

PENGEMBANGAN WEBSITE PENGENALAN WAJAH DENGAN *LIBRARY face-api.js* & MEMAHAMI KONSEP *CONVOLUTIONAL NEURAL NETWORK (CNN)*

TUGAS AKHIR

**Diajukan untuk Memenuhi Salah Satu Syarat dalam Menempuh Ujian Sidang
Sarjana di Program Studi Informatika**

Oleh:

**Mochamad Darmawan Hardjakusumah
0618101098**



SK BADAN AKREDITASI PERGURUAN TINGGI (BAN-PT)

Nomor: 2035/SK/BAN-PT/Akred/S1/IX/2016

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK UNIVERSITAS WIDYATAMA
BANDUNG**

2023

LEMBAR PENGESAHAN

**PENGEMBANGAN WEBSITE PENGENALAN WAJAH DENGAN *LIBRARY*
face-api.js & MEMAHAMI KONSEP *CONVOLUTIONAL NEURAL NETWORK*
(CNN)**

TUGAS AKHIR

**Program Studi Informatika
Fakultas Teknik
Universitas Widyatama**

Oleh:

**Mochamad Darmawan Hardjakusumah
0618101098**

Telah disetujui dan disahkan di Bandung, Tanggal .../.../2023

Pembimbing Kampus

**Sunjana, S.Si., M.T.
NIDN. 0413126803**

Ka. Prodi Teknik Informatika,

Dekan Fakultas Teknik,

**Ari Purno Wahyu, S.Kom., M.Kom.
NIDN. 0415078402**

**Dr. Didit Damur Rochman, S.T., M.T.
NIDN 0415117401**

LEMBAR PENGESAHAN

**PENGEMBANGAN WEBSITE PENGENALAN WAJAH DENGAN *LIBRARY*
face-api.js & MEMAHAMI KONSEP *CONVOLUTIONAL NEURAL NETWORK*
(CNN)**

TUGAS AKHIR

**Program Studi Informatika
Fakultas Teknik
Universitas Widyatama**

Oleh:

**Mochamad Darmawan Hardjakusumah
0618101098**

Telah disetujui dan disahkan di Bandung, Tanggal .../.../2023

Pembimbing Kampus



Sunjana, S.Si., M.T.

NIDN. 0413126803

Ka. Prodi Teknik Informatika,

Dekan Fakultas Teknik,



Ari Purno Wahyu, S.Kom., M.Kom.

NIDN. 0415078402

Dr. Didit Damur Rochman, S.T., M.T.

NIDN 0415117401

SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini :

Nama : Mochamad Darmawan Hardjakusumah
Tempat dan Tanggal Lahir : Karawang, 22 Oktober 2000
NPM : 0618101098
Alamat Orang Tua : Perum Pemda Blok C 1 No. 11 A, TelukJambe Timur.
Karawang

Menyatakan bahwa Skripsi ini dengan judul “**PENGEMBANGAN WEBSITE PENGENALAN WAJAH DENGAN *LIBRARY face-api.js* & MEMAHAMI KONSEP *CONVOLUTIONAL NEURAL NETWORK (CNN)***” adalah benar hasil karya saya sendiri.

Apabila terbukti tidak demikian, saya bersedia menerima segala akibatnya, termasuk pencabutan kembali gelar sarjana teknik yang telah saya peroleh. Demikian surat pernyataan ini dibuat sebagaimana mestinya dan benar adanya.

Bandung, .../.../2023

Mochamad Darmawan Hardjakusumah

ABSTRAK

Penelitian ini bertujuan mengembangkan sebuah website yang dapat mengenali wajah manusia, namun tanpa membuat model secara mandiri. Yaitu, dengan memanfaatkan *library face-api.js*. Selain itu, penelitian ini juga mencoba untuk memahami konsep *AI (Artificial Intelligence)* dan *Machine Learning (ML)*, khususnya pada teknik *Convolutional Neural Network (CNN)*. Pada penelitian ini terdapat tantangan utama yaitu serangan *spoofing* saat pengenalan wajah menggunakan live video webcam, tantangan ini berhasil diatasi dengan *library Silent-Face-Anti-Spoofing*. Hasil penelitian menunjukkan bahwa *library face-api.js* efektif dalam mengenali wajah, dan *library Silent-Face-Anti-Spoofing* juga baik dalam mengenali gambar wajah yang tidak asli. Penelitian ini memiliki potensi aplikasi luas dalam bidang keamanan dan identifikasi pengguna.

Kata Kunci: Pengenalan Wajah, *Machine Learning*, *Convolutional Neural Network*, *face-api.js*, Serangan *Spoofing*.

ABSTARCT

This research aims to develop a website capable of recognizing human faces, but without create a model from scratch, by utilizing the face-api.js library. Additionally, this research also seeks to understand the concepts of Artificial Intelligence (AI) and Machine Learning (ML), particularly focusing on Convolutional Neural Network (CNN) techniques. The main challenge in this research is spoofing attacks during face recognition using a live video webcam, and this challenge was successfully addressed with the Silent-Face-Anti-Spoofing library. The results indicate that the face-api.js library is effective in recognizing faces, and the Silent-Face-Anti-Spoofing library is also proficient in identifying non-authentic facial images. This research has the potential for broad applications in the fields of security and user identification.

Keywords: Face Recognition, Machine Learning, Convolutional Neural Network, face-api.js, Spoofing Attacks.

KATA PENGANTAR

Pertama-tama penulis ini mengucapkan syukur Alhamdulillah, segala puji syukur bagi Allah Subhanahu wa ta'ala karena atas taufiq dan hidayah-Nya penulis dapat menyelesaikan kegiatan tugas akhir ini yang merupakan salah satu syarat untuk menempuh ujian sarjana Teknik Informatika Universitas Widyatama.

Dalam proses penyelesaian tugas akhir ini, penulis menerima banyak sekali dukungan dari beberapa pihak. Untuk itu, melalui halaman ini penulis ingin mengucapkan banyak terimakasih kepada:

1. Mamah dan Papah tercinta, yang senantiasa memberikan doa, dukungan, serta pendidikan baik secara formal maupun melalui berbagai kursus yang telah didukung secara finansial.
2. Kepada Bapak Ari Purno Wahyu, S.Kom., M.Kom., selaku Ketua Program Studi di Fakultas Teknik Informatika, Universitas Widyatama, atas arahan dan bantuan-bantuan lainnya dalam perjalanan perkuliahan.
3. Kepada Bapak Sunjana, S.Si., M.T. Selaku pembimbing, terimakasih atas waktu, tenaga, dan kesabaran yang telah diberikan dalam membimbing penulis agar dapat menyelesaikan laporan tugas akhir ini.
4. Ibu Yan Pupitarani, S.T., M.T. Selaku Sek. Prodi Teknik Informatika dan Bapak Dr. Feri Sulianta, S.T., M.T. Selaku Ka. Lab. Information Technology, atas kontribusi dan bantuan dalam pengembangan penelitian ini.
5. Ibu Azizah Zakiah, S.Kom., M.T., selaku dosen wali, atas arahan dan panduan yang berharga selama perjalanan akademik saya.
6. Terima kasih kepada teman-teman sekelas dan teman-teman Kerja Praktek di Fakultas Ilmu Budaya dari angkatan yang sama atau berikutnya (angkatan 19-20).
7. Seluruh Civitas Akademika Universitas Widyatama, yang telah menciptakan lingkungan belajar yang inspiratif dan mendukung.

Penulis menyadari bahwa tugas akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis menerima dengan tulus terhadap kritikan dan masukan, guna perbaikan yang lebih baik. Terakhir, semoga penulisan ini dapat bermanfaat yang berarti.

Bandung, .../.../2023

Mochamad Darmawan Hardjakusumah

DAFTAR ISI

LEMBAR PENGESAHAN	i
SURAT PERNYATAAN	ii
ABSTRAK	iii
ABSTRACT	iv
KATA PENGANTAR	v
DAFTAR ISI	vii
DAFTAR TABEL	ix
DAFTAR GAMBAR	x
BAB I PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Batasan Masalah	3
1.4. Tujuan Penelitian	3
1.5. Manfaat Penelitian	4
1.6. Metodologi Penelitian	4
1.7. Sistematika Penelitian	5
BAB II LANDASAN TEORI	7
2.1. Pengolahan Citra atau <i>Image Processing</i>	7
2.2. <i>Artificial Intelligence (AI)</i>	7
2.3. <i>Machine Learning (ML)</i>	8
2.4. <i>Neural Network (NN)</i>	10
2.4.1. Cara Kerja <i>Neural Network (NN)</i>	11
2.4.1.1. Contoh Soal <i>Neural Network (NN)</i>	13
2.4.2. Jenis-jenis <i>Neural Network (NN)</i>	19
2.4.3. Hubungan <i>Neural Network (NN)</i> Dengan <i>Deep Learning (DL)</i>	20
2.5. <i>Convolutional Neural Network (CNN)</i>	20
2.5.1. Cara Kerja <i>Convolutional Neural Network (CNN)</i>	22
2.5.1.1. Contoh Soal <i>Convolutional Neural Network (CNN)</i>	26
2.6. <i>TensorFlow (TF)</i>	37
2.6.1. <i>Tensor</i>	37

2.7. <i>Library face-api.js</i>	38
2.8. <i>Library Silent-Face-Anti-Spoofing</i>	39
2.8.1. <i>Liveness</i>	39
BAB III ANALISIS DAN PERANCANGAN SISTEM	41
3.1. Usecase Diagram Gambaran Umum Sistem	41
3.2. Flowchart Diagram Simpan Gambar Sementara Pada <i>Local Storage</i>	41
3.3. Flowchart Diagram Upload Citra Wajah	42
3.4. Flowchart Halaman Pengenalan Wajah Pada Citra	42
3.5. Flowchart Diagram <i>CNN</i> Dalam Kostumisasi Dataset	43
BAB IV IMPLEMENTASI DAN PENGUJIAN	44
4.1. Implementasi	44
4.2.1. Implementasi <i>Library face-api.js & silent-face-anti-spoofing</i>	44
4.2.2. Antarmuka Pengguna & Design Sistem	45
4.2. Pengujian Skenario I Halaman Training	47
4.3. Pengujian Skenario II Halaman Test	51
BAB V PENUTUP	54
5.1. Kesimpulan	54
5.2. Saran	55
LAMPIRAN	56
DAFTAR PUSTAKA	57

DAFTAR TABEL

Table 1	Tabel Soal Diketahui untuk Soal <i>ANN</i>	13
Table 2	Pengujian Halaman Training	50
Table 3	Pengujian Halaman Test	52

DAFTAR GAMBAR

Fig. 1 Visualisasi Citra Digital	7
Fig. 2 Paradigma Kecerdasan Buatan	8
Fig. 3 Perbedaan Tradisional & ML Pemrograman	8
Fig. 4 Visualisasi Implementasi Saraf Otak Pada Manusia Pada Neural Network	10
Fig. 5 Visualisasi Perhitungan Untuk ke 1 Neuron Berikutnya	11
Fig. 6 Rumus Perhitungan	11
Fig. 7 Visualisasi ANN (Untuk Soal)	13
Fig. 8 Visualisasi Jaringan Saraf Tiruan	20
Fig. 9 Visualisasi Proses CNN	20
Fig. 10 Filter Yang Ditentukan Untuk Mendeteksi Sesuatu	21
Fig. 11 Filter Hasil Pembelajaran	21
Fig. 12 Visualisasi Konvolusi	22
Fig. 13 Visualisasi Kerja Fungsi Konvolusi	23
Fig. 14 Visualisasi Filter deteksi yang Didapatknan Dari Hasil Pembelajaran	23
Fig. 15 Visualisasi Fungsi <i>Flatten</i>	24
Fig. 16 Visualisasi Fungsi Pooling	25
Fig. 17 Visualisasi Fungsi Flatten	25
Fig. 18 Tensor & Flow	37
Fig. 19 Tensor	37
Fig. 20 Gambaran Umum Sistem	41
Fig. 21 Flow Gambaran Penyimpanan Semestara Data Gambar Pada Local Storage	41
Fig. 22 Flow Upload Gambar / Citra ke Server	42
Fig. 23 Flow Sistem Melakukan Prediksi Nama Wajah Yang Diberikan	42
Fig. 24 Gambar Tambahan Untuk Flow CNN Bekerja	43
Fig. 25 Gambar Tampilan Awal Sistem	45
Fig. 26 Halaman Training (Tamilan Awal / Pilihan)	46
Fig. 27 Tampilan Upload Melalui Webcam	46
Fig. 28 Tampilan Upload Melalui Pilih File	46
Fig. 29 Pengenalan Melalui Live Webcam atau File, dan Deteksi Spoofing	47

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dalam era teknologi yang terus berkembang pesat, pengenalan wajah manusia telah menjadi aspek penting dalam berbagai aplikasi, seperti keamanan dan identifikasi pengguna. Penelitian ini bertujuan untuk mengembangkan sebuah website yang mampu mengenali wajah manusia namun tanpa membangun teknologi tersebut secara mandiri, melainkan dengan memanfaatkan *library face-api.js*.

Pada penelitian ini, walaupun tidak mengembangkan teknologi pengenalan wajah secara mandiri atau model sendiri. Pada kesempatan ini, dilakukan riset dengan mengutip dari berbagai sumber tentang bagaimana komputer dengan berteknologi pengenalan wajah dapat melakukan identifikasi terhadap gambar atau video wajah yang diberikan.

Oleh karena itu, selain mengembangkan website pengenalan wajah, penelitian ini juga mencakup studi mendalam tentang *AI (Artificial Intelligence)* dan *Machine Learning (ML)* yang berfokus pada teknik *Convolutional Neural Network (CNN)*.

Awalnya, penelitian ini dilatarbelakangi oleh keinginan peneliti untuk membuat sebuah aplikasi web yang mampu melakukan pengenalan wajah untuk tujuan absensi. Selama pencarian solusi untuk mencapai tujuan tersebut, ditemukan sebuah *library* bernama *face-api.js* yang dikembangkan menggunakan bahasa pemrograman *JavaScript*. Setelah berhasil mengimplementasikan *library* tersebut, peneliti menjadi penasaran tentang bagaimana *library* ini bekerja. Karena rasa penasaran, menjadikan terdorong untuk menggali lebih dalam tentang bagaimana teknologi pengenalan wajah dapat diimplementasikan. Selama proses studi, ternyata hal ini sangat menarik karena terkait dengan sains/ilmu (ilmu komputer) dan memutuskan untuk mengangkat studi ini sebagai bagian dari tugas akhir yang akan menghasilkan sebuah paper yang akan menjadi sumber referensi terkhusus bagi peneliti untuk di masa depan, dan umumnya untuk pembaca bagi

yang tertarik atau memerlukan pemahaman tentang teknologi *AI* dan *ML* terkhusus pada teknik *CNN*.

Dalam membangun dan mengembangkan website pengenalan wajah ini, terdapat juga beberapa tantangan yang perlu diatasi, salah satunya yaitu serangan *spoofing* atau penipuan ketika dilakukan pengenalan melalui live video webcam, di mana seseorang berupaya untuk menipu sistem dengan memberikan gambar palsu atau bukan wajah asli. tantangan ini berhasil diatasi dengan menggunakan *library Silent-Face-Anti-Spoofing*, *library* ini dikembangkan menggunakan bahasa pemrograman *Python* sehingga ini menjadi tantangan lain juga untuk mengintegrasikan andata bahasa pemrograman *JavaScript* dan *Python*, namun syukur tantangan tersebut dapat teratasi.

