BAB 2
DASAR TEORI

2.1 KAJIAN PUSTAKA

Beberapa penelitian dengan pemanfaatan AI telah dilakukan sebelumnya berfokus dengan bahasa isyarat dan klasifikasi citra CNN. Penelitian pertama terdapat penelitian yang dilakukan oleh Adigtama Gafar A et al., [8] dengan topik "Sistem Pengenalan Bahasa Indonesia dengan Menggunakan Metode *Fuzzy K-Nearest Neighbor*". Penelitian ini adalah penelitian untuk menghasilkan sistem untuk pengenalan bahasa isyarat SIBI (Sistem Isyarat Bahasa Indonesia) untuk isyarat huruf A-Z dengan memanfaatkan FKNN. Hasil dari penelitian menunjukkan diperolehnya 88,8% untuk akurasi, namun masih ada beberapa huruf yang konsisten belum bisa dikenali, yaitu huruf C, E, L, U, dan V.

Penelitian kedua merupakan penelitian oleh Syulistyo A et al., [6] pada tahun 2020 yang berjudul "SIBI (Sistem Isyarat Bahasa *Indonesia translation using Convolutional Neural Network* (CNN))". Penelitian berupa pendeteksian bahasa isyarat Indonesia dengan memanfaatkan CNN. Pada penelitian ini terdapat 3 kelas dengan dataset sejumlah 110. Berdasarkan eksperimen penelitian, CNN mampu menerjemahkan gambar *input* menjadi label kelas yang diharapkan dengan akurasi 100% .

Penelitian ketiga yaitu yang dilakukan oleh Dicky Saputra A et al., [9] pada tahun 2020 dengan judul penelitiannya "Klasifikasi Alfabet Bahasa Isyarat Indonesia (BISINDO) dengan Metode *Template Matching* dan K-*Nearest Neighbors* (KNN)". Tujuan penelitian ini adalah untuk mendeteksi abjad isyarat statis dan dinamis lalu mengklasifikasikannya sehingga *output*nya adalah sebuah teks yang dapat dipahami. Penelitian menyatakan bahwa penggunaan metode *template matching* sangat efektif dalam mengklasifikasi dan mendeteksi gerakan abjad isyarat sehingga dapat dikonversi menjadi sebuah teks dengan baik dengan hasil yang didapatkan dari uji kecocokan *template matching* yaitu 11 gambar yang cocok dengan gambar *template* dari total 17 gambar yang diujikan. Dengan

persentase kecocokan gambar sebesar 85,04% untuk abjad isyarat statis dan 84,65% untuk abjad isyarat dinamis. Sedangkan hasil yang didapatkan dari tahap klasifikasi KNN sebesar 96,52%.

Penelitian keempat pada tahun 2022 yaitu oleh Siswanto F et al., [7] dengan judul penelitian "Klasifikasi Bahasa Isyarat Amerika Menggunakan *Convolutional Neural Network*". Fokus dari penelitian ini adalah untuk memberikan gambaran mengenai pengimplementasian *deep learning Convolutional Neural Network* (CNN) pada *American Sign Language* (ASL) *Classifier*. Dataset yang digunakan pada penelitian ini sejumlah 26 kelas, yaitu alfabet A-Z. Hasil penelitian yang didapatkan adalah akurasi di angka 82,1% dengan huruf yang diprediksi paling bagus adalah v dan n, sedangkan model paling buruk saat memprediksi huruf i, c, d, dan x.

