BAB II
LANDASAN TEORI

2.1 Tinjauan Teori

2.1.1 Covid-19

Coronavirus disease 2019 atau lebih dikenal sebagai COVID-19, merupakan sebuah tipe penyakit yang menginfeksi sistem pernafasan manusia dan hewan. Penyakit ini pertama kali bermula di provinsi Wuhan China sejak Desember 2019 dan telah di nyatakan sebagai pandemi oleh World Health Organization sejak 11 Maret 2020 [11]. Virus ini memiliki tingkat penularan yang tinggi karena tingkat transmisinya yang sangat mudah melalui tetesan kecil (*droplet*) seperti cairan yang dikeluarkan saat orang batuk, bersin, atau air liur yang dikeluarkan saat bicara lah yang membuat infeksi dari penyakit ini sangat mudah [2].

Gejala yang biasanya di rasakan oleh pasien yang terjangkit Covid-19 berupa demam ringan, batuk-batuk, dan dispnea hingga badai sitokin, gagal napas, dan kematian. Namun ada juga pasien yang tidak menunjukan gejala sehingga dapat membahayakan orang disekitar mereka [11].

Oleh sebab itu, seluruh negara di dunia mengeluarkan kebijakan sebagai upaya untuk mencegah penularan Covid-19 salah satu upaya tersebut adalah dengan menggunakan masker saat beraktifitas di luar. Indonesia telah menerapkan 3M (Memakai masker, Mencuci tangan dan Menjaga jarak) sebagai metode pencegahan penularan [4]. Pemerintah juga membuat rapid test dan PCR test sebagai sarana untuk mendeteksi kalau seseorang tertular penyakit Covid-19 [12].

2.1.2 Python

Python adalah bahasa pemrograman yang interpretatif multiguna yang memakai filosofi perancangan dengan fokus kepada tingkat keterbacaan kode. Python dibuat oleh Guido van Rossum pada tahun 1991. Python memiliki teknik menggabungkan kemampuan, kapabilitas dan sintaksis kode serta fungsi pustaka yang berkualitas tinggi [13]. Python juga dikenal dengan bahasa pemrograman yang mudah dipelajari, karena struktur sintaknya rapi dan mudah dipahami.

2.1.3 Keras

Keras adalah *opensource library* yang yang menyediakan antarmuka Python untuk *library neural network* lainnya. Keras mampu berjalan menggunakan MXnet, Deeplearning4j, Tensorflow, *Microsoft Cognitive Toolkit* atau Theano. Keras dirancang untuk melakukan eksperimen cepat dengan *deep neural network*, hal ini berfokus pada ukuran minimal, *modular* dan dapat diperluas kembali. Keras dikembangkan sebagai bagian dari upaya penelitian proyek ONEIROS dan penulis dan pengelola utamanya adalah Francois Chollet. Pada tahun 2017, tim Google Tensorflow memutuskan untuk mendukung Keras dalam TensorFlow's *core library* [14].

2.1.4 OpenCV

OpenCV (*Open Computer Vision*) adalah sebuah API *library* yang digunakan untuk memproses gambar *computer vision* atau *image processing*. OpenCV sangat di rekomendasikan untuk menciptakan aplikasi yang berbasis *image processing* karena library ini merupakan *library* yang kuat dalam bidang *digital vision* dan memiliki kapabilitas prosesing *visual* seperti manusia[14].

2.1.5 Tensorflow

Tensorflow adalah *library opensource* yang dikembangkan oleh *Google Brain* untuk melakukan *machine learning* dan *neural network*. Tensorflow menggabungkan aljabar komputasi dengan teknik optimasi kompilasi, yang

memfasilitasi perhitungan banyak ekspresi matematika. Fitur utama yang terdapat dalam tensorflow adalah:

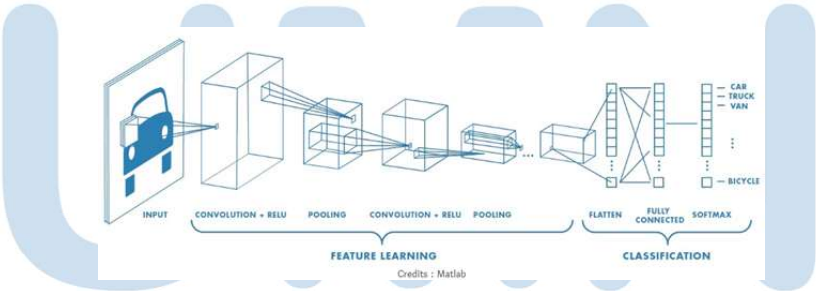
1. Mendefinisikan, mengoptimalkan, dan menghitung secara matematis ekspresi wajah yang melibatkan *array multidimension* (*tensors*).