Hasil penelitian ini menunjukkan bahwa *library face-api.js* sangat efektif dalam mengenali wajah manusia pada data gambar yang diberikan, dan *library Silent-Face-Anti-Spoofing* juga efektif dalam mengenali gambar wajah yang bukan asli.

Melalui penelitian ini dan mengatasi permasalahan tersebut, diharapkan akan terciptanya solusi pengenalan wajah yang aman, fleksibel dan cepat. Sehingga penelitian ini dapat digunakan dalam berbagai bidang aplikasi seperti keamanan, identifikasi pengguna, dan lain sebagainya, sesuai dengan kebutuhan yang diperlukan.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah disampaikan, rumusan dalam penelitian ini dapat dibagi menjadi beberapa poin utama:

- a. Bagaimana konsep dan cara kerja dari teknologi *Machine Learning (ML)* dengan fokus pada teknik *Convolutional Neural Network (CNN)* dalam pengenalan wajah?
- b. Bagaimana cara mengimplementasikan *library face-api.js* ke dalam halaman *web* sehingga sistem dapat melakukan proses pengenalan wajah?
- c. Bagaimana cara untuk mengatasi permasalahan serangan *spoofing* atau penipuan ketika system melakukan proses pengenalan wajah?

1.3. Batasan Masalah

Berdasarkan latar belakang yang telah disampaikan, cakupan dan batasan dalam penelitian dapat diuraikan sebagai berikut:

- a. Penelitian ini akan berfokus pada pemahaman dan cara kerja dari teknologi *Machine Learning (ML)* khususnya pada teknik *Convolutional Neural Network (CNN)*, namun tanpa membangun atau mengimplementasikan pembuatan model secara mandiri, penelitian ini hanya akan difokuskan pada implementasi *library face-api.js*, yang mana *library* ini telah menerapkan konsep *CNN* secara menyeluruh.
- b. Penelitian ini hanya akan memfokuskan pada teknik deteksi wajah, deteksi landmark pada wajah dan pengenalan wajah pada gambar yang diberikan. Pada penelitian ini tidak akan membahas aspek *computer vision* lainnya seperti deteksi objek, prediksi jenis kelamin, estimasi usia, emosi pada wajah dan lain sebagainya, walaupun pada *library face-api.js* ini memiliki *class* untuk memprediksi jenis kelamin, usia dan juga emosi pada wajah. Dalam penelitian ini hal tersebut tidak akan menjadi fokus utama.

Dengan menetapkan cakupan dan batasan dalam penelitian, studi ini dapat lebih fokus dan terarah dalam mencapai tujuannya untuk memahami teknik *CNN* dan mengembangkan solusi pengenalan wajah yang aman, fleksibel, cepat dan dapat diandalkan, juga dapat diakses melalui halaman web.

1.4. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah sebagai berikut:

- a. Memahami konsep *AI* dan *Machine Learning (ML)* khususnya pada konsep dan cara kerja teknik *Convolutional Neural Network (CNN)*.
- b. Mengimplementasikan *library face-api.js* ke dalam halaman *web* agar *web* tersebut dapat melakukan pengenalan wajah pada gambar yang diberikan.
- c. Mengatasi permasalahan serangan *spoofing* atau penipuan pada saat sistem melakukan proses pengenalan wajah.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini nantinya dapat dikembangkan ke berbagai aplikasi sesuai kebutuhan yang memerlukan teknologi pengenalan wajah di dalamnya. Beberapa contoh dari manfaat penelitian ini adalah sebagai berikut:

- a. Peningkatan pemahaman pada konsep *AI* dan *Machine Learning (ML)* khususnya pada konsep dan cara kerja teknik *Convolutional Neural Network (CNN)*.
- b. Dapat dikembangkan ke level produksi misal dikembangkan ke sektor pendidikan untuk absensi atau ke sektor pengawasan dan keamanan publik, seperti membantu mendeteksi wajah individu yang menjadi buronan hukum.
- c. Dan lain sebagainya dengan yang membutuhkan teknologi *Face Recognition*.

Dengan manfaat penelitian ini, diharapkan dapat memberikan wawasan terhadap manfaat dari penelitian ini dan memberikan dampak positif dalam meningkatkan keamanan, efisiensi, dan kenyamanan diberbagai sektor kehidupan. Dan contoh-contoh manfaat penelitian di atas tentu saja hanya beberapa dari banyaknya potensi aplikasi dan pengembangan teknologi pengenalan wajah ini ke dalam berbagai sektor. Masih banyak potensi lain yang dapat dijelajahi dan dikembangkan dalam penelitian pengenalan wajah ini di masa depan.

1.6. Metodologi Penelitian

- a. Studi Literatur

Tahap awal penelitian akan melibatkan studi literatur untuk memahami konsep *Machine Learning (ML)*, khususnya pada teknik *Convolutional Neural network (CNN)*, *TensorFlow*, *library face-api.js*, dan Pengolahan Citra. Informasi dan pengetahuan yang diperoleh dari literatur akan menjadi dasar untuk merumuskan kerangka teori penelitian.

- b. Pengumpulan Data

Data yang diperlukan dalam penelitian ini adalah dataset wajah untuk disimpan pada sistem sehingga sistem dapat melakukan pengenalan pada citra wajah yang disimpan. Dataset wajah dapat diambil dari sumber publik yang

tersedia atau disediakan oleh institusi terkait, juga data wajah ini dapat diambil melalui peserta penelitian yang bersedia berpartisipasi.

c. Implementasi

Sistem pengenalan wajah akan diimplementasikan menggunakan *library face-api.js*. Pengenalan wajah akan mencakup tahap deteksi wajah, deteksi landmark wajah dan pengenalan wajah. Selain itu, strategi penyimpanan hasil *komputasi* ketika upload wajah ke database akan diimplementasikan untuk mengatasi pengulangan *komputasi* pada halaman pengenalan wajah.

d. Pengujian

Setelah *library face-api.js* di implementasikan pada halaman web, dan data wajah berhasil dikumpulkan, maka selanjutnya akan dilakukan pengujian terhadap sistem seperti memasukan data-data wajah ke dalam sistem dan dilakukan pengenalan wajah dengan gambar wajah yang belum di lihat sebelumnya.

e. Penanganan Tantangan Spoofing

Untuk mengatasi tantangan *spoofing* atau upaya penipuan dalam pengujian model, *library Silent-Face-Anti-Spoofing* yang dikembangkan dengan bahasa pemrograman Python akan diterapkan dan diuji.

f. Kesimpulan dan Rekomendasi

Setelah mengevaluasi hasil pengujian dan mengatasi masalah atau tantangan tersebut, kesimpulan akan diambil mengenai keberhasilan teknologi pengenalan wajah yang diimplementasikan dengan *library face-api.js* ini. Rekomendasi juga akan diberikan untuk pengembangan lebih lanjut dan penerapan solusi pengenalan wajah ini dalam berbagai bidang aplikasi.

1.7. Sistematika Penelitian

Agar laporan tugas akhir ini lebih terstruktur dan mudah dipahami oleh pembaca, penulis membuat ringkasan sistematika laporan tugas akhir sebagai berikut:

BAB I PENDAHULUAN

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

BAB II LANDASAN TEORI

Pada bab ini berisi tentang landasan teori yang digunakan untuk membangun sistem pengenalan wajah, mencakup penjelasan tentang Pengolahan Citra atau *Image Processing*, *Artificial Intelligence (AI)*, *Machine Learning (ML)*, *Neural Network (NN)*, *Convolutional Neural Network (CNN)*, *Deep Learning (DL)*, *TensorFlow (TF)*, dan *library face-api.js*.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Bab ini berisi analisis sistem dan perancangan untuk sistem pengenalan yang mencakup berbagai diagram, seperti diagram gambaran umum sistem, diagram proses upload citra wajah, diagram proses pengenalan wajah pada citra dan diagram *flow* dari teknik *CNN*.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dibahas mengenai implementasi sistem yang telah dirancang dan data-data yang perlu dipersiapkan untuk membangun sistem. Selain itu, pada bab ini juga akan dijelaskan mengenai proses pengujian yang dilakukan untuk mengevaluasi kinerja sistem.

BAB V PENUTUP

Bab ini akan membahas kesimpulan dari hasil penelitian, dan saran-saran untuk pengembangan penelitian selanjutnya.

BAB II

LANDASAN TEORI

2.1. Pengolahan Citra atau Image Processing



Fig. 1 Visualisasi Citra Digital

Pengolahan citra atau *image processing* adalah proses atau serangkaian teknik untuk memanipulasi gambar atau citra digital dengan tujuan meningkatkan kualitas, mengubah karakteristik, atau mengekstrak informasi tertentu dari citra tersebut [1] [2]. Pada pengolahan citra, ekstraksi gambar merupakan tahap inti dari pengolahan citra itu sendiri dan pada tahap ekstraksi gambar juga merupakan kunci dalam pengenalan wajah atau analisis citra pada wajah secara umum.

Citra digital pada gambar sebenarnya, jika dilihat lebih mendalam, hanyalah sebuah kumpulan titik-titik (*pixel*) yang tersusun dengan variasi warna pada setiap *pixelnya*. Warna pada *pixel* direpresentasikan sebagai angka, dan angka inilah yang dapat diolah untuk memanipulasi gambar seperti yang sudah disebutkan sebelumnya. [1][2]

Pada citra khususnya citra RGB, gambar tersebut terdiri dari tiga komponen warna yang mewakili kedalaman warna yang diwakili oleh angka 0 hingga 255. R yaitu untuk warna merah atau red, dari 0 untuk hitam sampai 255 yaitu merah, G untuk hijau atau green, dari 0 yaitu hitam sampai 255 yaitu biru. Ketika ketiga komponen ini jika digabungkan, mereka akan membentuk warna baru yang diinginkan, seperti mencampur warna antara biru dan kuning maka akan membuat warna baru yaitu hijau. [1] [2] [3] [4]

2.2. Artificial Intelligence (AI)

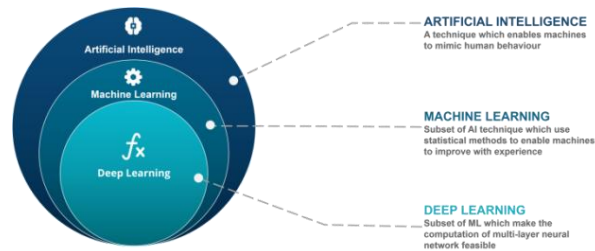


Fig. 2 Paradigma Kecerdasan Buatan

Artificial Intelligence (AI) atau Kecerdasan Buatan adalah bidang ilmu komputer atau *computer science* yang bertujuan untuk mengembangkan sistem komputer yang mampu melakukan tugas-tugas yang biasanya memerlukan kecerdasan manusia. Dalam konteks ini, tujuan utamanya adalah menciptakan mesin yang dapat berpikir cerdas, mampu belajar, merencanakan, dan menyelesaikan masalah dengan cara yang mirip dengan kemampuan manusia.^{[1][5]}

Tidak semua program atau aplikasi merupakan bagian dari pengembangan *AI*. Terdapat perbedaan antara pemrograman non-*AI* (pemrograman tradisional) dengan penerapan teknologi *AI* ^{[5][6]}. Perbedaan tersebut dijelaskan pada penjelasan berikutnya yaitu tentang *Machine Learning (ML)*.

2.3. Machine Learning (ML)



Fig. 3 Perbedaan Tradisional & ML Pemrograman

Machine Learning (ML) secara singkat adalah sebuah pendekatan untuk mencapai *AI* atau untuk mencapai kecerdasan secara tiruan untuk sebuah mesin.^[5]

Program *ML* menjadi hebat karena hasil program ini dapat digunakan (dilatih kembali) dengan contoh atau data baru tanpa mengubah kodenya lagi, misal jika kita membuat sebuah program *ML* untuk mengenali kucing, maka dengan pemrograman *ML*, mesin dapat mengenali anjing tanpa seorang programmer

memprogram kembali untuk mengenali anjing, cukup memberikan gambar anjing untuk pelatihan atau data yang berbeda untuk mesin mempelajari (pola data) nya. ^[5]

Dengan penjelasan diatas, maka terdapat 2 macam pemograman dan terdapat perbedaannya, yaitu program *Machine Learning (ML)* dan dengan program yang umumnya mahasiswa belajar tentang pemograman (*traditional programming*). Perbedaan ini yaitu ada pada penggunaan kode atau program untuk di masa depan. Maksudnya, *traditional programming* di gunakan hanya untuk beberapa logika yang programmer sudah atur, sedangkan *ML programming* diatur untuk menemukan sebuah pola dari data sehingga mesin dapat belajar dari data yang diberikan tersebut. Contohnya, misal dengan deteksi spam pada email, dengan *tradisional programming* mungkin akan banyak sebuah logika atau pencarian untuk memeriksa apakah suatu kata dikaitkan dengan spam atau tidak. Jika ya, maka kita akan atur di program kita dengan true (misalnya) untuk memblokir email tersebut. Namun, pelaku spam mungkin akan memahami hal ini, dengan mengubah kata sedikit, dan sistem akan tembus (tidak terdeteksi spam). Dengan demikian, hal ini akan Tarik-menarik antara spammer dan programmer, dan akan membuang-buang waktu. Sekarang, kita dapat menggunakan logika *ML* untuk mengatasi masalah ini, dengan banyaknya pengguna menandai email sebagai spam, mesin akan secara otomatis mengetahui kata atau fitur apa yang paling mungkin berkontribusi pada email spam. Dengan demikian tidak ada lagi manusia yang harus terlibat untuk memelihara daftar secara manual. ^[5]

Pada pemograman *ML* terdapat banyak metode untuk mencapai *AI*, salah satu teknik populer dan membuat *AI* menjadi menarik adalah *Artificial Neural Network (ANN)* ^[7], yang akan dibahas pada bagian berikutnya.