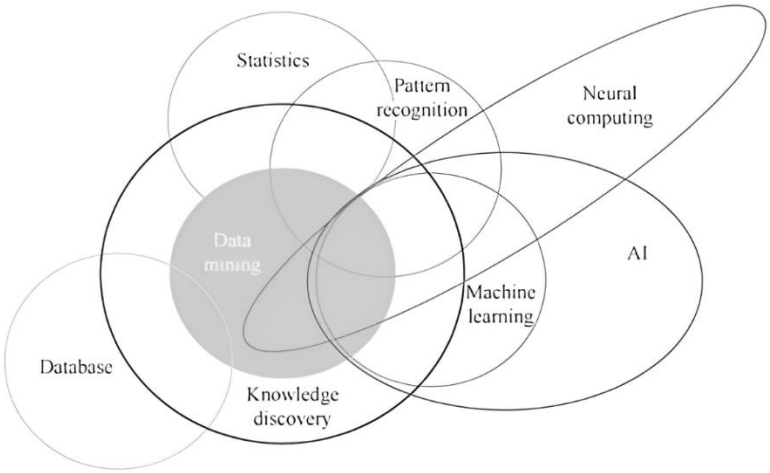
Penelitian kelima yaitu pada tahun 2023 oleh Tamam M et al., [10] dengan topik penelitiannya "*Classification of Sign Language in Real Time Using Convolutional Neural Network*". Penelitian ini bertujuan untuk mengklasifikasikan gerakan tangan dari bahasa isyarat menjadi huruf-huruf menggunakan CNN. Dataset yang digunakan diperoleh dari Kaggle, dengan total 34.627 data yang dibagi dengan rasio data pelatihan dan pengujian sebesar 80:20. Dari hasil pengujian, huruf-huruf alfabet yang dapat diterjemahkan adalah: A- Z. Dengan mengimplementasikan tahap *pra*-proses yang melibatkan pengubahan gambar menjadi bingkai (*frames*) dan memberikan gerakan alfabetik pada bagian tangan yang ditampilkan. Kemudian, pemotongan dilakukan pada area gerakan tersebut. Hasil akhir dari *pra*-proses ini akan disimpan di penyimpanan internal sistem. Hasilnya menunjukkan bahwa CNN mampu mengenali alfabet dengan sukses. Huruf-huruf yang paling mudah ditebak adalah V dan N, sedangkan huruf-huruf yang paling sulit ditebak adalah n, c, j, dan z. Dengan *pra*-proses yang diterapkan nilai *loss* dapat dikurangi, sehingga hasil ahirnya memberikan akurasi yaitu 95,4%.

2.2 DASAR TEORI

2.2.1 *Artificial Intelligence* (AI)

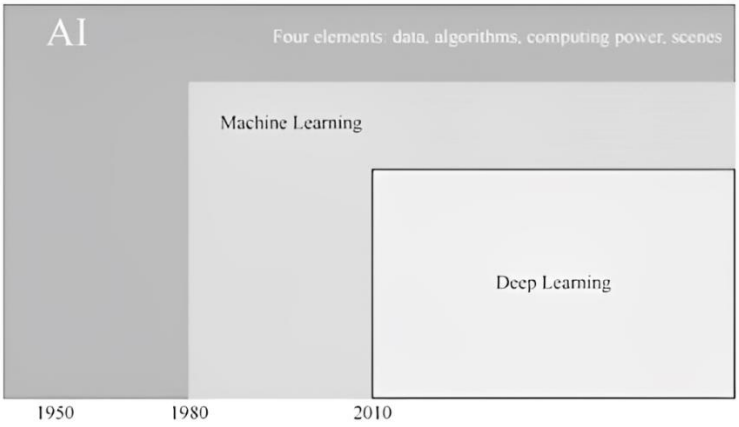
Kecerdasan buatan (AI) adalah sistem komputer yang mampu melakukan tugas-tugas yang biasanya membutuhkan kecerdasan manusia. Teknologi ini dapat

mengambil keputusan dengan menganalisis dan menggunakan data yang tersedia di sistem. Proses yang terjadi pada AI meliputi pembelajaran, penalaran, dan *self-correction*. Proses ini mirip dengan manusia yang melakukan analisis sebelum mengambil keputusan [11].