2. Pemrograman pendukung jaringan syaraf dalam dan teknik machine learning.
3. Pemakaian GPU (*Graphics Processing Unit*) yang efisien, mengotomasi manajemen dan optimalisasi memori yang sama terhadap data yang digunakan. Tensorflow mampu menulis kode yang sama dan menjalankannya di CPU atau GPU. Lebih khususnya lagi tensorflow dapat mengetahui bagian mana yang harus dipindahkan ke GPU.

4. Skalabilitas komputasi yang tinggi pada keseluruhan mesin terhadap kumpulan data yang besar [15].

2.1.6 Convolutional Neural Network

Algoritma Convolutional Neural Network (CNN) merupakan pengembangan dari jaringan syaraf tiruan yang terinspirasi dari jaringan syaraf manusia, dan biasanya digunakan dalam data gambar untuk mendeteksi dan mengenali objek dalam gambar.



Gambar 2.1 Ilustrasi Algoritma CNN

Sumber: [16]

Convolutional Neural Network (CNN) merupakan pengembangan dari *Multilayer Perceptron* (MLP) yang dirancang untuk mengolah data dua dimensi. Di CNN, setiap neuron direpresentasikan dalam dua dimensi, berbeda

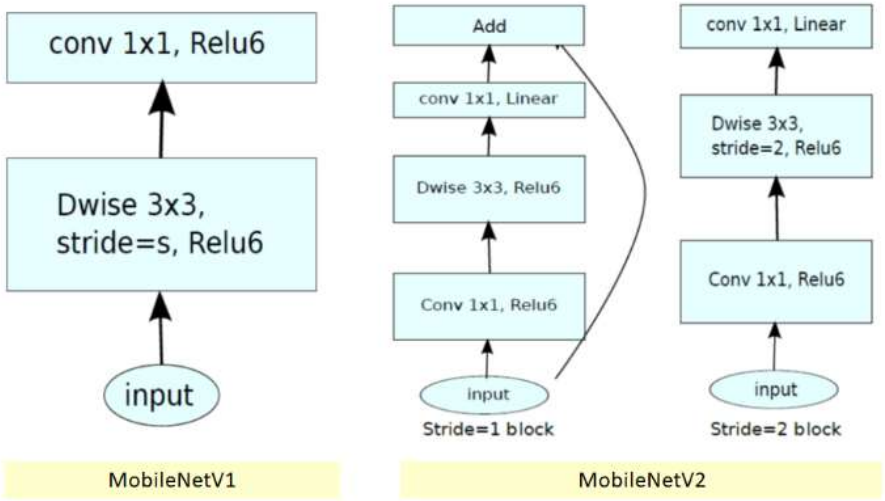
dengan MLP, pada MLP setiap neuron hanya satu dimensi. CNN termasuk dalam deep neural network karena kedalaman jaringannya yang tinggi dan banyak digunakan dalam data gambar. CNN hampir sama dengan neural network pada umumnya, umumnya yang memiliki neuron yang memiliki bobot dan bias. CNN memiliki 1 tahap *training* (*Supervised Backpropagation*). Secara teknis, CNN adalah sebuah arsitektur yang dapat dilatih dan terdiri dari beberapa tahap. Masukan (*input*) dan keluaran (*output*) dari setiap tahap adalah terdiri dari beberapa *array* yang biasa disebut *feature map*.