2.4. Neural Network (NN)

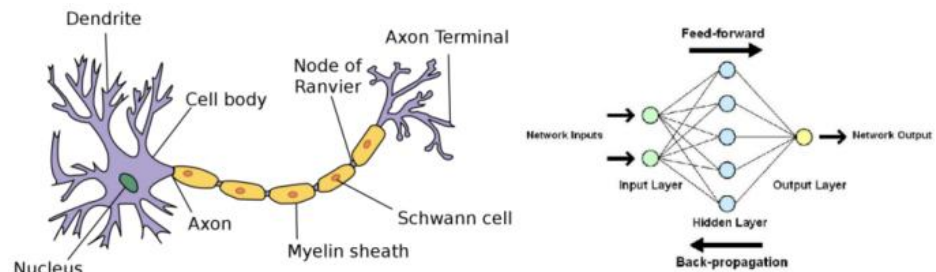


Fig. 4 Visualisasi Implementasi Saraf Otak Pada Manusia Pada Neural Network

Neural Network (NN) dikenal juga sebagai *Artificial Neural Network (ANN)* atau dalam bahasa Indonesia yaitu Jaringan Saraf Tiruan, merupakan cabang dari *Machine Learning (ML)*, nama dan struktur dari *ANN* terinspirasi dari otak manusia, yang meniru cara neuron biologis memberi sinyal dari satu neuron ke neuron lainnya. [1] [7] [8] [9]

Artificial Neural Network (ANN) terdiri dari lapisan simpul atau *node*, yang berisi lapisan masukan (*input layer*), lapisan tersembunyi (*hidden layer*) yang terdiri dari satu atau lebih, dan lapisan keluaran (*output layer*). Setiap *node* atau *neuron* terhubung ke yang lain dan memiliki bobot dan ambang (*threshold*) yang ditentukan. Jika output (setelah melalui proses fungsi aktivasi) dari setiap *node* berada di atas nilai ambang yang ditentukan, *node* tersebut diaktifkan, dan mengirimkan data ke lapisan jaringan berikutnya. Jika tidak lebih dari ambang yang ditentukan, maka tidak ada data yang diteruskan ke lapisan jaringan berikutnya. [9]

Neural Network (NN) mengandalkan banyaknya data pelatihan untuk meningkatkan akurasi. Namun, meskipun jumlah data pelatihan yang terbatas, *NN* tetap dapat memberikan akurasi yang tinggi jika disesuaikan melalui pengaturan algoritma pembelajaran. Oleh karena itu, algoritma ini menjadi alat yang kuat dalam ilmu komputer dan kecerdasan buatan, mampu mengklasifikasikan dan mengelompokkan data dengan kecepatan tinggi. Tugas-tugas seperti pengenalan ucapan atau pengenalan gambar dapat diselesaikan dalam hitungan menit dibandingkan dengan pengidentifikasian manual oleh para ahli manusia yang dapat memakan waktu hingga berjam-jam. Salah satu jaringan saraf yang paling terkenal adalah algoritma pencarian Google. [9]

2.4.1. Cara Kerja Neural Network (NN)

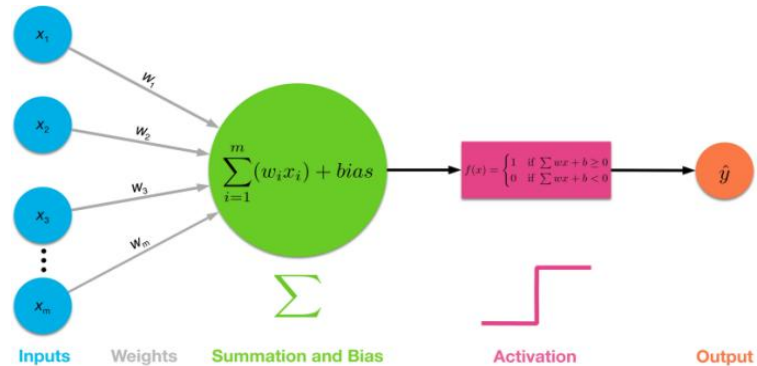


Fig. 5 Visualisasi Perhitungan Untuk ke 1 Neuron Berikutnya

Rumus secara matematis:

$$z = \sum_{i=1}^m w_i x_i + bias$$

$$H(z) = \begin{cases} 0 & z < 0 \\ \frac{1}{2} & z = 0 \\ 1 & z > 0 \end{cases}$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Rumus secara program:

$$\sum w_i x_i + bias = w_1 x_1 + w_2 x_2 + w_3 x_3 + bias$$

$$output = f(x) = 1 \text{ if } \sum w_1 x_1 + b \geq 0; 0 \text{ if } \sum w_1 x_1 + b < 0$$

$$activation = 1/(1 + e^{-x})$$

Fig. 6 Rumus Perhitungan

Setelah lapisan masukan (*input layer* - x) ditentukan sebelumnya, maka bobot (*weight* - w) diberikan pada setiap koneksi. Bobot ini membantu menentukan seberapa besar pengaruh setiap input (x_1, x_2, x_3, \dots) terhadap output dari *neuron* tersebut. Dalam langkah ini, nilai input (x) dikalikan dengan bobot (w) untuk setiap koneksi, sehingga memberikan "kekuatan" atau "nilai penting" yang berbeda pada setiap input, sesuai dengan bobot yang diberikan.^[9]

Setelah dilakukan penjumlahan bobot dengan nilai inputnya, maka hasil penjumlahan tersebut ditambahkan dengan nilai *bias* ^[9]. *Bias* adalah parameter tambahan pada setiap neuron yang memungkinkan *neuron* untuk

memiliki nilai ambang (*threshold*) tertentu sehingga dapat mempengaruhi output *neuron* dan agar tidak mendapatkan nilai 0.^[1]

Setelah proses penjumlahan dan penambahan dengan nilai *bias* dilakukan, hasilnya akan diaplikasikan pada fungsi aktivasi ($f(x)$). Fungsi aktivasi bertugas untuk menentukan apakah *neuron* tersebut diaktifkan (output 1) atau tidak diaktifkan (output 0) berdasarkan hasil dari $\sum w_i x_i + \text{bias}$.^[9]

Dengan menggunakan rumus $\text{output} = f(x) = 1$ if $\sum w_i x_i + b \geq 0$; 0 if $\sum w_i x_i + b < 0$, neuron akan mengeluarkan output 1 jika hasil dari $\sum w_i x_i + \text{bias}$ lebih besar atau sama dengan 0, dan mengeluarkan output 0 jika hasil dari $\sum w_i x_i + \text{bias}$ lebih kecil dari 0. Proses ini berlaku untuk setiap neuron pada lapisan berikutnya dalam jaringan saraf.^[9]

Sebagian besar depp *neural network* bersifat feedforward, artinya hanya mengalir dalam satu arah, dari input ke output. Namun, kita juga dapat melatih model melalui *backpropagation*, yaitu bergerak berlawanan arah dari keluaran ke masukan. *Backpropagation* adalah teknik dalam pembelajaran jaringan saraf yang digunakan untuk mengoptimalkan bobot dan *bias* berdasarkan selisih antara output yang dihasilkan oleh jaringan dengan target yang seharusnya.^[9]

Dari penjelasan di atas, sebenarnya masih belum lengkap dan menyeluruh, oleh karena itu, agar pembahasan tentang *ANN* ini lebih jelas dan menyeluruh atau komprehensif, mari kita kerjakan sebuah contoh soal sederhana yang menggambarkan bagaimana *ANN* beroperasi mulai dari input hingga menghasilkan sebuah nilai prediksi.

2.4.1.1. Contoh Soal Neural Network (NN)

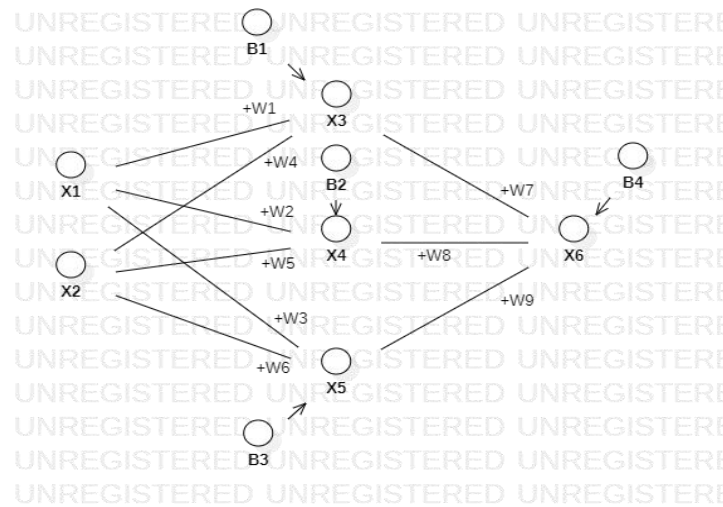


Fig. 7 Visualisasi ANN (Untuk Soal)

Diketahui:

Input:	Bobot /	Bias	Threshold	Target:	Learning	Target	Dataset:
$x_1 = 1$	Weight:	$b_1 = 1$	d:	$t = 0,8$	Rate:	Loss:	$N = 1$
$x_2 = 0$	$w_1 = 0,15$	$b_2 = 1$	$\Theta = 0,5$		$\alpha / \eta = 0,1$	$L = 0,00..$	
	$w_2 = 0,4$	$b_3 = 1$					
	$w_3 = 0,6$						
	$w_4 = 0,1$						
	$w_5 = 0,21$						
	$w_6 = 0,31$						
	$w_7 = 0,5$						
	$w_8 = 0,41$						
	$w_9 = 0,1$						

Table 1 Tabel Soal Diketahui untuk Soal ANN

Pertanyaan: Selesaikan perhitungan *Artificial Neural Network (ANN)* ini hingga nilai fungsi kerugian (*loss function*) mencapai 0, dengan dua angka 0 setelah koma (0,00...) ^[11]. Sertakan juga langkah-langkah dan rumus yang digunakan.

Jawaban: Pada halaman berikutnya.

Langkah 1: Hitung semua keluaran pada lapisan tersembunyi yang pertama (first hidden layer)

Rumus:

$$\sum w_i x_i + \text{bias} = w_1 x_1 + w_2 x_2 + w_3 x_3 + \text{bias}^{[9]}$$

$$\text{output} = f(x) = 1 \text{ if } \sum w_1 x_1 + b \geq \Theta; 0 \text{ if } \sum w_1 x_1 + b < \Theta^{[9]}$$

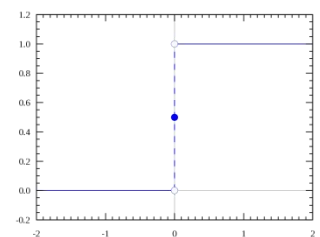
Quick note: Jika diperhatikan dengan lebih teliti pada bagian $\geq \Theta$ dan $< \Theta$, terdapat perbedaan dengan rumus sebelumnya yang dituliskan sebagai 0. Di sana, simbol Θ disebut sebagai threshold, yang dapat diatur menggunakan sebuah variabel jika dalam program. Seperti yang kita ketahui, dalam contoh soal ini, threshold diberikan nilai 0,5. Sehingga nilai pada $\geq \Theta$ mengandung nilai 0,5.

$$x_3 = (x_1 \cdot w_1) + (x_2 \cdot w_4) + b_1$$

$$= (1 \cdot 0,5) + (0 \cdot 0,1) + 1$$

$$= 0,15 + 0 + 1$$

$$= 1,15 \text{ (lebih dari 0,5 maka node/neuron di aktifkan berikan nilai aktivasi)}^{[9]}$$



Rumus:

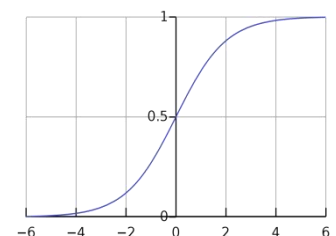
$$1/(1+e^{-x})^{[10]}$$

$$x_3 = 1 / (1 + e^{-x})$$

$$= 1 / (1 + e^{-1,15})$$

$$= 1 / (1 + 0,316063)$$

$$= 0,7602499389 \text{ (ini adalah nilai keluaran untuk } x_3)$$



- Lakukan ke semua node pada lapisan hidden pertama sehingga:

$$x_4 = 0,8021838886$$

$$x_5 = 0,8320183851$$

Langkah 2: Karna pada fase hidden layer hanya di set 1 maka selanjutnya hitung keluaran pada lapisan output

Rumus:

$$\sum w_i x_i + \text{bias} = w_1 x_1 + w_2 x_2 + w_3 x_3 + \text{bias}^{[9]}$$

$$\text{output} = f(x) = 1 \text{ if } \sum w_1 x_1 + b \geq \Theta; 0 \text{ if } \sum w_1 x_1 + b < \Theta^{[9]}$$

$$\begin{aligned} x_3 &= (x_3 \cdot w_7) + (x_4 \cdot w_8) + (x_5 \cdot w_9) + b_4 \\ &= (0,7602499389 \cdot 0,5) + (0,8021838886 \cdot 0,41) + (0,8320183851 \cdot 0,1) + 1 \\ &= 0,3801249694 + 3298913733 + 0,0832018385 + 1 \\ &= 1,7932181792 \text{ (lebih dari 0,5 maka node/neuron di aktifkan berikan nilai aktivasi }^{[9]})} \end{aligned}$$

Rumus:

$$1/(1+e^{-x})$$

$$\begin{aligned} x_3 &= 1 / (1 + e^{-x}) \\ &= 1 / (1 + e^{-1,7932181792}) \\ &= 1 / (1 + 0,16642372557) \\ &= 0,8565322336 \text{ (Ini adalah nilai keluaran untuk } x_6, \text{ di mana } x_6 \text{ merupakan neuron} \\ &\quad \text{output. Oleh karena itu, nilai ini adalah nilai akhir atau prediksi} \\ &\quad \text{dari ANN. Namun, sebelum nilai ini dapat digunakan, prediksi ini} \\ &\quad \text{harus diverifikasi kebenarannya }^{[10]} \text{ dengan menggunakan yang} \\ &\quad \text{disebut fungsi kerugian (loss function))} \end{aligned}$$

Langkah 3: Hitung nilai kerugian (loss function), menggunakan Mean Squared Error (MSE) ^[11]

Rumus:

Rumus secara matematis:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Rumus dalam program:

$$\sum(\text{target-output})^2 / \text{jumlah_data}$$

$$\text{MSE} = (0.8 - 0.8565322336)^2 / 1$$

$$= (-0.0565322336)^2 / 1$$

$$= 0.0031964741 \text{ (note: Secara program atau soal ANN seperti ini, MSE dapat}$$

dijalankan setelah proses pelatihan pada dataset selesai.

Misalnya, jika dalam dataset terdapat 2 data, maka seluruh data harus diselesaikan terlebih dahulu sebelum menghitung MSE ^[11]. Namun, pada contoh soal ini, hanya terdapat 1 data, sehingga pada langkah 3 langsung dihitung nilai loss-nya. Jika terdapat lebih dari 1 data, pada langkah 3 ini harus menghitung data dalam dataset yang diberikan secara berulang seperti pada langkah 1, yang tentunya input x nya akan berbeda)

Langkah 4: Kesimpulan

Karena hasil loss telah sesuai dengan target yang ditentukan, yaitu 0 dengan dua angka 0 di belakang koma ($L = 0,00...$) ^[11], maka secara teknis backpropagation tidak diperlukan. Namun, untuk menjelaskan secara komprehensif, halaman berikut akan menjelaskan perhitungan backpropagation. Sebelumnya, karna disini merupakan tahap kesimpulan, berikut adalah ringkasan hasil prediksi ANN ini:

Epoch: 1x (Satu kali iterasi seluruh dataset)

Output Layer: 0,8565322336 (hasil prediksi)

Loss Function (Mean Squared Error, MSE): 0,0031964741

Dengan nilai MSE sebesar 0,0031964741, dapat disimpulkan bahwa prediksi ANN sudah cukup akurat berdasarkan data masukan ($x_1=1$, $x_2=0$) dan target yang diinginkan ($t=0,8$). Error yang kecil menunjukkan bahwa prediksi ANN mendekati nilai target yang diinginkan dengan baik.

Catatan:

Pada output ANN bisa berupa 1 node (binary classification) seperti contoh soal ini, atau juga bisa diatur dengan lebih dari 1 node (multi-class classification) dan mode lebih dari 1 node ini merupakan mode untuk mengklasifikasikan wajah. Dalam kasus multi-class classification, hasil prediksi kelas diambil dari node dengan nilai terbesar (metode argmax) pada output layer. Secara menyeluruh, perhitungannya tetap sama, tetapi dalam mengklasifikasikan (cara menghitung loss function) lebih baik menggunakan Cross-Entropy Loss. Pembahasan detail tentang Cross-Entropy Loss tidak menjadi fokus di sini walaupun sebenarnya cukup penting. Penelitian ini berfokus pada pemahaman mendasar tentang CNN agar saat berpraktik nanti sudah memiliki dasar dan tidak bingung.