Gambar 2.1 Bidang Cakupan *Artificial Intelligence* (AI) [12]

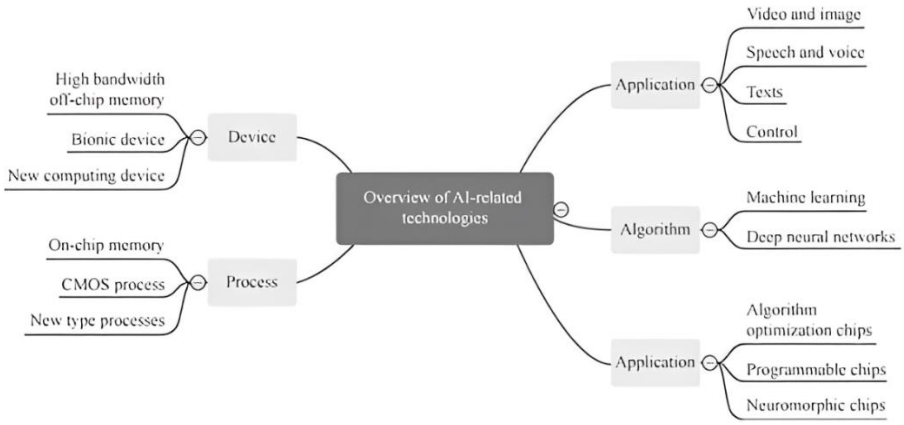
Kecerdasan buatan (AI) adalah jenis ilmu teknologi yang relatif baru yang berfokus pada penelitian dan pengembangan teori, metode, teknologi, serta sistem aplikasi untuk mensimulasikan, meningkatkan, dan memperbarui kecerdasan manusia. AI dirancang untuk memungkinkan mesin berpikir seperti manusia dan diberi kemampuan kecerdasan. Saat ini, cakupan AI telah berkembang secara signifikan, menjadikannya subjek yang bersifat interdisipliner, sebagaimana ditunjukkan oleh gambar 2.1. Salah satu fokus utama dari subjek interdisipliner adalah *machine learning*. Penelitian dalam *Machine Learning* (ML) bertujuan untuk memungkinkan komputer meniru kemampuan belajar manusia serta mengembangkan pengetahuan dan keterampilan baru. *Deep Learning* (DL) berasal dari studi tentang jaringan saraf tiruan (*Artificial Neural Networks*/ANN) yang mana merupakan cabang baru dalam *machine learning* dan berfokus pada meniru cara otak manusia menginterpretasikan data seperti gambar, suara, dan teks [12]. Hubungan antara kecerdasan buatan (AI), *machine learning*, dan *deep learning* dapat dilihat dalam ilustrasi pada gambar berikut:



Gambar 2.2 Hubungan Antara AI, *Machine Learning*, dan *Deep Learning* [12]

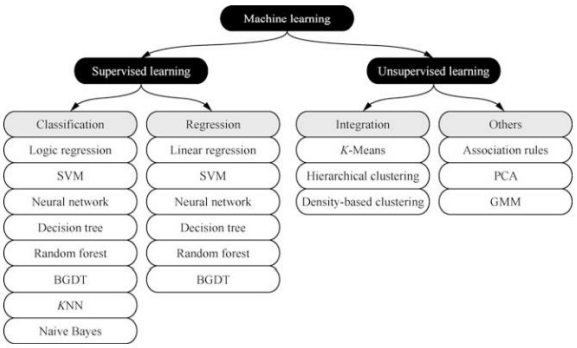
Dari tiga konsep ini, *machine learning* adalah cara atau bagian dari kecerdasan buatan, sementara *deep learning* merupakan bentuk khusus dari pembelajaran mesin. Dalam analogi dengan AI sebagai otak, *machine learning* mewakili proses di mana kemampuan kognitif diperoleh, sedangkan *deep learning* adalah sistem pelatihan mandiri yang sangat efisien yang mendominasi proses tersebut. Kecerdasan buatan adalah tujuan dan hasil akhir, sementara *deep learning* dan *machine learning* adalah metode dan alat-alat yang digunakan dalam mencapai tujuan tersebut [12].