CNN mempunyai beberapa jenis *layer* yang dapat digunakan, yaitu *subsampling lay*er, *convolutional layer, loss layer* dan *fully connected layer*. *Subsampling layer* biasa digunakan untuk mengurangi dimensi yang berasal dari *input*. Tujuannya adalah untuk mengurangi jumlah parameter yang dibutuhkan dalam CNN dan juga untuk membuat CNN invarian terhadap perubahan kecil di dalam suatu citra. *Convolution layer* bekerja dengan meniru sifat-sifat dari *visual cortex* otak dan mempelajari filter-filter dari *input image*. *Layer* ini disebut dengan *convolutional layer* karena operasi yang dilakukan adalah *convolution* antara filter dengan input image. Filter yang dipelajari pada layer ini dapat berbentuk berbagai macam, misalnya jika digunakan untuk mempelajari image, maka *filter* tersebut mungkin mempelajari untuk melakukan *edge detection*. *Loss layer* merupakan *output layer* dari CNN. Dalam klasifikasi citra, jika terdapat dua kelas klasifikasi, maka *loss layer* yang digunakan adalah *sigmoid loss*. Pada klasifikasi citra dengan banyak kelas, maka digunakan distribusi *Categorical*, yang biasa disebut dengan *softmax layer*. Pada permasalahan lain, dapat digunakan *loss layer* yang sesuai. Seperti pada permasalahan *regression*. *Fully connected layer* merupakan *layer* yang biasa ditemukan pada *Neural Network* biasa. Berbeda dengan *convolutional layer*, *fully connected layer* tidak menggunakan operasi convolution untuk mendapatkan keluaran dari *layer* tersebut, tetapi menggunakan perkalian matriks. Dengan penggunaan operasi tersebut, maka setiap *input node* akan

dihubungkan dengan *hidden node* yang ada di dalam *layer*, sehingga *layer* ini disebut sebagai *fully connected layer* [17].

2.1.7 *MobileNet*

MobileNet adalah salah satu arsitektur dari *Convolutional Neural Network* (CNN) yang pertama kali di rilis di tahun 2017, sebagai arsitektur CNN dalam *computer vision* (CV) yang ringan dan cocok untuk di implementasi ke perangkat *mobile*. Inti dari arsitektur MobileNet berbasis dari menggunakan *depth-wise seperable convolution* untuk membangun *deep neural network* yang ringan. Depthwise separable convolution terdiri dari 2 operasi yaitu *depth wise* dan *point wise convolution*, dimana *depth wise convolution* melakukan pemfilteran ringan dengan menerapkan filter konvolusi tunggal per saluran input kemudian *point wise convolution layer* bertanggung jawab untuk membangun fitur baru melalui komputasi kombinasi linier dari saluran input. MobileNetV2 merupakan iterasi kedua dari *MobileNetV1*, yang memiliki konsep dan cara kerja yang mirip dengan pendahulunya dengan menambahkan fitur *linear bottleneck* dan *shortcut connections* antara *bottleneck* sehingga dapat mencapai waktu *training* yang lebih cepat dengan akurasi yang lebih baik[18]. Komparasi *MobileNetV1* dan *MobileNetV2* dapat seperti di gambar 2.2.

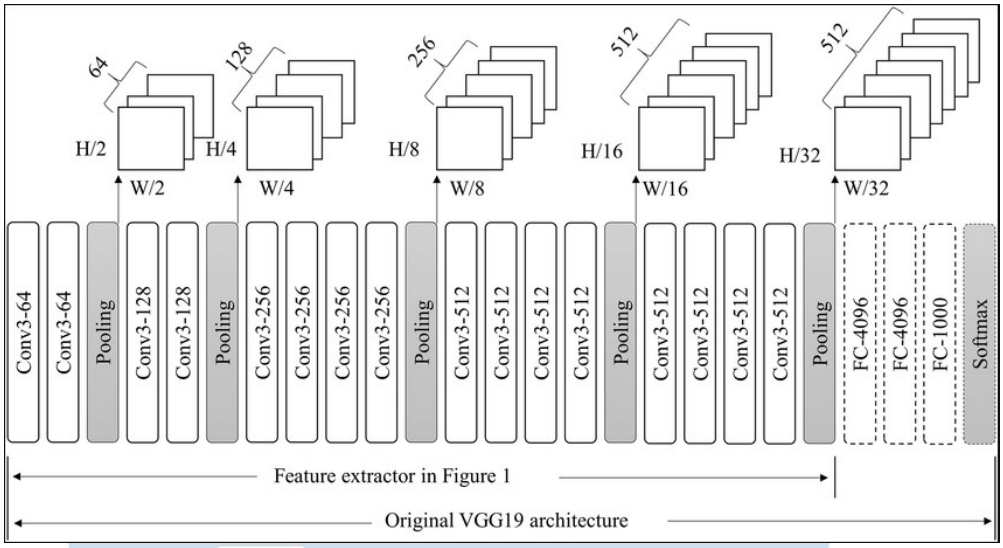


Gambar 2.2 Komparasi arsitektur MobileNet

Sumber: [18]