Langkah 5: Ceritanya loss function nya masih jauh dari yang diharapkan, maka lakukan backpropagation (memperbaharui weight/bobot dan biasnya)

- perbarui bobot:

Rumus:

$$w_{\text{baru}} = w_{\text{lama}} + (\text{learning_rate} * (\text{target} - \text{output}) * x_{\text{lama}})^{[11]}$$

$$\begin{aligned} w_9 &= 0,1 + (0,1 * (0,8 - 0,85655322336) * 0,8320183851) \\ &= 0,1 + (0,1 * -0,0565532234 * 0,8320183851) \\ &= 0,1 + -0,0047136090 \\ &= 0,0952863910 \end{aligned}$$

- Lakukan ke semua weigh / bobot yang ada.

- Perbaharui bias:

Rumus:

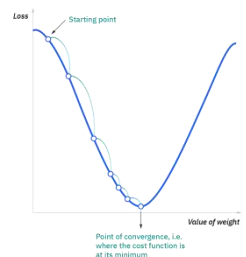
$$b_{\text{baru}} = b_{\text{lama}} + (\text{learning_rate} * (\text{target} - \text{output}))^{[11]}$$

$$\begin{aligned} b_4 &= 1 + (0,1 * (0,8 - 0,85655322336)) \\ &= 1 + (0,1 * -0,0565532234) \\ &= 1 + -0,00565532234 \\ &= 0,99434467766 \end{aligned}$$

- Lakukan ke semua bias yang ada.

Catatan:

Setelah memperbarui semua nilai bias dan weight, langkah selanjutnya adalah mengulangi langkah 1-3 dengan menggunakan parameter yang telah diperbarui, hingga mencapai nilai loss yang diinginkan. Proses ini juga dikenal dengan sebutan "learning" atau proses pembelajaran, di mana model terus memperbaiki dirinya dan menyesuaikan dengan data untuk mencapai performa yang lebih baik.



2.4.2. Jenis-jenis Neural Network (NN)

Jaringan Saraf (*Neural Network*) dapat diklasifikasikan ke dalam berbagai jenis, dan digunakan untuk tujuan yang berbeda. Meskipun ini bukan daftar jenis yang lengkap, di bawah ini akan mewakili jenis jaringan saraf yang paling umum yang akan biasa ditemui untuk kasus penggunaan: ^[9]

a. *Convolutional Neural Network (CNN)*

CNN merupakan fokus utama dalam skripsi atau penelitian ini. Pada bagian berikutnya, akan dijelaskan secara detail mengenai konsep dan cara kerja dari *CNN*. *CNN* dikhususkan untuk tugas-tugas pengenalan pola dalam data berstruktur *grid* atau *matriks*, seperti citra dan video. Arsitektur *CNN* memiliki lapisan *konvolusi* yang berperan dalam mengidentifikasi fitur-fitur penting dari data input. ^[9]

b. *Recurrent Neural Network (RNN)*

RNN memiliki sifat memori, sehingga cocok untuk tugas yang melibatkan data berurutan. *RNN* sering digunakan dalam pengolahan bahasa alami, pemodelan urutan, dan tugas-tugas lain yang melibatkan urutan data. ^[9]

c. *Transformer*

Transformer adalah arsitektur yang revolusioner dalam bidang pemrosesan bahasa alami. *Transformer* menggunakan *mekanisme attention* untuk memahami hubungan antara kata dalam kalimat dan telah menunjukkan performa yang luar biasa dalam tugas-tugas pemodelan bahasa. ^[9]

Perbedaan antara berbagai jenis arsitektur jaringan saraf, seperti *CNN*, *RNN*, *Transformer*, dan jenis lainnya, adalah cara mereka memproses data. Cara mereka mengorganisasi, menghubungkan, dan mengolah informasi dalam jaringan adalah yang membuat mereka unik dan cocok untuk tugas tertentu. ^[1]

2.4.3. Hubungan Neural Network (NN) Dengan Deep Learning (DL)

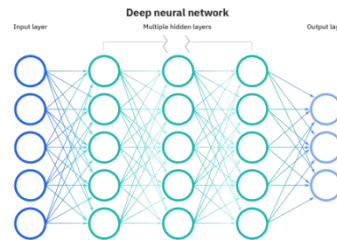


Fig. 8 Visualisasi Jaringan Saraf Tiruan

Deep Learning (DL) merupakan *neural network* dengan lebih dari dua lapisan (*deep layers*). Pada dasarnya *ANN* hanya terdiri tidak lebih dari 2 atau kurang dari 2 *hidden layer*, jika lebih dari 2 *layer* maka biasanya lebih dikenal sebagai *deep neural network* atau *deep learning (DL)* ^[7]. Dengan menggunakan banyak lapisan ini, *DL* memiliki kemampuan untuk mengekstraksi pola dan fitur yang kompleks atau abstrak dari data dengan lebih efisien, sehingga memungkinkan untuk penyelesaian tugas-tugas yang lebih kompleks dan akurat ^[1].

Jadi hubungan antara *ANN* dan *DL* ini hanya terletak pada jumlah lapisan (*hidden layer*) yang digunakan.

2.5. Convolutional Neural Network (CNN)

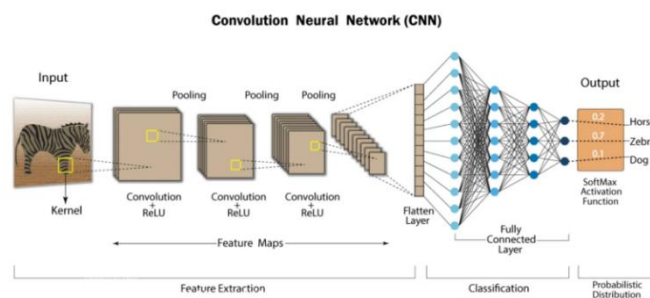


Fig. 9 Visualisasi Proses CNN

Convolutional Neural Network (CNN) adalah jenis khusus dari jaringan saraf (*NN*) atau yang paling umum digunakan dalam tugas pengenalan citra. *CNN* memiliki lapisan-lapisan khusus yang secara otomatis mengekstrak fitur dari gambar dan mengidentifikasi pola dalam hierarki. Lapisan konvolusi dan lapisan *pooling* adalah komponen inti dari *CNN*, yang memungkinkan pemrosesan citra

secara efisien dan mendalam ^[1]. Tujuan tahap konvolusi ini adalah untuk mereduksi dimensi gambar input dengan mengekstrak fitur-fitur penting, sehingga ketika dikirim ke tahap *fully connected*, beban tidak terlalu besar ^[13]. Sebagai contoh, jika kita memiliki gambar input dengan ukuran 1000 pixel x 1000 pixel x 3 saluran RGB, maka jika mengirimkannya langsung ke *ANN* sama saja dengan mengirimkan 3 juta piksel, ini akan menjadi tidak efisien dan bahkan dapat menyebabkan *overfitting* ^[14] ^[15]. Oleh karena itu, tahapan lapisan konvolusi sangat penting untuk mengambil hanya informasi penting dalam gambar.

Dalam proses konvolusi, elemen yang paling penting adalah penggunaan filter, juga dikenal sebagai *kernel*. Filter ini dapat dikonfigurasi secara manual dengan bobot yang telah diteliti sebelumnya, seperti misal penggunaan filter *edge horizontal* atau *vertical detection*, atau jenis filter lainnya ^[16], yang dapat dilihat seperti gambar dibawah ini sebagai contoh. Selain itu, filter ini juga dapat diatur secara acak, sehingga memungkinkan *CNN* untuk memperbarui bobot filter tersebut secara iteratif, dan menghasilkan filter yang mungkin belum pernah ditemukan sebelumnya ^[16].

Fungsi lain filter selain untuk mereduksi gambar agar semakin kecil dengan mengambil hal-hal penting dengan melakukan filtering, sistem filtering ini akan membantu *ANN* dalam mengenali pola atau kombinasi angka nanti dari sebuah gambar yang diberikan ^[1].



Fig. 10 Filter Yang Ditentukan Untuk Mendeteksi Sesuatu



Fig. 11 Filter Hasil Pembelajaran

2.5.1. Cara Kerja Convolutional Neural Network (CNN)

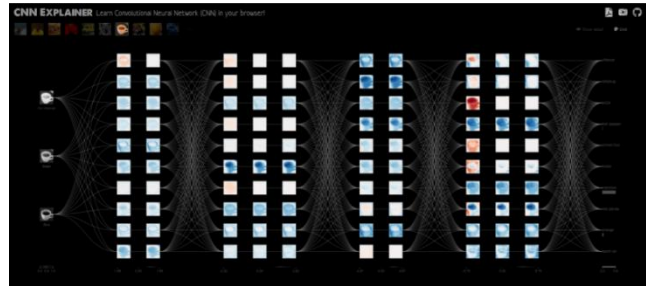


Fig. 12 Visualisasi Konvolusi

Dalam merancang arsitektur *CNN*, kita dapat mengaturnya sesuai dengan kebutuhan dan masalah yang ingin dipecahkan. Pada dasarnya, *CNN* bekerja dengan mengikuti beberapa proses secara hierarkis. Pertama, data input berupa gambar atau citra digital harus diolah sesuai dengan permasalahan yang ada, seperti melakukan cropping terlebih dahulu misalnya untuk memastikan semua piksel memiliki ukuran yang sama. Ini juga sering disebut sebagai *augmentasi data*. Setelah itu, data tersebut dapat dimasukkan ke dalam lapisan konvolusi sebelum dikirimkan ke lapisan *fully connected* untuk klasifikasi ^[1]. Misalnya, jika kita mengatur lapisan konvolusi dengan 4 lapisan konvolusi dengan 2 kali *pooling* pada lapisan ke dua dan lapisan akhir sebelum *flatten*, maka urutannya akan dapat dituliskan seperti berikut: Konvolusi + *ReLU* > Konvolusi + *ReLU* > *Pooling* > Konvolusi + *ReLU* > Konvolusi + *ReLU* > *Pooling* > *Flatten* > Lapisan *Dense*.

Setelah semua proses di lapisan konvolusi selesai, data citra yang terakhir tersebut dapat diteruskan ke lapisan *fully connected* untuk proses klasifikasi. Untuk informasi lebih lanjut tentang proses *fully connected* atau *artificial neural network*, penjelasan tersebut dapat dilihat pada bagian 2.4 hingga 2.4.3.

Untuk penjelasan lebih rinci tentang apa itu *augmentasi data*, konvolusi, *ReLU*, *pooling*, *flatten* dan sampai ke *dense layer*, berikut penjelasannya yang lebih rinci terkait pengertian beserta cara kerjanya tersebut.

a. Input Data dan *Augmentasi Data*

Pertama-tama, dalam tahap konvolusi pada *CNN*, *CNN* menerima data input dalam bentuk *grid* atau *matriks array* ^[14]. Yang sebelumnya, data ini telah dilakukan *augmentasi*. *Augmentasi data* adalah proses mengubah data pelatihan dengan melakukan transformasi seperti rotasi, pemotongan, dan perubahan warna ^[16]. Tujuannya adalah untuk meningkatkan variasi data pelatihan sehingga model dapat lebih baik dalam mengenali berbagai variasi gambar ^[1].

b. Konvolusi + Aktifasi *ReLU*

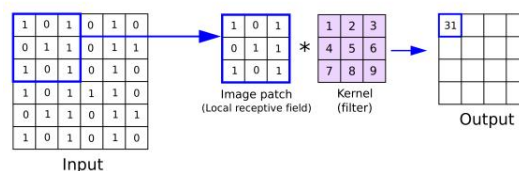


Fig. 13 Visualisasi Kerja Fungsi Konvolusi

Selanjutnya, setelah citra diperoleh, tahap pertama adalah melewati tahap lapisan konvolusi. Pada lapisan ini, terdapat beberapa filter atau *kernel* yang akan diterapkan pada citra ^[1]. Dalam merancang arsitektur *CNN*, filter atau *kernel* ini dapat diatur secara manual dengan bobot yang telah diteliti sebelumnya, seperti penggunaan filter *edge horizontal detection* atau *edge vertikal detection*, atau jenis lainnya. Selain itu, filter ini juga dapat diatur secara acak, memungkinkan *CNN* untuk melakukan pembaruan pada bobot filter tersebut, sehingga menghasilkan filter yang mungkin belum pernah ditemukan sebelumnya, seperti yang ditunjukkan dalam gambar di bawah ini. ^[16]



Fig. 14 Visualisasi Filter deteksi yang Didapatkan Dari Hasil Pembelajaran

Di sana, terlihat terdapat beberapa filter yang belum ditemukan sebelumnya salah satunya mungkin dapat diberi nama sebagai filter deteksi kerutan (*wrinkles detection*) dan ditemui juga filter untuk

mendeteksi kata-kata (*words detection*), dan yang paling menarik adalah penemuan filter pada lapisan ke 6 konvolusi untuk mendeteksi wajah selama proses konvolusi ini ^[16], dapat dilihat bahwa dalam gambar di atas, tangan dan objek lainnya diabaikan, sementara wajah diberi penekanan dengan mengubah warnanya menjadi putih. Hal-hal ini sebenarnya tidak ditentukan sebelumnya, melainkan ditemukan oleh *CNN* selama proses pembelajaran melalui proses *backpropagation*.

Pada tahapan lapisan konvolusi ini, filter tersebut agar dapat mengubah dengan gambar baru atau melakukan filtering yaitu dengan cara digeser secara berulang-ulang di seluruh saluran gambar input dengan *stride* yang ditentukan ^[12]. *Stride* adalah salah satu parameter dalam operasi konvolusi pada *CNN* yang mengontrol seberapa jauh *kernel* (filter) bergerak melintasi gambar saat melakukan konvolusi. Dalam konvolusi dengan *stride* 1, *kernel* akan bergerak satu langkah (pixel) pada setiap iterasi ^[12]. Setiap filter akan mengidentifikasi pola atau fitur-fitur tertentu pada citra dengan mengalikan nilai-nilai piksel pada citra dengan nilai bobot pada filter. Proses konvolusi ini menghasilkan peta fitur (*feature map*) ^[12].

Filter atau *kernel* ini dapat disesuaikan sesuai dengan kebutuhan. Penting untuk diingat bahwa semakin banyak filter yang digunakan, semakin banyak fitur abstrak yang dapat diidentifikasi dan tentunya akan membuat pada tahapan *dense layer* mempermudah dalam mengenali pola. Namun, peningkatan jumlah filter juga dapat meningkatkan beban komputasi, dan bahkan berpotensi menyebabkan *overfitting*. ^[1]

Setelah filter bergerak melintasi seluruh gambar, hasil konvolusi tersebut akan lebih baik jika diikuti oleh fungsi aktivasi *ReLU*. Fungsi ini memetakan nilai-nilai piksel negatif menjadi 0 dan mempertahankan nilai positif. Hal ini membantu dalam menghadirkan non-linearitas dan pemodelan fitur-fitur yang lebih kompleks ^[1]. Untuk lebih memahami visualisasi konsep *ReLU*, dapat dilihat pada gambar di bawah ini.