Teknologi AI bersifat *multi-layer*, mencakup berbagai tingkat teknis seperti aplikasi, algoritma, *chip*, perangkat, dan proses, seperti yang ditunjukkan pada gambar berikut:



Gambar 2.3 Ilustrasi Teknologi Terkait dengan AI Berdasarkan Perangkat, Proses, Algoritma, dan Aplikasi [12]

Algoritma dan rumus dalam kecerdasan buatan (AI) adalah dasar dari berbagai teknologi dan aplikasi AI. Berikut adalah beberapa algoritma dan rumus yang umum digunakan dalam pengaplikasian *machine learning* pada AI:



Gambar 2.4 Algoritma Umum Dalam *Machine Learning* (AI) [12]

Beberapa penjelasan mengenai algoritma tersebut beserta persamaannya adalah sebagai berikut:

1. *Linear Regression*: Menggunakan analisis regresi dalam statistik matematika untuk menentukan hubungan kuantitatif antara dua atau lebih variabel, dinyatakan dalam persamaan:

$$h(x) = w^{T}x + b \tag{2.1}$$

   - w = parameter bobot, b = bias, dan x = sampel.

   Dengan hubungan nilai prediksi model dan nilai sebenarnya sebagai berikut:

$$y = h(x) + \varepsilon \tag{2.2}$$

   - y mewakili nilai sebenarnya dan ε mewakili kesalahan.

2. *Logistic Regression*: Digunakan untuk klasifikasi biner, memprediksi probabilitas hasil yang dikategorikan, formula yang digunakan;

$$h(x)\ \ P(Y = 1|X) = g(w^{T}x + b) \tag{2.3}$$

   - g = fungsi sigmoid, w = bobot, b = bias, dan rumus $w^{T}x + b$ adalah fungsi linear dari x.

   Dengan definisi fungsi sigmoid sebagai berikut:

$$g(x) = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{exp}^{(-x)}} \tag{2.4}$$

3. *Support Vector Machine* (SVM): Digunakan untuk klasifikasi atau regresi, yang mencari *hyperplane* optimal yang memisahkan dua kelas data. Untuk kasus klasifikasi dengan SVM linear, fungsinya:

$$f(x) = sign\ (w \cdot x + b) \tag{2.5}$$

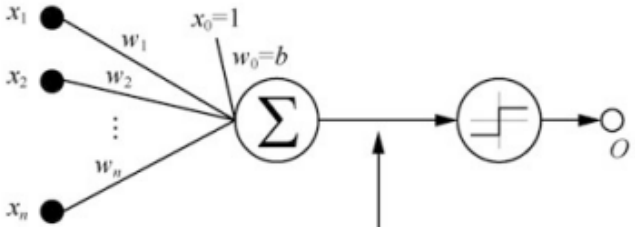
- w = vektor bobot, b = bias, sign(·) adalah fungsi tanda, yang mengembalikan tanda dari ekspresi di dalamnya (+1 atau -1)

4. *K-Nearest Neighbors* (KNN): Metode ini tanpa parameter, cenderung lebih efektif pada dataset dengan batas keputusan yang kompleks atau tidak teratur, dengan ide bahwa jika sebagian besar dari K tetangga terdekat dalam ruang fitur termasuk dalam suatu kategori, maka sampel juga dapat diklasifikasikan ke dalam kategori tersebut.

$$\hat{\boldsymbol{y}} = \arg\max_j \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \boldsymbol{I}(\boldsymbol{yi} = \boldsymbol{j}) \quad (2.6)$$

   - $\hat{\boldsymbol{y}}$ = prediksi data baru, k = jumlah tetangga terdekat yang dipertimbangkan, $\boldsymbol{yi}$ = kelas dari tetangga ke-i, dan I(·) adalah fungsi indikator yang bernilai 1 jika pernyataan di dalamnya benar, dan 0 jika salah.