2.1.8 *VGGNet*

Visual Geometry Group atau VGGNet merupakan arsitektur *deep Convolutional Neural Network* (CNN) yang di rilis di tahun 2014. VGG memiliki 2 tipe arsitektur yaitu VGG16 dan VGG19 dimana yang membedakan keduanya adalah jumlah *convolutional layer* dalam setiap arsitektur dimana VGG16 memiliki 16 *convolutional layer* dan VGG19 memiliki 19 *convolutional layer*. Kekurangan dari network ini adalah, karena ini *network* yang besar dengan parameter yang banyak, membuat waktu pelatihan setiap parameter memakan waktu yang lama dan memiliki ukuran yang besar[19]. Contoh dari arsitektur VGGNet dapat dilihat seperti gambar 2.3.

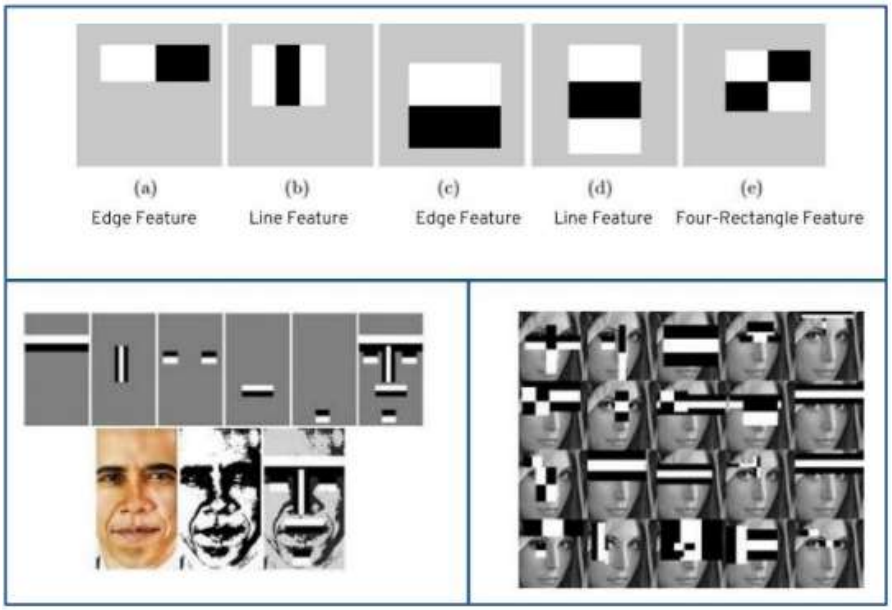


Gambar 2.3 Contoh Arsitektur VGG19

Sumber: [19]

2.1.9 *Haar Feature Based Cascade*

Haar Cascade Classifier merupakan *rectangular* (persegi) *feature* yang memberikan indikasi spesifik pada sebuah gambar. Secara teknis, *Haar Cascade Classifier* mengenali objek berdasarkan nilai sederhana dari fitur tetapi bukan merupakan nilai piksel dari image objek tersebut. Metode ini memiliki kelebihan yaitu komputasi yang sangat cepat, karena hanya tergantung pada jumlah piksel dalam persegi bukan setiap nilai piksel dari sebuah image. Metode ini merupakan metode yang menggunakan *statistical model* (*classifier*). Pendekatan untuk mendeteksi objek dalam gambar menggabungkan empat kunci utama yaitu *Haar like feature*, *Integral Image*, *Adaboost learning* dan *Cascade Classifier*.