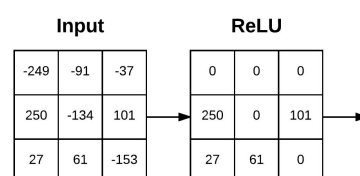


Fig. 13 Visualisasi Perhitungan ReLU

c. Pooling

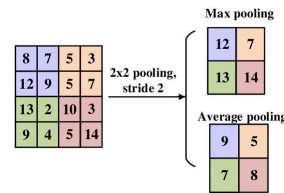


Fig. 16 Visualisasi Fungsi Pooling

Citra-citra fitur yang telah melalui lapisan konvolusi + aktivasi *ReLU*, selanjutnya akan melewati lapisan *pooling*. Pada lapisan ini, citra-citra fitur akan dikurangi ukurannya dengan melakukan operasi seperti mengambil nilai maksimum (*max pooling*) atau dengan mengambil nilai rata-rata (*average pooling*).^[13]

d. Flatten

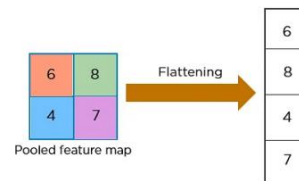


Fig. 17 Visualisasi Fungsi Flatten

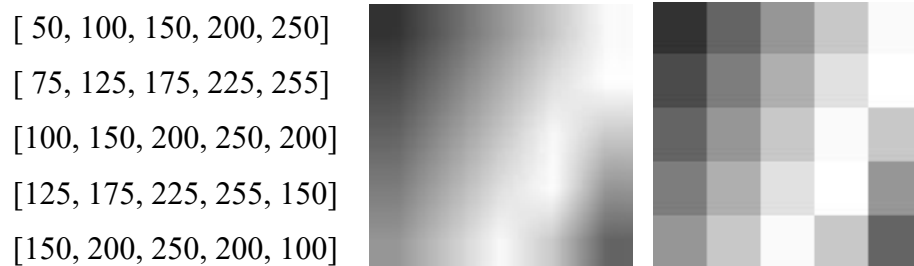
Setelah beberapa lapisan konvolusi dan *pooling* yang ditentukan, peta fitur akan di-flatten menjadi *vektor 1d* untuk dihubungkan dengan lapisan *Dense* atau *Fully Connected Layer*. Hal ini diperlukan karena lapisan *Dense* membutuhkan input berupa *vektor*, bukan *matriks*.^[13]

Setelah melalui tahap-tahap sebelumnya, hasilnya akan dikirimkan ke lapisan *Fully Connected* atau *Dense Layer* atau *Artificial Neural Network (ANN)* sebagai input dari fungsi *flatten* sebelumnya^[13]. Dalam praktiknya, konfigurasi *CNN* akan bervariasi. Jumlah lapisan konvolusi, filter, dan *neuron* pada lapisan *Dense* dapat disesuaikan dengan kompleksitas masalah dan ketersediaan data latihan^[1]. Selain itu, penggunaan *dropout* juga bisa dipertimbangkan untuk mengurangi *overfitting*^[17]. Fungsi *loss* serta *optimizer* juga dipilih sesuai dengan jenis tugas yang dihadapi.^[1]

2.5.1.1. Contoh Soal Convolutional Neural Network (CNN)

a. Contoh Soal 1 (Citra Grayscale)

Selesaikan pengerjaan CNN berikut, namun kerjakan hanya sampai tahap lapisan fungsi flatten saja, tidak perlu sampai ke lapisan fully connected. Diketahui input gambar RGB setelah dilakukan augmentasi data, dilakukan seperti cropping menjadi 5x5 dan konversi ke citra grayscale, input citra menjadi seperti berikut:



Lakukan pengerjaan dengan settingan lapisan konvolusi sebagai berikut:

- Menggunakan 2 lapisan konvolusi dengan 1x pooling (max pooling) pada lapisan akhir konvolusi sebelum lapisan flatten.
- Di lapisan pertama konvolusi (hanya pada lapisan pertama) gunakan padding (zero padding) 2 pixel.
- Gunakan 5 filter dengan stride 1.
- Dan terakhir, untuk operasi pooling gunakan ukuran 2x2 dengan stride 2.

Pada contoh soal ini, filter digunakan dengan nilai yang ditentukan (tanpa proses pembelajaran). Berikut adalah filter yang akan digunakan pada setiap lapisan konvolusi:

- Lapisan Konvolusi 1

- Filter 1

Nama: Filter rata-rata atau Filter penghalus (Average or Smoothing Filter)

Fungsi: Menghaluskan gambar dengan merata-ratakan intensitas piksel di sekitarnya. Filter ini digunakan untuk mengurangi noise dan menghasilkan gambar yang lebih halus.

Matriks Filter:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- Filter 2

Nama: Filter Laplacian atau Filter deteksi tepi (Laplacian or Edge Detection Filter)

Fungsi: Mendeteksi tepi dalam gambar dengan menyoroti perubahan tajam dalam intensitas warna. Filter ini umumnya digunakan untuk deteksi tepi.

Matriks Filter:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

- Filter 3

Nama: Filter Laplacian Tinggi atau Filter deteksi tepi tajam (High Laplacian or Sharpening Edge Detection Filter)

Fungsi: Sama seperti Filter Laplacian, tetapi lebih menyoroti perubahan yang lebih tajam dalam intensitas warna, sehingga digunakan untuk mempertajam gambar.

Matriks Filter:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Filter 4

Nama: Filter Sobel atau Filter deteksi tepi Sobel (Sobel Edge Detection Filter)

Fungsi: Mendeteksi tepi dalam gambar dengan menghitung gradien intensitas warna. Filter Sobel digunakan untuk mendeteksi tepi dan fitur dalam gambar dengan lebih baik.

Matriks Filter:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

- Filter 5

Siahsan buat atau mencoba bereksperimen atau jika tidak silahkan cari sendiri di internet filter yang ingin anda gunakan, jangan lupa berikan nama dan fungsi dari filter yang anda pilih filter tersebut berguna untuk apa.

- Lapisan Konvolusi 2

Menggunakan filter yang sama seperti pada Lapisan Konvolusi 1.

b. Contoh Soal 2 (Citra RGB)

Semua konfigurasi, seperti lapisan konvolusi, filter, pooling, dan konfigurasi lainnya, semuanya sama seperti dalam contoh soal 1, perbedaannya hanya pada input masukannya sebagai berikut:

- Saluran Red (R)

[255, 0, 0, 255]

[0, 0, 0, 0]

[255, 0, 0, 255]

[0, 0, 0, 0]
- Saluran Green (G):

[0, 255, 0, 0]

[255, 0, 255, 0]

[0, 255, 0, 0]

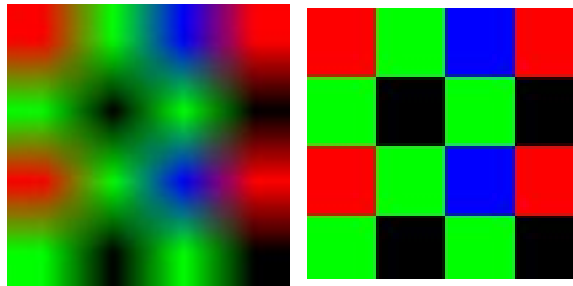
[255, 0, 255, 0]
- Saluran Blue (G):

[0, 0, 255, 0]

[0, 0, 0, 0]

[0, 0, 255, 0]

[0, 0, 0, 0]



Jawaban: Pada halaman berikutnya.

Langkah 3: Selanjutnya melewati lapisan ReLU, yaitu dengan mengubah nilai negatif menjadi 0, maka menjadi:

[[50, 150, 300, 450, 600, 450, 250]

[125, 350, 675, 975, 1255, 930, 505]

[225, 600, 1125, 1575, 1905, 1380, 705]

[300, 750, 1350, 1780, 1935, 1335, 605]

[375, 900, 1575, 1905, 1830, 1155, 450]

[275, 650, 1125, 1305, 1180, 705, 250]

[150, 350, 600, 650, 550, 300, 100]]

Input			ReLU		
-249	-91	-37	0	0	0
250	-134	101	250	0	101
27	61	-153	27	61	0

Langkah 4: Tambahan, karna sebenarnya angka citra yang valid itu adalah rentang dari 0 hingga 255, maka citra akhir konvolusi menjadi:

[[50, 150, 255, 255, 255, 255, 250]

[125, 255, 255, 255, 255, 255, 255]

[225, 255, 255, 255, 255, 255, 255]

[255, 255, 255, 255, 255, 255, 255]

[255, 255, 255, 255, 255, 255, 255]

[255, 255, 255, 255, 255, 255, 250]

[150, 255, 255, 255, 255, 255, 100]]



Langkah 5: Selanjutnya melewati lapisan pooling, namun pada lapisan konvolusi 1, konfigurasi menyebutkan tidak dilakukan pooling, maka konvolusi untuk filter 1 selesai.

Langkah 6: Lakukan konvolusi menggunakan *filter 2*, maka menjadi:

[[0, -50, -100, -150, -200, -250, 0]

[-50, 25, 75, 125, 175, 545, -250]

[-75, 25, 0, 0, 20, 345, -255]

[-100, 50, 0, 0, 120, 145, -200]

[-125, 75, 0, 20, 195, 45, -150]

[-150, 275, 225, 375, 195, 50, -100]

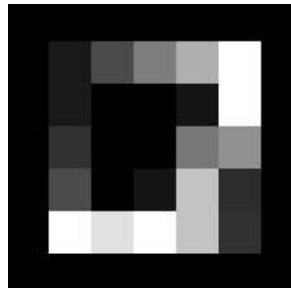
[0, -150, -200, -250, -200, -100, 0]]

Langkah 7: Melewati lapisan ReLU, maka menjadi:

```
[[0, 0, 0, 0, 0, 0, 0]
[0, 25, 75, 125, 175, 545, 0]
[0, 25, 0, 0, 20, 345, 0]
[0, 50, 0, 0, 120, 145, 0]
[0, 75, 0, 20, 195, 45, 0]
[0, 275, 225, 375, 195, 50, 0]
[0, 0, 0, 0, 0, 0, 0]]
```

Langkah 8: Ubah ke format yang valid, maka menjadi:

```
[[0, 0, 0, 0, 0, 0, 0]
[0, 25, 75, 125, 175, 255, 0]
[0, 25, 0, 0, 20, 255, 0]
[0, 50, 0, 0, 120, 145, 0]
[0, 75, 0, 20, 195, 45, 0]
[0, 255, 225, 255, 195, 50, 0]
[0, 0, 0, 0, 0, 0, 0]]
```

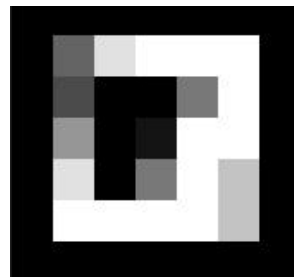


Langkah 9: Selanjutnya melewati lapisan pooling, namun pada lapisan konvolusi 1, konfigurasi menyebutkan tidak dilakukan pooling, maka konvolusi untuk filter 1 selesai.

Langkah 10: Terus lanjutkan sampai semua filter mendapatkan hasil akhir, sehingga:

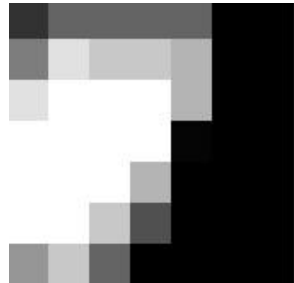
Filter 3:

```
[[0, 0, 0, 0, 0, 0, 0]
[0, 100, 225, 255, 255, 255, 0]
[0, 75, 0, 0, 120, 255, 0]
[0, 150, 0, 20, 255, 255, 0]
[0, 225, 0, 120, 255, 195, 0]
[0, 255, 255, 255, 255, 195, 0]
[0, 0, 0, 0, 0, 0, 0]]
```



Filter 4:

[[50, 100, 100, 100, 100, 0, 0]
 [125, 225, 200, 200, 180, 0, 0]
 [225, 255, 255, 255, 180, 0, 0]
 [255, 255, 255, 255, 5, 0, 0]
 [255, 255, 255, 180, 0, 0, 0]
 [255, 255, 200, 80, 0, 0, 0]
 [150, 200, 100, 0, 0, 0, 0]]

Filter 5:Nama:

Filter Gaussian atau Gaussian Blur Filter.

Fungsi:

Filter Gaussian digunakan untuk menghaluskan gambar dengan merata-ratakan intensitas piksel di sekitarnya, namun dengan bobot yang lebih besar pada piksel di tengah dan bobot yang semakin berkurang saat menjauh dari piksel tengah. Hal ini menghasilkan efek penghalusan yang lebih lembut dan alami. Filter ini berguna untuk mengurangi noise dalam gambar dan menciptakan efek bokeh pada fotografi.

Matriks filter:

[[1, 2, 1],
 [2, 4, 2],
 [1, 2, 1]]

Hasil akhir:

[[50, 200, 255, 255, 255, 255, 250]
 [175, 255, 255, 255, 255, 255, 255]
 [255, 255, 255, 255, 255, 255, 255]
 [255, 255, 255, 255, 255, 255, 255]
 [255, 255, 255, 255, 255, 255, 255]
 [255, 255, 255, 255, 255, 255, 255]
 [150, 255, 255, 255, 255, 255, 100]]



- Lapisan Konvolusi 2

Konfigurasi: tidak ada padding, 4 filter ditentukan | 1 filter custom | stride 1, max pooling | stride 2.

Note: Pada lapisan konvolusi kedua ini, lapisan ini akan menerima 5 input karena menggunakan 5 filter dari lapisan sebelumnya. Oleh karena itu, dalam proses filtering di lapisan ini, setiap filter akan menghasilkan 5 output, yang secara total akan menghasilkan 25 hasil. Hasil-hasil ini juga dapat disebut sebagai peta fitur (feature map).

Langkah 1: Lakukan konvolusi menggunakan filter 1 input 1, maka menjadi:

[[1825, 2190, 2295, 2295, 2290]
[2135, 2295, 2295, 2295, 2295]
[2265, 2295, 2295, 2295, 2295]
[2295, 2295, 2295, 2295, 2290]
[2190, 2295, 2295, 2295, 2135]]

Langkah 2: Lakukan ReLU function:

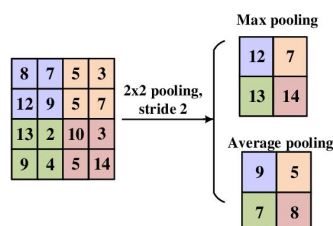
[[1825, 2190, 2295, 2295, 2290]
[2135, 2295, 2295, 2295, 2295]
[2265, 2295, 2295, 2295, 2295]
[2295, 2295, 2295, 2295, 2290]
[2190, 2295, 2295, 2295, 2135]]

Langkah 3: Ubah ke format yang valid, maka menjadi:

[[255, 255, 255, 255, 255]
[255, 255, 255, 255, 255]
[255, 255, 255, 255, 255]
[255, 255, 255, 255, 255]
[255, 255, 255, 255, 255]]

Langkah 4: Selanjutnya melewati lapisan pooling, (max pooling, 2x2, stride 2), maka menjadi:


[[255, 255]
[255, 255]]



Langkah 5: Selanjutnya untuk operasi filter lainnya, prosesnya sama saja mulai dari konv, relu sampai pooling. Maka untuk mempersingkat penulisan berikut adalah hasil dari semua filter setelah melewati operasi sampai proses pooling menggunakan max pooling sesuai dengan konfigurasi CNN:


Filter 2, input 1:

[[235, 0]
[0, 0]]




Filter 3, input 1:

[[255, 0]
[30, 0]]



Filter 4, input 1:

[[255, 0]
[30, 0]]



Filter 5, input 1:


[[255, 255]
[255, 255]]

Filter 1, input 2:

[[255, 255]
[255, 255]]

Filter 2, input 2:

[[150, 255]
[100, 255]]




Filter 3, input 2:

[[255, 255]
[255, 255]]

Filter 4, input 2:

[[75, 255]
[225, 255]]



Filter 5, input 2:

[[255, 255]
[255, 255]]

Karena proses ini cukup memakan waktu dan tenaga jika dilakukan secara manual, maka dirasa cukup sampai ini.