5. *Neural Network*: Merupakan sistem komputasi yang terdiri dari elemen pemrosesan sederhana yang sangat terhubung, yang disusun secara khusus dan memproses informasi melalui *respons* dinamis terhadap masukan dari luar. Untuk contoh simpelnya digunakan *single-layer perceptron* yang ditunjukkan sebagai berikut:



Gambar 2.5 Ilustrasi *Single-Layer Perceptron* [12]

Penjelasan dari gambar 2.5 yaitu, vektor masukan X = [x0,x1,…,xn]^T pertama-tama dihitung dengan fungsi *dot* dengan bobot W = [w0,w1,…,wn]^T yang disebut net. Diantaranya, x0 sama dengan 1 dan w0 disebut bias. Dan untuk masalah klasifikasi, *net* perlu diaktivasi dengan fungsi aktivasi yang disebut Sgn(*net*) agar dapat diambil sebagai *output*. Fungsi Sgn(x) bernilai 1 ketika x > 0, dan -1 sebaliknya. Untuk rumusnya digunakan persamaan berikut:

$$net = \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{wixi} = W^T X \quad (2.7)$$

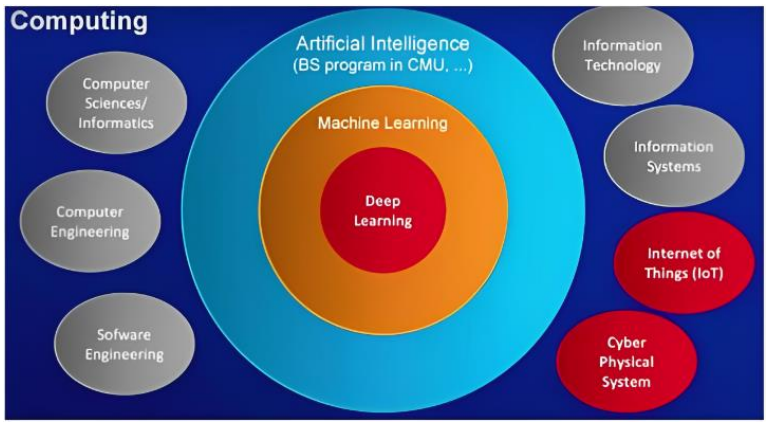
Neural network merupakan dasar dari *deep learning*, atau dinamakan *multi-layer neural networks* [12].

Kelebihan dari *Artificial Intelligence* adalah:

1. Sifatnya yang dimiliki lebih permanen, artinya tidak akan ada perubahan selama sistem komputer dan program berubah. Berbeda dengan sifat yang dimiliki manusia, yaitu pelupa.
2. Lebih mudah disalin dan disebar secara luas. Tidak seperti mengirimkan pengetahuan dari satu orang ke orang yang lain, yang mana membutuhkan proses yang tidak cepat dan bahkan pengetahuan atau keahlian tidak akan pernah dapat ditransfer dengan lengkap.
3. Biaya terjangkau. Dibandingkan dengan memanggil seorang yang ahli untuk mengerjakan pekerjaan yang kemungkinan memakan lebih banyak waktu, dengan menggunakan komputer akan lebih praktis dan murah.
4. Sifatnya konsisten karena AI merupakan bagian dari teknologi komputer sedangkan kecerdasan alami tidak selamanya sama atau berubah-ubah.
5. Dalam pengerjaannya lebih bisa untuk didokumentasi. Dalam pengerjaan hal yang dibuat oleh keputusan komputer akan dapat didokumentasi dengan mudah dengan cara melacak setiap aktivitas yang sebelumnya dilakukan dari sistem tersebut.
6. Efisiensi waktu, karena pengerjaan dilakukan secara cepat.
7. Hasil lebih unggul dari pada pengerjaan oleh manusia [11].

Dalam pengerjaan *Artificial Intelligence* sesuai dengan prosedur algoritma program pada komputer sistem yang disediakan dalam proses pembuatannya. Kerangka berpikir yang digunakan dalam AI yaitu algoritma pemrograman yang dapat digunakan dalam mengolah banyak jenis data. Dengan banyaknya data algoritma yang kompleks, sistem tampaknya mampu berpikir sendiri, mengambil keputusan, belajar, dan beradaptasi. Jadi dalam konteks klasifikasi bahasa isyarat menggunakan *Convolutional Neural Networks* (CNN) dengan python untuk pembelajaran, AI dapat berperan penting dalam mempercepat dan menyederhanakan proses pembelajaran bahasa isyarat. Sistem AI dapat secara otomatis mengidentifikasi dan mengklasifikasikan isyarat-isyarat ini. Memberikan manfaat besar dalam pembelajaran bahasa isyarat bagi masyarakat pendengaran, mereka dapat mempelajari bahasa isyarat dengan lebih efisien dan efektif, serta membuka jalan komunikasi yang lebih inklusif.