Gambar 2.4 Contoh Haar Feature Based Cascade

Sumber: [20]

Model klasifikasi *Haar Cascade* adalah *library* yang tersedia di OpenCV, yang dibangun di atas bahasa C / C ++ menggunakan python API. *Haar Cascade* menggabungkan tiga elemen dasar. Pertama adalah memiliki sekumpulan fitur yang dapat dihitung secara akurat dan cepat, yang dapat mengurangi variabilitas dalam suatu kelas dan meningkatkan variabilitas antar kelas. Yang kedua adalah mengimplementasikan algoritma yang memungkinkan pemilihan fitur dan pelatihan. Ketiga, secara bertahap membentuk *cascade*, hasil skema klasifikasi dan deteksi akan lebih kompleks, cepat dan efisien. Haar *Cascade* dapat dilatih untuk mendeteksi banyak objek, yang harus kita lakukan adalah menentukan area yang paling memungkinkan di wajah. Wajah memiliki kulit dan memiliki warna kulit setingkat piksel. Pilih teknik segmentasi untuk warna piksel wajah. Kemudian menggunakan pengklasifikasi Haar Cascade untuk memverifikasinya. Jika piksel yang diverifikasi cocok dengan gambar geometris, sistem akan menemukan wajah yang relevan. Jika tidak cocok, sistem akan mengabaikannya [21].

2.1.10 *Flask*

Flask adalah suatu *web framework* berbasis Python. *Flask* adalah WSGI (*Web Server Gateway Interface*) *micro framework* yang menyediakan fungsionalitas minimal dengan kemampuan untuk menambah modul sesuai kebutuhan pengembang. Flask dapat menghasilkan aplikasi seperti halaman web yang menggunakan HTML, CSS, dan JavaScript.[22]

2.2 Penelitian Terdahulu

Sebelumnya sudah ada beberapa penelitian terdahulu yang mengambil topik "*face mask detection*" menggunakan beragam model dan algoritma.

Table 2.1 Penelitian Terdahulu

Judul	Jurnal	Penjelasan dan Kesimpulan
A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic[6]	**Nama Peneliti** M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa **Nama Jurnal** Measurement: Journal of the International Measurement Confederation, 2021, vol 167.	**Latar Belakang:** Penelitian dalam jurnal ini menggunakan algoritma *Support Vector Machine* (SVM) dan *ensemble method* sebagai metode deteksi *facemask*. Penelitian ini terdiri dari dua komponen, dimana komponen pertama melakukan *feature extraction* dengan menggunakan Resnet50. Dan komponen kedua untuk proses klasifikasi masker dengan menggunakan *decision tree*, SVM dan *ensemble algorithm*. Penelitian ini menggunakan 3 tipe dataset untuk di uji coba yaitu *Real-World Masked Face Dataset* (RMFD) yang memiliki jumlah data sebanyak 5000 foto dengan masker dan 9000 foto dengan orang tanpa masker, *the Simulated Masked Face*

Judul	Jurnal	Penjelasan dan Kesimpulan
		Dataset (SMFD) dengan total gambar sebanyak 785 gambar untuk orang dengan masker dan 785 gambar orang tanpa masker sehingga pembagian data gambar untuk dataset ini menjadi seimbang, dan the *Labeled Faces in the Wild* (LFW) yang memiliki 13000 gambar artis yang menggunakan masker. **Hasil** Di akhir penelitian ini, SVM berhasil mencapai akurasi sebesar 99.64% untuk dataset RMFD, 99.49% untuk SMFD dan 100% untuk LFW.
Multi-Stage CNN Architecture for Face Mask Detection[23]	**Nama Peneliti** A. Chavda, J. Dsouza, S. Badgujar, and A. Damani **Nama Jurnal** 2021 6th International Conference for Convergence in Technology (I2CT)	**Metode:** Rancangan dari penelitian ini menggunakan arsitektur CNN dengan 2 tahap. Cara kerja arsitektur ini berupa, tahap pertama berfungsi untuk menangkap satu atau lebih muka orang dalam gambar terlebih dahulu, kemudian gambar yang telah tertangkap kemudian di kirim ke tahap kedua untuk melakukan klasifikasi menggunakan CNN untuk menentukan apakah orang-orang di gambar tersebut menggunakan masker atau tidak. Dalam tahap klasifikasi, model CNN dilatih dengan 3 model *image classification* yaitu: MobileNetV2, DenseNet121, dan