Filter 1, input 3:

Filter 2, input 3:

Filter 3, input 3:

Filter 4, input 3:

Filter 5, input 3:

Filter 1, input 4:

Filter 2, input 4:

Filter 3, input 4:

Filter 4, input 4:

Filter 5, input 4:

Filter 1, input 5:

Filter 2, input 5:

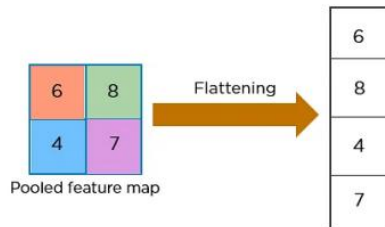
Filter 3, input 5:

Filter 4, input 5:

Filter 5, input 5:

Langkah 6 terakhir: Setelah dilapisan akhir konvolusi, hasil citra yang akhir (dalam contoh kasus ini berarti hasil akhirnya adalah yang hasil pooling) maka langkah selanjutnya hasil-hasil pooling tersebut harus di ubah dan disatukan ke dalam

bentuk vektor 1 dimensi ^[13] atau bisa disebut juga fungsi flattening, berikut visualisasi gambar dari fungsi flatten:



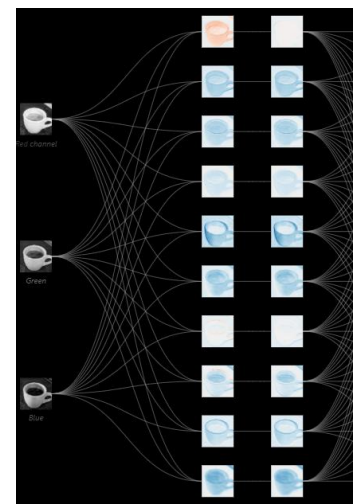
Dan jika dari contoh soal ini maka hasil nya dapat di tuliskan seperti berikut ini, yang mana hasil ini akan diberikan ke input layer Artificial Neural Network (ANN) nanti:

[235, 0, 0, 0, 255, 0, 30, 0, 255, 0, 30, 0, 255, 255, 255, 255, 255, 255, 255, 150, 255, 100, 255, 255, 255, 255, 255, 75, 255, 225, 255, 255, 255, 255, ...]

Dalam membuat kode sendiri, formatnya dapat bervariasi, yang terpenting adalah bahwa kita dapat mengakses data ini dalam bentuk vektor satu dimensi.

Jawaban Soal Contoh 2 (Citra RGB)

Untuk jawaban contoh soal 2, penulis memutuskan untuk tidak menyelesaikan prosesnya secara rinci. Prinsip perhitungannya tetap sama, hanya ada perbedaan dalam input awal yang terdiri dari tiga saluran (R, G, dan B). Oleh karena itu, prosesnya hampir mirip dengan contoh soal 1 pada bagian konvolusi lapisan kedua, di mana setiap filter akan menerima semua saluran input. Sebagai contoh, jika ada 5 filter, maka setiap filter akan menghasilkan 3 output yang mewakili hasil dari masing-masing saluran input.



2.6. TensorFlow (TF)

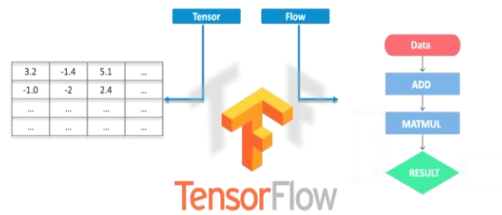


Fig. 18 Tensor & Flow

“TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. - Wikipedia”

TensorFlow (TF) adalah sebuah framework (kerangka kerja) open-source gratis yang dikembangkan oleh Google. Awalnya, *TensorFlow* digunakan secara internal oleh perusahaan Alphabet, yang merupakan induk perusahaan Google. Pada awalnya, *TensorFlow* digunakan untuk memenuhi kebutuhan internal perusahaan Alphabet. Namun, pada tahun 2015, *TensorFlow* dirilis untuk publik. Pada saat itu, penggunaan *TensorFlow* dalam *machine learning* lebih banyak menggunakan bahasa pemrograman *Python*, sehingga *library TensorFlow* di-load dalam bahasa *Python*. Namun, kemudian tim pengembang internal di Google mengembangkan *TensorFlow* untuk *JavaScript*, sehingga dapat digunakan di mana saja di mana *JavaScript* dapat berjalan. Hal ini menghasilkan versi yang dikenal sebagai *TensorFlow.js*.^[18]

2.6.1. Tensor

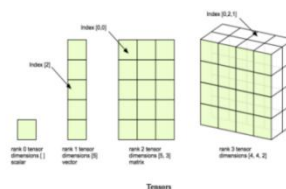


Fig. 19 Tensor

Dalam matematika, tensor adalah objek aljabar yang menggambarkan hubungan multilinear antara himpunan objek aljabar yang terkait dengan ruang vektor. *Tensor* dapat memetakan antara objek yang berbeda seperti *scalar*, *vector*, *matrix*, dan *tensor* lainnya.^[18]

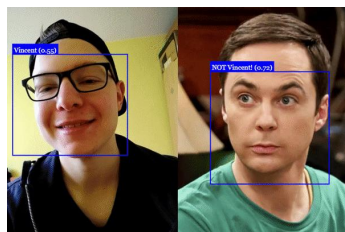
2.7. Library face-api.js

Library face-api.js adalah sebuah *library JavaScript* yang menyediakan fungsi-fungsi pengenalan wajah berbasis web. *Library* ini dikembangkan menggunakan *Tensorflow.js Core*, sehingga memungkinkan pengguna untuk dengan mudah memanggil dan menggunakan kelas-kelas serta fungsi-fungsi yang telah disediakan. Dengan adanya *library face-api.js*, pengembang dapat mengimplementasikan pengenalan wajah secara efisien dalam aplikasi web mereka tanpa perlu membuat kelas dan fungsi-fungsi dari awal. ^[1]

Library ini memanfaatkan teknologi *CNN* untuk mendeteksi, mengenali, dan melacak wajah pada gambar dan video melalui antarmuka yang mudah digunakan. Dalam penelitian ini, *library face-api.js* akan digunakan untuk mengimplementasikan pengenalan wajah dalam konteks aplikasi web dan mendukung pengenalan wajah secara *real-time*. ^[19]

Dan adapun fitur-fitur atau *class* yang disediakan oleh *library face-api.js* adalah sebagai berikut: ^[20]

a. *Face Recognition*



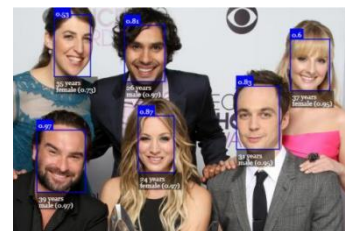
b. *Face Expression Recognition*



c. *Face Landmark Detection*



d. *Age Estimation & Gender*



Sumber: <https://github.com/justadudewhohacks/face-api.js>

2.8. Library Silent-Face-Anti-Spoofing

“The silent-face-anti-spoofing detection model is used to determine if the face in an image is real or fake.

It is designed to prevent people from tricking facial identification systems, such as those used for unlocking phones or accessing secure locations. This is achieved through a process called "liveness" or "anti spoofing" which judges whether the face presented is genuine or not.

The face presented by other media can be defined as a fake: photo prints of faces, faces on phone screens, silicone mask, 3D human image, etc. This model outputs three concepts: fake2d, fake3d, real.”

Model deteksi *library silent-face-anti-spoofing* digunakan untuk menentukan apakah wajah dalam suatu gambar asli atau palsu.

Ini dirancang untuk mencegah orang menipu sistem identifikasi wajah, seperti yang digunakan untuk membuka kunci ponsel atau mengakses lokasi aman. Hal ini dicapai melalui proses yang disebut *"liveness"* atau *"anti spoofing"* yang menilai apakah wajah yang ditampilkan asli atau tidak.

Wajah yang ditampilkan oleh media dapat diartikan palsu ketika dari: cetakan foto wajah, wajah di layar ponsel, masker silikon, gambar manusia 3D, dll. Model ini menghasilkan tiga konsep: *palsu2d, palsu3d, real*.

Sumber: https://github.com/minivision-ai/Silent-Face-Anti-Spoofing/blob/master/README_EN.md

2.8.1. Liveness

Liveness dalam konteks teknologi pengenalan wajah dan *biometrik* merujuk pada kemampuan sistem untuk mengidentifikasi apakah data *biometrik* yang dihadirkan adalah dari sumber yang hidup atau dari sesuatu yang tidak hidup seperti foto atau rekaman video. Istilah ini sering digunakan dalam sistem keamanan dan otentikasi untuk mengatasi masalah potensial dengan penggunaan citra statis (foto) sebagai upaya penipuan.

Sistem deteksi *liveness* berusaha untuk membedakan antara data *biometrik* yang berasal dari sumber yang hidup, seperti wajah seseorang yang sebenarnya, dengan data yang berasal dari sumber palsu atau rekaman, seperti foto wajah. Ini dapat dicapai dengan berbagai cara, termasuk analisis dinamika (seperti gerakan mata atau perubahan warna kulit), penggunaan teknologi 3D untuk mendeteksi kedalaman, atau pengujian tantangan (*challenges*) seperti meminta pengguna untuk melakukan tindakan tertentu (misalnya, menggerakkan kepala).

Sistem deteksi *liveness* adalah salah satu langkah keamanan tambahan yang digunakan dalam aplikasi seperti otentikasi wajah untuk memastikan bahwa sumber *data biometrik* adalah manusia yang sebenarnya dan bukan representasi data statis. Dengan demikian, sistem ini membantu mencegah upaya penipuan dengan menggunakan foto atau rekaman video sebagai cara untuk membuka kunci perangkat atau layanan. ^[1]

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Usecase Diagram Gambaran Umum Sistem

