan itu hanyalah rasio pengamatan yang diprediksi dengan benar terhadap total pengamatan. Orang mungkin berpikir bahwa, jika memiliki akurasi tinggi maka model adalah yang terbaik. Akurasi adalah ukuran yang bagus tetapi hanya jika memiliki kumpulan data simetris di mana nilai positif palsu dan negatif palsu hampir sama. Oleh karena itu, harus melihat parameter lain untuk mengevaluasi kinerja model. Hasil *Confusion Matrix* dapat dilihat di gambar 19 dan tabel hasil *Confusion Matrix* dapat dilihat di gambar 20.



Gambar 14. *Confusion Matrix*

	precision	recall	f1-score	support
with_mask	0.98	0.97	0.98	383
without_mask	0.97	0.98	0.98	384
accuracy			0.98	767
macro avg	0.98	0.98	0.98	767
weighted avg	0.98	0.98	0.98	767

Gambar 15. Detail hasil dari *confusion matrix*

6. KESIMPULAN DAN SARAN

Pada bab ini akan dijelaskan tentang kesimpulan yang diperoleh dalam penelitian untuk mengenali deteksi masker menggunakan metode *VGG16 Net* dan *SSD Resnet*. Pada bab ini juga disertakan sejumlah saran untuk pengembangan skripsi lebih lanjut.

6.1 Kesimpulan

Dari hasil perancangan sistem, dapat diambil kesimpulan yaitu performa dari model VGG16Net menggunakan modul SSD ResNet dapat menghasilkan keseluruhan performa yaitu *accuracy* sebesar 98%, f1-score sebesar 98%.

6.2 Saran

Saran yang dapat diberikan untuk menyempurnakan dan mengembangkan skripsi ini lebih lanjut antara lain: Menambah jumlah dataset yang lebih bervariasi agar dapat meningkatkan akurasi. Menguji konfigurasi lain pada VGG16Net agar dapat meningkatkan akurasi.

7. DAFTAR PUSTAKA

[1] Chen, Y., Jiang, H., Li, C., Jia, X., & Ghamisi, P. (2016). IEEE Transactions on Geoscience and Remote Sensing. *Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks*, 54(10), 6232-6251.

[2] 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..

[3] 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.

[4] Novian, T. (2018). *Detection of Motor Vehicles License Plateon Streaming Media with Convoutional Neural Network Algorithm Using Tensorflow*. https://DetectionofMotorVehiclesLicensePlateonStreamingMediawithConvoutionalNeuralNetworkAlgorithmUsingTensorflow

[5] Prabhu. (2018). *Understanding of Convolutional Neural Network (CNN) – Deep Learning*. Retrieved December 20, 2020, from https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

[6] Udofia, U. (2018). *Basic overview of Convolutional Neural Network (CNN)*. Retrieved December 20, 2020, from https://medium.com/dataseries/basic-overview-of-convolutional-neural-network-cnn-4fcc7dbb4f17

[7] V. Agarwal, "Face Detection Models: Which to Use and Why?," 42 medium.com2, 2020. [Online]. Available: https://towardsdatascience.com/face-detection-models-which-to-use-andwhy-d263e82c302c. [Accessed: 11-Dec-2020]

[8] 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