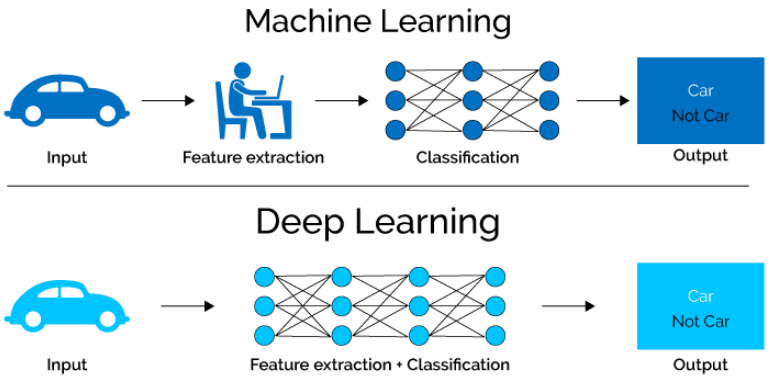
2.2.2 *Deep Learning*



Gambar 2.6 *Deep Learning* [13]

Gambar 2.6 memperlihatkan bahwa *deep learning* adalah bagian dari *machine learning* dan *Artificial Intelligence*. *Deep learning* mampu meniru proses kerja otak manusia, sehingga efektif dalam mengklasifikasi gambar, suara, teks, dan video [14]. Teknologi ini digunakan untuk membangun model abstraksi dengan tingkat kompleksitas yang tinggi. *Deep Learning* dipilih karena memiliki beberapa keunggulan, antara lain, fitur yang dipelajari lebih mudah beradaptasi dan cepat dipelajari. Sering kali, fitur yang dirancang secara manual terlalu spesifik, tidak lengkap, dan sulit untuk dibuat. Selain itu, *deep learning* menawarkan kerangka kerja yang sangat fleksibel, serbaguna, dan dapat dipelajari [13].

Walaupun *deep learning* adalah subkategori dari *machine learning*, keduanya memiliki metode operasional yang cukup berbeda. Perbedaan mendasar dalam pendekatan ini dijelaskan lebih rinci dalam gambar berikut:



Gambar 2.7 Perbedaan *Machine Learning* dengan *Deep Learning* [13]

Dari gambar sebelumnya, perbedaan *machine learning* dengan *deep learning* cukup sekali terlihat. *Machine learning* memerlukan proses ekstraksi fitur untuk mengidentifikasi karakteristik unik dari data yang ada. Ekstraksi fitur ini melibatkan reduksi dimensi, yaitu merangkum kumpulan data yang besar menjadi kelompok-kelompok yang lebih kecil sehingga lebih mudah untuk dikelola dan diproses. Sederhananya, ekstraksi fitur menggabungkan variabel-variabel dalam data menjadi fitur-fitur yang lebih ringkas, sehingga mengurangi volume data yang perlu diproses. Dalam konteks *deep learning* [14], kendala yang sering muncul adalah bertambahnya ukuran jaringan saraf sehingga menambah beban komputasi yang diperlukan. Namun penggunaan GPU dapat mempercepat proses pelatihan data sehingga mengatasi masalah tersebut. Pada *deep learning*, ekstraksi fitur dan klasifikasi dilakukan secara bersamaan dalam satu tahap, berbeda dengan *machine learning* tradisional yang melakukan kedua proses tersebut secara terpisah.