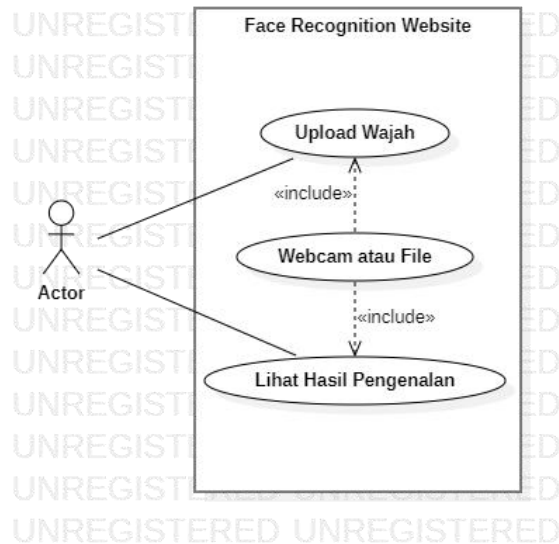


Fig. 20 Gambaran Umum Sistem

3.2. Flowchart Diagram Simpan Gambar Sementara Pada Local Storage

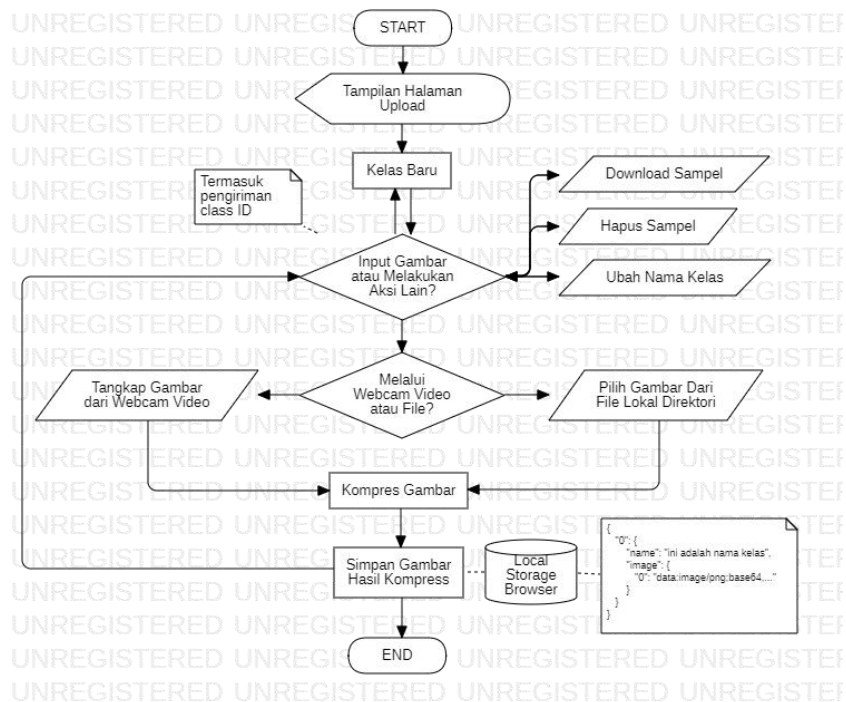


Fig. 21 Flow Gambaran Penyimpanan Semestara Data Gambar Pada Local Storage

3.3. Flowchart Diagram Upload Citra Wajah

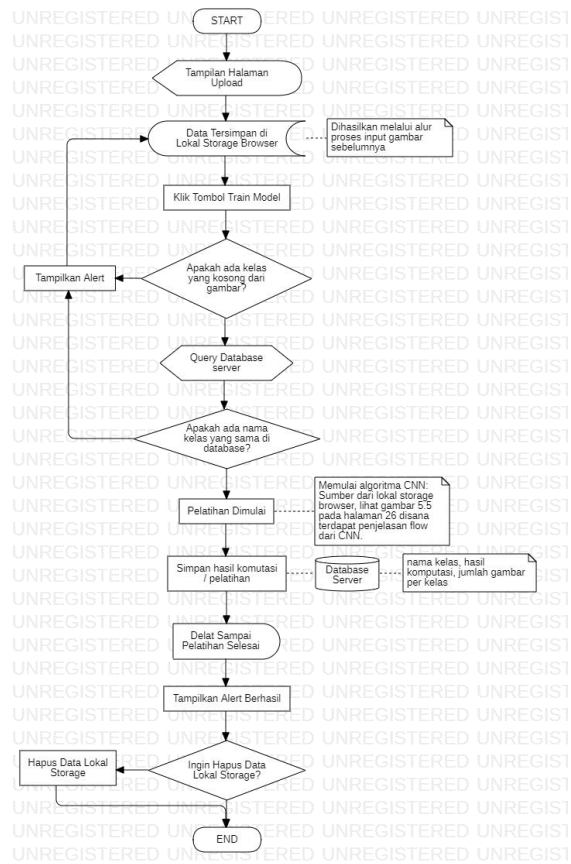


Fig. 22 Flow Upload Gambar / Citra ke Server

3.4. Flowchart Halaman Pengenalan Wajah Pada Citra

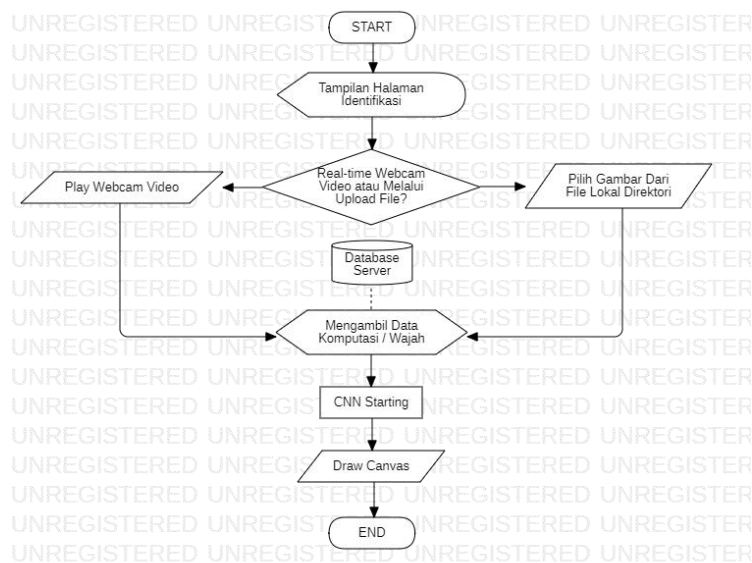


Fig. 23 Flow Sistem Melakukan Prediksi Nama Wajah Yang Diberikan

3.5. Flowchart Diagram CNN Dalam Kostumisasi Dataset

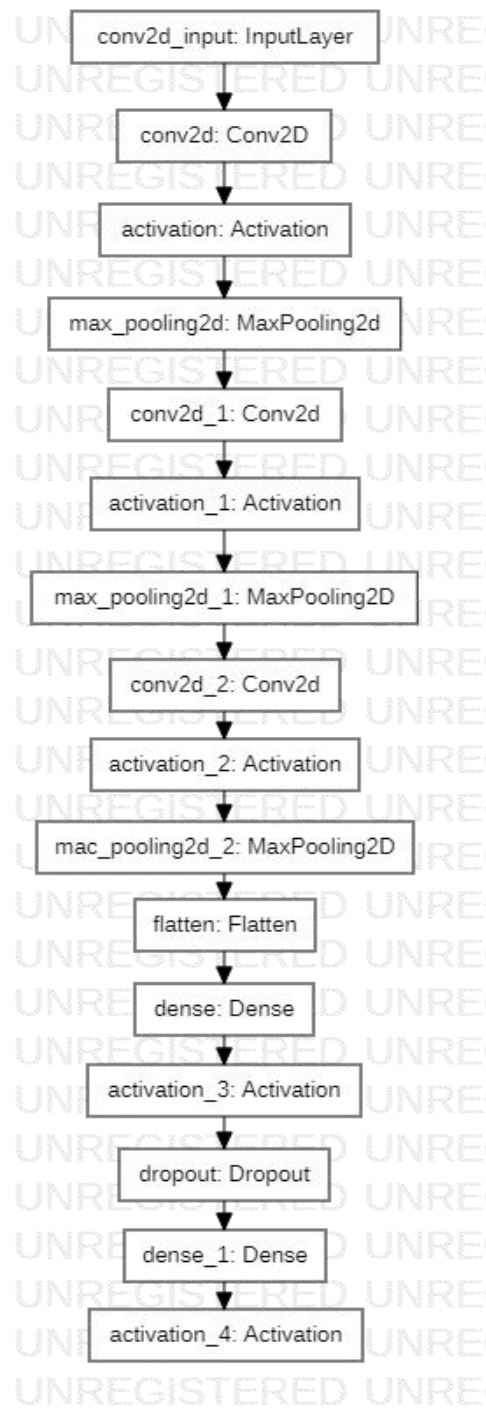


Fig. 24 Gambar Tambahan Untuk Flow CNN Bekerja

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1. Implementasi

Pada bab ini, akan dijelaskan langkah-langkah konkret dalam melaksanakan implementasi teknologi pengenalan wajah menggunakan atau memanfaatkan pustaka *face-api.js*. Sebagai catatan, implementasi di sini tidak melibatkan pembuatan model *CNN* secara mandiri. Mengikuti judul, abstrak penelitian dan batasan masalah yang telah disebutkan sebelumnya. Teknik *Convolutional Neural Network (CNN)* hanya sedekedar pemahaman mendasar tentang konsep dan cara mekanisme kerjanya, hal ini dimaksudkan agar pembaca, khususnya penulis, memiliki gambaran umum tentang hal-hal terkait teknik tersebut. Dengan begitu, ketika ingin membuat model sendiri di masa depan, sudah memiliki dasar-dasarnya dan tidak bingung. *Library face-api.js* telah mengimplementasikan model *CNN* yang telah dilatih sebelumnya untuk pendeteksian wajah, fitur wajah (*landmark*), dan pengenalan wajah. Berikut adalah beberapa penjelasan yang akan menjelaskan tentang implementasi *library face-api.js & silent-face-anti-spoofing* lalu penjelasan antar muka dan design sistem yang telah dibuat dan terakhir penjelasan tentang pengujian sistem.

4.2.1. Implementasi Library face-api.js & silent-face-anti-spoofing

Dalam pengimplementasian *library face-api.js & silent-face-anti-spoofing* ini, penulis lebih berfokus pada penggunaan yang sederhana daripada memberikan penjelasan rinci tentang proses pembuatan aplikasi yang telah dikembangkan. Hal ini dipilih karena jika penjelasan yang mendalam mengenai aplikasi yang telah dibuat ini, hal ini akan menjadi cukup panjang dan rumit. Sebagai gantinya, pada point ini hanya menjelaskan cara penggunaan *library* tersebut dengan studi kasus yang sederhana.

Selain itu, daripada menjeaskan secara tertulis, penulis memutuskan untuk membuat sebuah video yang memberikan penjelasan yang mungkin

akan lebih mudah dipahami dan mungkin juga akan dapat lebih mendetail. Video tersebut dapat ditemukan di tautan berikut: <https://github.com/mochamaddarmawanh/skripsi/tree/main/video>.

Dengan demikian, penjelasan ini akan lebih fokus pada memberikan panduan praktis mengenai penggunaan *library face-api.js*, daripada membahas proses pembuatan sistem ini yang mungkin saja ada beberapa hal yang tidak penting untuk dijelaskan yang tidak ada hubungannya dengan pembangunan sistem pengenalan wajah nantinya.

4.2.2. Antarmuka Pengguna & Design Sistem

Pada point antar muka dan design sistem penulis terinspirasi dari demo *TensorFlow* pada bagian "*TensorFlow for Web*" atau pada kategori "*Tensorflow.js*". Salah satu contoh demo yang menjadi sumber inspirasi adalah pada point "*Teachable Machine*", yang dapat diakses melalui tautan berikut (<https://www.tensorflow.org/js/demos>). Disana dapat ditemukan informasi lebih lanjut pada bagian yang berjudul "*Teachable Machine*".

Di bawah ini adalah beberapa tampilan yang telah dibuat, yang terinspirasi oleh konsep "*Teachable Machine*" dari *TensorFlow for Web*:

a. Halaman Awal

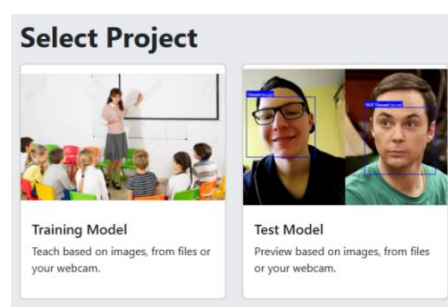


Fig. 25 Gambar Tampilan Awal Sistem

Pada halaman ini pengguna di haruskan memilih 1 dari 2 pilihan, antara men-training (upload wajah) atau pengujian (men-identifikasi atau melakukan pengenalan pada gambar wajah yang di input).

b. Halaman Training

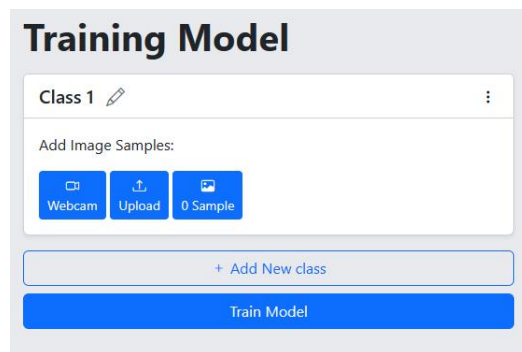


Fig. 26 Halaman Training (Tampilan Awal / Pilihan)

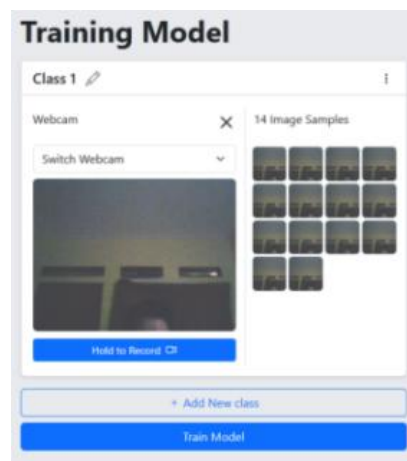


Fig. 27 Tampilan Upload Melalui Webcam

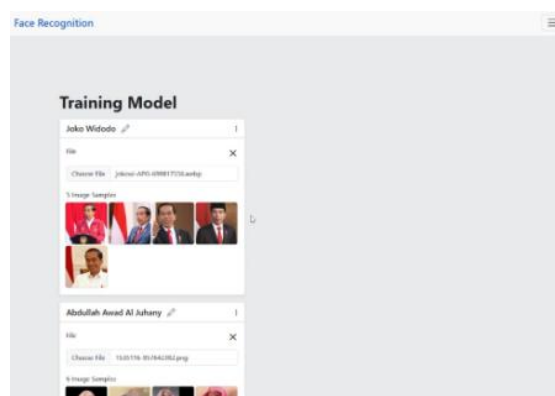


Fig. 28 Tampilan Upload Melalui Pilih File

Pada halaman training, pengguna dapat mengunggah gambar wajah melalui file atau webcam. Selain itu, pengguna dapat memodifikasi nama, mengunduh yang sudah disimpan dalam local

storage, dan menghapus contoh yang telah diambil. Setelah semua langkah selesai, pengguna dapat menjalankan pelatihan sistem, dan gambar-gambar tersebut akan diolah untuk diambil hasil komutasinya. Hasilnya kemudian akan disimpan dalam database pada tabel yang diberi nama "face".

- c. Halaman testing (Mengidentifikasi atau Proses Pengenalan Wajah pada Gambar/Citra yang Diberikan)

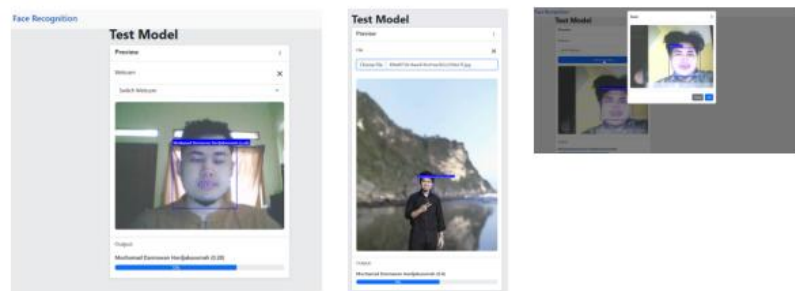
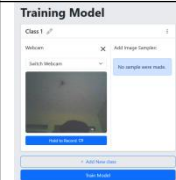
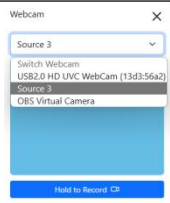
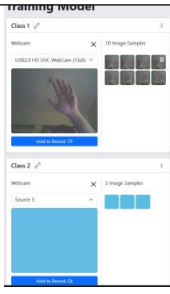
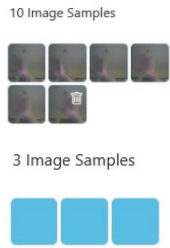
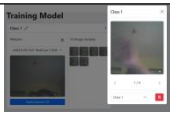



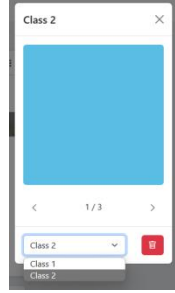


Fig. 29 Pengenalan Melalui Live Webcam atau File, dan Deteksi Spoofing

Pada halaman pengenalan, konsepnya, di mana pengguna dapat memilih metode webcam atau file. Pada halaman pengujian ini, terdapat tombol "Detect Spoofing" yang dapat digunakan untuk mendeteksi serangan spoofing. Meskipun dalam konteks penelitian ini deteksi spoofing belum sepenuhnya real-time, setidaknya langkah-langkah pendeteksian telah diimplementasikan untuk mengatasi tantangan tersebut.