Judul	Jurnal	Penjelasan dan Kesimpulan
		NASNet. Permasalahan yang ada dalam menggunakan metode ini adalah metode ini membutuhkan data *training* yang banyak dan membutuhkan *hardware* bagus agar waktu *training* dapat dilakukan dengan cepat. **Hasil:** Dari ketiga model yang dilakukan *training*, NASNetMobile memiliki performa akurasi terbaik dibanding model lain dengan nilai 99.82% untuk *training* dan 99.45% untuk *validation*.
Face Mask Detection using Transfer Learning of InceptionV3 [10]	**Nama Peneliti** G. Jignesh Chowdary, N. S. Punn, S. K. Sonbhadra, and S. Agarwal **Nama Jurnal** Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2020, vol. 12581	**Metode:** Penelitian ini menggunakan algoritma *deep neural network*, inceptionV3. Dataset yang digunakan dalam penelitian ini berasal dari *Simulated Masked Face Dataset* (SMFD) yang memiliki pembagian dataset gambar yang seimbang untuk gambar orang yang menggunakan masker dan yang tidak menggunakan masker. *Deep neural network* adalah suatu sistem pemrosesan informasi yang mencoba meniru kinerja otak manusia. DNN yang digunakan untuk melakukan klasisfikasi gambar memiliki performa yang lebih unggul dibanding algoritma lainnya, akan tetapi membutuhkan kekuatan

Judul	Jurnal	Penjelasan dan Kesimpulan
		komputasi/*hardware* yang kuat, memakan sumber daya yang banyak dan memakan waktu yang lama. **Hasil:** Penelitian ini berhasil mendapat *accuracy score* sebesar 99.9% saat *training* dan 100% saat *testing*.
Face mask detection using deep learning: An approach to reduce risk of Coronavirus spread [24]	**Nama Peneliti** S. Sethi, M. Kathuria, and T. Kaushik **Nama Jurnal** Journal of Biomedical Informatics, 2021, vol. 120	**Metode:** Penelitian ini menggunakan teknik *one-stage dan two-stage detector* untuk mencapai hasil *interference* waktu yang rendah dan memiliki akurasi yang tinggi dengan menggunakan *deep learning*. Penelitian menggunakan dataset MAFA yang memiliki total data sebanyak 25,876 untuk *training* dan *testing* dengan tambahan data gambar mahasiswa sekaligus memanfaatkan algoritma *image processing* untuk mendeteksi nama dan id mahasiswa yang ada di gambar. *Dataset* di latih dengan *library* python bernama *caffe* untuk menghilangkan bias dari gambar agar gambar *facemask* saja yang di deteksi. Penelitian ini akan menggunakan tiga model visual yaitu ResNet50, AlexNet and MobileNet. **Hasil:**

Judul	Jurnal	Penjelasan dan Kesimpulan
		Hasil implementasi menggunakan model ResNet50, mencapai hasil akurasi yang tinggi sebesar 98.2% dibanding kedua model lainnya.

Berdasarkan beberapa penelitian terdahulu seperti yang terdapat di table 2.1, dapat disimpulkan bahwa banyak penelitian membuat model *machine learning* berbasis CNN dan *Tensorflow* untuk melakukan deteksi masker. Namun sebagian besar penelitian tersebut hanya berforkus untuk pengembangan modelnya saja dengan *volume* data yang berbeda, salah satu bentuk implementasi model dari penelitian terdahulu merupakan penerapan model deteksi masker ke CCTV seperti dalam penelitian "*A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic"* [6] dan *Face mask detection using deep learning: An approach to reduce risk of Coronavirus spread* [24]. Oleh sebab itu, penelitian ini akan berforkus untuk merancang aplikasi web untuk melakukan deteksi masker menggunakan python *webframework flask* dengan menggunakan salah satu model *image classification* yang digunakan dalam penelitian terdahulu yaitu *mobilenetV2* dengan tambahan model *VGG19* dimana hasil performa dari kedua model akan diuji dan model terbaiklah yang akan digunakan untuk aplikasi web yang dibangun.