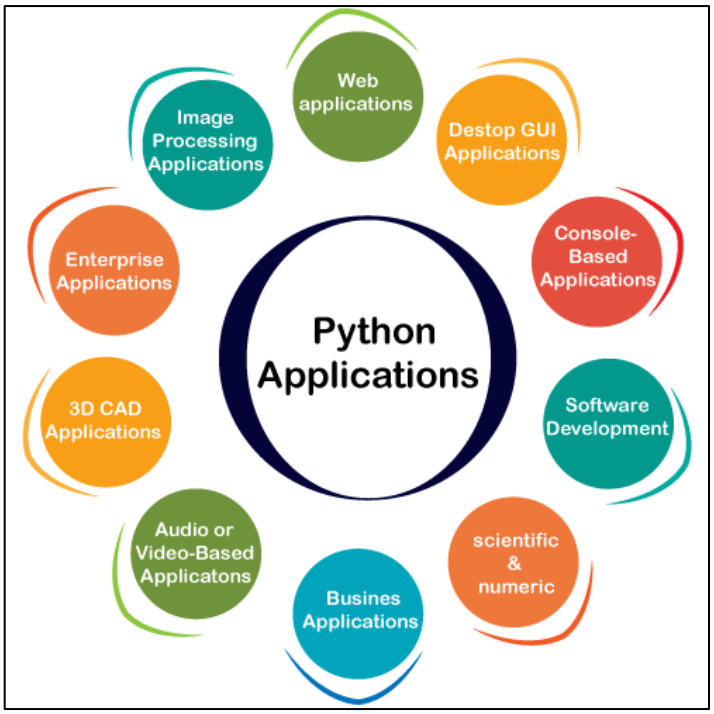
Klasifikasi merupakan suatu proses yang bertujuan untuk mencari atau menentukan kategori dari suatu masukan yang diberikan. Selama proses klasifikasi, data dianalisis berdasarkan perilaku dan atribut yang telah dikelompokkan dan didefinisikan sebelumnya. Misalnya saja klasifikasi dapat dilihat pada ilustrasi ilustrasi gambar 2.7 berikut yang menunjukkan bagaimana data dikelompokkan ke dalam kategori tertentu berdasarkan kategorinya. Dalam klasifikasi, jika hanya ada dua kategori yang diidentifikasi, proses ini disebut sebagai klasifikasi biner (*binary classification*). Sebaliknya, jika terdapat lebih dari dua kategori, maka disebut sebagai klasifikasi multi kelas (*multiclass classification*) [13]. Pemilihan antara *machine learning* dan *deep learning* tergantung pada kebutuhan spesifik. *Deep learning* lebih cocok digunakan ketika data yang akan diproses sangat besar dan kompleks. Ini juga membutuhkan infrastruktur komputasi tingkat tinggi untuk memungkinkan pelatihan dalam waktu yang efisien. Selain itu, *deep learning* digunakan ketika diperlukan analisis mendalam terhadap fitur yang rumit, dan sangat efektif dalam kasus-kasus kompleks seperti klasifikasi gambar, pemrosesan bahasa alami (NLP), dan pengenalan suara.

2.2.3 Python

Python merupakan bahasa pemrograman tingkat tinggi yang dinamis, yaitu bahasa pemrograman interpreter yaitu bahasa yang mengubah kode sumber menjadi kode mesin secara langsung pada saat program dijalankan.

Gambar 2.8 Logo Python [15]

Bahasa ini juga mendukung pengembangan aplikasi dengan pendekatan pemrograman berorientasi objek dan dilengkapi dengan berbagai struktur data tingkat tinggi, menjadikannya mudah dipelajari. Python, sebagai bahasa skrip, terkenal karena kemudahannya dalam belajar serta kekuatan dan fleksibilitasnya, menjadikannya pilihan menarik untuk pengembangan aplikasi. Python mendukung berbagai gaya pemrograman seperti berorientasi objek, imperatif, fungsional, dan prosedural, serta menawarkan alokasi memori dinamis.