4.2. Pengujian Skenario I Halaman Training

PENGUJIAN	HASIL YANG DIHARAPKAN	IMAGE	KESIMPULAN
Tombol Webcam	Ketika tombol ditekan, video akan diaktifkan dan daftar webcam akan ditampilkan. Jika tidak ada gambar kelas yang tersimpan, maka munculkan alert bootstrap, tetapi jika terdapat gambar yang telah disimpan, maka tampilkan gambar-gambar tersebut tentunya dengan ID		OK

	yang sesuai.		
Switch Webcam	Mengganti ke kamera lain yang tersedia.		OK
Hold to Record	Gambar akan ditangkap dan disimpan ke ID imgSample_ sesuai ID kelas yang sebelumnya di lakukan proses cropping.		OK
Hapus Gambar	Menghapus satu gambar pada saat klik icon trash ketika di hover.		OK
Modal	Diharapkan gambar muncul di modal dengan penomoran sesuai gambar dipilih. Nama kelas juga diharapkan muncul di atas. Pagination untuk pindah gambar juga muncul dengan jumlah sesuai jumlah gambar di kelas dipilih. Dan terakhir terdapat tombol hapus dengan ikon trash merah.		OK
Uji pagination	Diharapkan pagination di modal berfungsi dengan seharusnya untuk mengganti gambar atau jika secara sistem memuat ulang modal dengan ID gambar dan ID kelas yang sesuai.		OK

Select Kelas	Memuat ID gambar dan ID kelas yang sesuai saat dipilih.		OK
Uji Tombol Trash di Modal	Menghapus gambar yang muncul dan secara otomatis me-refresh atau memuat ulang dengan ID di depannya.		OK
Tombol Upload Melalui File	Menampilkan form untuk memilih file. Gambar yang disimpan di penyimpanan lokal juga ditampilkan di bawahnya sesuai gambar kelas nya.		OK
Upload Melalui File	Ketika onchange gambar yang dipilih dimasukkan ke dalam container di bawahnya yang sebelumnya di lakukan proses cropping.		OK
Tombol Lihat Sampel	Diharapkan tombol ini terdapat nomor yang mempresentasikan jumlah dari gambar yang telah disimpan dari kelas. Dan ketika diklik, akan membuka sampel gambar yang telah disimpan di penyimpanan lokal.		OK
Ganti Nama Kelas	Diharapkan ganti nama kelas berjalan dengan seharusnya		OK

	dengan cara melihat di console browser apakah nama kelas yang terganti sudah sesuai ID nya atau masih acak.		
Hapus Semua Sampel	Diharapkan action ini akan menghapus semua gambar sesuai dengan kelas yang dipilih.		OK
Unduh Semua Sampel	Diharapkan pengguna dapat mengunduh semua sampel dalam bentuk ZIP dengan nama ZIP sesuai nama kelas dan di dalam gambar nya dalam format JPG.		OK
Tambah Kelas Baru	Diharapkan sistem dapat menambahkan card kelas baru dengan ID yang baru untuk membedakan antara kelas.		OK
Tombol Train Model	Menyimpan gambar dari kelas yang dibuat ke dalam tabel komputasi, jumlah gambar ke dalam kolom sum, dan nama kelas ke dalam kolom name. Sistem juga membuat folder baru di server dengan nama kelas untuk menyimpan gambar. Sistem akan memeriksa jika ada kelas tanpa gambar atau jika ada nama kelas yang sama di database.		OK

Table 2 Pengujian Halaman Training

4.3. Pengujian Skenario II Halaman Test

PENGUJIAN	HASIL YANG DIHARAPKAN	IMAGE	KESIMPILAN
Webcam	Muncul kotak, landmark dan nama di webcam.		OK
Test Spoofing Dengan Handphone	Ketika di kasih gambar melalui handphone diharapkan sistem akan mendeteksi ini adalah fake.		OK
Test Spoofing Dengan Kertas Print	Diharapkan semua percobaan berhasil di deteksi ini adalah fake.		Tidak Konsisten (penjelasan lebih lanjut dibawah setelah tabel ini)

			
Melalui File	Diharapkan sistem dapat mendeteksi wajah pada gambar yang diberikan atau di upload.		OK

Table 3 Pengujian Halaman Test

KESIMPULAN PENJELASAN DETEKSI SPOOFING

- Kesimpulan Umum & Kelemahan yang Teridentifikasi

Berdasarkan hasil pengujian yang dilakukan, dapat dinyatakan bahwa model Silent-Face-Anti-Spoofing menunjukkan performa yang bervariasi dalam mendeteksi wajah nyata dan palsu.

Pengamatan menunjukkan bahwa model cenderung memberikan hasil yang tidak konsisten, terkadang mengidentifikasi wajah asli sebagai palsu, dan sebaliknya. Hal ini dapat terkait dengan beberapa kelemahan dalam arsitektur model atau dataset pelatihan yang digunakan.

Hal ini mungkin dipengaruhi oleh karakteristik dataset pelatihan, yang mungkin tidak mencakup secara memadai, seperti variasi serangan spoofing atau kondisi pencahayaan tertentu.

- Kemungkinan Pengaruh Karakteristik Kualitas Gambar

Dalam beberapa kasus, model mungkin mengalami kesulitan dalam mengenali karakteristik wajah tertentu, seperti wajah dengan kualitas tinggi dari foto studio.

- Saran untuk Library

Sebagai langkah lanjutan, akan lebih baik jika memperkaya dataset pelatihan dengan kasus-kasus serangan spoofing yang lebih bervariasi dan mempertimbangkan pengoptimalan ulang pada parameter model untuk meningkatkan konsistensi dan keakuratan deteksi.

- Pemeliharaan Model

Penting bahwa deteksi anti-spoofing adalah bidang penelitian yang terus berkembang, dan pemeliharaan model secara berkala dengan memperbarui dataset dan parameter dapat meningkatkan performa model seiring waktu.

- Kesadaran atas Keterbatasan Teknologi Saat Ini

Penulis juga menyadari bahwa setiap teknologi memiliki keterbatasan, dan dalam konteks deteksi spoofing, tantangan ini tetap menjadi fokus penelitian di komunitas ilmiah dan industri.

Secara keseluruhan, untuk digunakan di level production penulis kira masih dapat digunakan dengan catatan pengguna/pemakai tidak mengetahui kelemahan dari sistem spoofing ini, dan penulis kira, mungkin pemakai tidak akan berani untuk melakukan kejahatan. Ya mungkin, jika ada dan ketahuan maka berikan saja peringatan, karna movie mengatakan “no system is safe”. Atau cari library lain atau buat model sendiri.

BAB V

PENUTUP

5.1. Kesimpulan

Dalam penelitian ini, penulis berhasil menguraikan dan memahami paradigma Teknologi Pembelajaran Mesin (*Machine Learning - ML*), yang berfokus pada konsep serta mekanisme kerja dari teknik Jaringan Saraf Konvolusi (*Convolutional Neural Network - CNN*) dalam pengenalan wajah.

Implementasi *library face-api.js* pada halaman web juga berhasil dijalankan, termasuk langkah-langkah untuk menghindari komputasi berulang saat sistem melakukan pengenalan wajah, sehingga hal ini sistem memungkinkan untuk mendeteksi wajah, mengenali titik-titik landmark wajah, serta melakukan pengenalan wajah secara efisien dan cepat.

Selain itu, permasalahan serangan *spoofing* juga telah berhasil diatasi, memberikan lapisan keamanan tambahan pada sistem. Ini menjadi suatu pencapaian karena implementasi dari *library face-api.js* sebelumnya belum tersedia untuk mengatasi serangan *spoofing* ini, walaupun permasalahan serangan *spoofing* belum dapat dilakukan secara real-time dan masih belum sempurna, penelitian ini telah memberikan langkah-langkah awal untuk mengatasi permasalahan ini. Dan oleh karena belum dapat dilakukan secara real-time dan masih belum sempurna (terkadang mengidentifikasi wajah asli sebagai palsu, dan sebaliknya), maka akan lebih baik jika dilakukan penelitian lebih lanjut terkait permasalahan serangan *spoofing* ini, sehingga nantinya tidak memerlukan intervensi secara manual lagi dan sempurna.

Secara keseluruhan, penelitian ini berhasil mencapai tujuan awalnya yaitu pemahaman tentang paradigma *ML* dengan fokus pada teknik *CNN*, serta pengembangan solusi pengenalan wajah yang berfokus pada basis web dengan memanfaatkan *library face-api.js*.

5.2. Saran

Saran dari penulis mengarah pada potensi pengembangan lebih lanjut dalam penelitian ini, yaitu dalam konteks pengimplementasian teknik *Convolutional Neural Network (CNN)* secara khusus. Gagasan ini mendorong untuk membuat model pengenalan wajah secara mandiri, sehingga tidak bergantung pada *library face-api.js* lagi. Meskipun langkah ini tentu saja memerlukan usaha dan tantangan yang tidak mudah, namun mencoba pendekatan ini memiliki nilai yang sangat berharga.

Pengembangan model pengenalan wajah berbasis *CNN* sendiri memberikan potensi untuk meningkatkan pemahaman mendalam terhadap mekanisme di balik teknologi pengenalan wajah. Dengan merancang dan melatih model sendiri, penelitian tersebut dapat memberikan wawasan lebih mendalam tentang bagaimana *CNN* bekerja untuk mengenali fitur-fitur wajah secara praktik. Langkah ini juga akan memungkinkan eksplorasi berbagai metode pra-pemrosesan gambar, optimisasi model, dan pemilihan parameter yang spesifik untuk tugas pengenalan wajah.

Oleh karena itu, saran ini mencerminkan semangat eksplorasi dan inovasi dalam penelitian. Dalam mengambil langkah lebih jauh dengan mengembangkan model pengenalan wajah berbasis *CNN*, peneliti dapat menciptakan kontribusi berharga terhadap pengetahuan dan pengembangan teknologi pengenalan wajah di masa depan.

LAMPIRAN

<https://github.com/mochamaddarmawanh/skripsi>

DAFTAR PUSTAKA

- [1] OpenAI, *Bantuan dalam Penulisan Riset, Kode, dan Pertanyaan Lainnya*, GPT-3.5 ed., OpenAI, 2021.
- [2] A. Purnama, *190653001 - [e] Grafika Komputer*, Bandung: Universitas Widyatama, Ganjil 2022/2023.
- [3] S. Violina, *190663003 - [e] Pengolahan Citra*, Bandung: Universitas Widyatama, Ganjil 2022/2023.
- [4] M. Fachrie, "Konsep Dasar Citra Digital - Perkuliahan Pengolahan Citra Digital #1." YouTube, 2021. [Online]. Available: https://www.youtube.com/watch?v=vMXTEXYQ4RM&list=PLBW2heg-PA3e_1ObQponUnL8I-eZWRbCy. [Accessed June 2023].
- [5] G. f. Deeloper, "Artificial Intelligence, Machine Learning, and Deep Learning," YouTube, 2023. [Online]. Available: <https://www.youtube.com/watch?v=bOUfOOCFCrE>. [Accessed August 2023].
- [6] R. Ilyas, "Perbedaan Machine Learning dengan Program Tradisional | Machine Learning 101 | Eps 1," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=crIQS9x3QnE&list=PLo6nZTcPsz2p5oKKkg6ZWx4Pw7ToYVtD&index=1>. [Accessed August 2023].
- [7] M. Astrid, "Bentuk Otaknya AI | Pengenalan Artificial Neural Network," YouTube, 2020. [Online]. Available: https://www.youtube.com/watch?v=VmQNVsU_mPU&t=5s. [Accessed June 2023].
- [8] Intellipat, "Artificial Intelligence Tutorial | AI Tutorial For Beginners | Intellipaat," YouTube, 2019. [Online]. Available: https://www.youtube.com/watch?v=SJ_6TD6X8UE. [Accessed August 2023].
- [9] "What is a Neural Network?. IBM," IBM, [Online]. Available: <https://www.ibm.com/topics/neural-networks#:~:text=Neural%20networks%2C%20also%20known%20as,neurons%20signal%20to%20one%20another>. [Accessed June 2023].

- [10] M. Fachrie, "Neural Networks untuk Pemula - Perkuliahan Soft Computing #06," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=O-tfsQPI3RE&t=2803s>. [Accessed June 2023].
- [11] R. Ilyas, "Perhitungan dan Simulasi Backpropagation Dengan MS Excel | Machine Learning 101 | Eps 6," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=iFcgzZOqYeU&list=PLo6nZTcPsz2p5oKKkg6ZWHx4Pw7ToYVtD&index=6>. [Accessed June 2023].
- [12] M. Astrid, "Mengenal Convolutional Neural Network," YouTube, 2020. [Online]. Available: <https://www.youtube.com/watch?v=3NwE3Eu8g7c&t=2s>. [Accessed June 2023].
- [13] B. Suman, "Convolutional Neural Networks | CNN | Kernel | Stride | Padding | Pooling | Flatten | Formula," YouTube, 2020. [Online]. Available: <https://www.youtube.com/watch?v=Y1qxI-Df4Lk&t=302s>. [Accessed June 2023].
- [14] J. Patel, "Convolutional Neural Network [Playlist]," YouTube, 2022. [Online]. Available: <https://www.youtube.com/playlist?list=PLuhqtP7jdD8CD6rOWy20INGM44kULvrHu>. [Accessed August 2023].
- [15] X. Yao, "CNN Convolutional Layer Explained." YouTube, 2018. [Online]. Available: <https://www.youtube.com/watch?v=7PZDbTfvDIQ>. [Accessed August 2023].
- [16] Wira, "S6E1 | Intuisi dan Cara Kerja Convolutional Neural Network (CNN) | Deep Learning Basic," YouTube, 2020. [Online]. Available: <https://www.youtube.com/watch?v=6Hb81DxD7yw>. [Accessed August 2023].
- [17] M. Astrid, "Dropout neuron untuk mengurangi overfitting," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=ciQTDDNoMcg&t=54s>. [Accessed June 2023].
- [18] J. Peter, "Belajar TensorFlow.js Bahasa Indonesia [Playlist]," YouTube, 2021. [Online]. Available: <https://www.youtube.com/playlist?list=PLBKh3ZtuAtGFdmchLIvFxBfGngqCWPI>

- QYP. [Accessed June 2023].
- [19] D. Gupta, "Face Detection Using JavaScript API — face-api.js. Towards Data Science," Medium, 2019. [Online]. Available: <https://towardsdatascience.com/face-recognition-using-javascript-api-face-api-js-75af10bc3dee>. [Accessed August 2023].
- [20] V. Mühler, J. Derrough, Javier, ... and K. Alexis, "JavaScript API for face detection and face recognition in the browser and Node.js with TensorFlow.js," GitHub, 2020. [Online]. Available: <https://github.com/justadudewhohacks/face-api.js>. [Accessed 2021].
- [21] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs and H. Lipson, "Understanding Neural Networks Through Deep Visualization. Cornell University," Cornell University, 2015. [Online]. Available: <https://arxiv.org/abs/1506.06579>. [Accessed August 2023].
- [22] Felipe, "Face recognition + liveness detection: Face attendance system," YouTube, [Online]. Available: https://www.youtube.com/watch?v=_KvtVk8Gk1A&t=1376s. [Accessed June 2023].
- [24] S. Violina, *190651005 - Artificial Intelligence*, Bandung: Universitas Widyatama, Ganjil 2022/2023.
- [25] Sunjana, Interviewer, *Chain Rule atau Aturan Rantai dalam Kalkulus*. [Interview]. November 2023.
- [26] Y. Syukriyah, *190621003 - Kalkulus II*, Bandung: Universitas Widyatama, Ganjil 2022/2023.
- [27] V. Powell, "Image Kernels Explained Visually," Setosa, [Online]. Available: <https://setosa.io/ev/image-kernels/>. [Accessed August 2023].
- [28] C. Edukaze, "Konsep Artificial Neural Networks (Jaringan Syaraf Tiruan)," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=TKFKt1dn788&t=112s>. [Accessed June 2023].

- [29] S. Raschka, "L13.6 CNNs & Backpropagation," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=-SwKNK9MIUU>. [Accessed August 2023].
- [30] G. Singh, "Introduction to Artificial Neural Networks," Analytics Vidhya, 2023. [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/09/introduction-to-artificial-neural-networks/>. [Accessed August 2023].
- [31] W. D. Simplified, "Easy Face Recognition Tutorial With JavaScript," Youube, 2019. [Online]. Available: <https://www.youtube.com/watch?v=AZ4PdALMqx0&t=822s>. [Accessed June 2023].
- [32] K. Naik, "Tutorial 6-Chain Rule of Differentiation with BackPropagation," YouTube, 2019. [Online]. Available: <https://www.youtube.com/watch?v=CRB266Eyjkg&list=PLZoTAE LR MXVPGU70ZGscrMdr0FteeRUi&index=10&t=5s>. [Accessed September 2023].
- [33] M. Astrid, "Pengenalan RNN (Recurrent Neural Network)," YouTube, 2021. [Online]. Available: <https://www.youtube.com/watch?v=2GgGu6kMSqE>. [Accessed June 2023].
- [34] M. Astrid, "Analogi loss function," YouTube, 2020. [Online]. Available: <https://www.youtube.com/watch?v=g9F4uK5b3ws>. [Accessed June 2023].
- [35] M. Asrid, "Menuruni grafik loss dengan Gradient Descent | Backpropagation (bagian 1)," YouTube, 2020. [Online]. Available: <https://www.youtube.com/watch?v=0y6mUUY--Es>. [Accessed June 2023].