Gambar 2.9 Penerapan Aplikasi Python [15]

Selain itu, Python dikenal sebagai bahasa serbaguna yang dapat digunakan untuk berbagai aplikasi seperti pengembangan *web*, aplikasi *enterprise*, 3D CAD, dan lainnya. Sintaks dan pengetikan dinamis pada python, bersama dengan sifatnya sebagai bahasa interpretatif, menjadikannya sangat cocok untuk pembuatan skrip dan pengembangan aplikasi. Python tidak memerlukan deklarasi tipe data untuk variabel karena pengetikannya dilakukan secara dinamis [15].

Python menyediakan banyak fitur berguna. Beberapa fitur penting tersebut diantaranya:

1. Mudah dipelajari dan digunakan, dibandingkan bahasa pemrograman lain python lebih mudah. Sintaksnya mudah dan hampir sama dengan bahasa Inggris.
2. Bahasa ekspresif, python dapat melakukan tugas-tugas kompleks menggunakan beberapa baris kode. Contohnya: program *hello world* cukup mengetikkan *print*(“*Hello World*”).
3. Python dapat dijalankan di berbagai sistem operasi seperti Windows, MacOS, Linux, dan sebagainya.
4. Python memiliki ekosistem luas dengan banyak *library* dan *framework* untuk berbagai kebutuhan seperti data *science*, *web development*, *machine learning*, dan automasi. Contoh *library*: NumPy, Pandas, Matplotlib, dan lainnya.

Bahasa Interpreter, yang berarti program python dijalankan satu baris dalam satu waktu. Keuntungannya adalah membuat proses debug menjadi mudah dan portabel.

2.2.4 *Library* Python

Library dalam konteks pemrograman merujuk pada kumpulan modul yang berisi kode siap pakai yang dapat dimanfaatkan dalam program. Python memiliki banyak *library* yang membantu menyederhanakan proses pembuatan program. *Library-library* ini sangat penting dalam berbagai bidang seperti pembelajaran mesin, data *science*, dan visualisasi data, serta banyak lagi. Python menyediakan lebih dari 137.000 *library*, yang memungkinkan pengembang untuk lebih mudah menulis kode dari awal dan memenuhi berbagai kebutuhan pemrograman [13].

Beberapa *library* python yang sering digunakan dan nantinya akan dimanfaatkan saat pembuatan sistem klasifikasi adalah sebagai berikut.

Gambar 2.10 *Library* Python [13]

1. *Numerical* Python (Numpy) adalah salah satu *library* python yang berguna untuk melakukan komputasi numerik. *Library* ini memungkinkan pembuatan *array* N-dimensi, yang merupakan kumpulan variabel dengan tipe data yang sama. Penggunaan numpy memiliki keunggulan dalam menyederhanakan operasi komputasi pada data. Numpy menyediakan beragam rutinitas untuk operasi cepat pada *array*, termasuk operasi matematika, logika, pengurutan, pemilihan, dan banyak lagi.
2. TensorFlow adalah sebuah *library open source* yang digunakan dalam proyek-proyek komputasi numerik dan *machine learning* yang berskala besar. Dalam TensorFlow, terdapat berbagai model dan algoritma *machine learning*, terutama dalam bidang *deep learning* (jaringan saraf). TensorFlow menyatukan berbagai model dan algoritma ini, dan menyediakan API tingkat tinggi yang didasarkan pada standar Keras API. Hal ini mempermudah dalam mendefinisikan dan melatih jaringan saraf, memudahkan proses pelatihan dan mendapatkan model yang optimal.
3. Keras adalah sebuah antarmuka pemrograman aplikasi (API) tingkat tinggi yang terintegrasi dengan TensorFlow 2.0. Ini adalah antarmuka yang mudah diakses dan sangat produktif untuk menyelesaikan masalah-masalah *machine learning*, terutama yang berkaitan dengan *deep* learning modern. Keras menonjol dengan tiga karakteristik utama: kesederhanaan, fleksibilitas, dan kekuatan.

4. Matplotlib adalah sebuah *library* yang digunakan untuk membuat visualisasi dari data numerik, memungkinkan pengguna untuk membuat berbagai jenis grafik seperti diagram lingkaran, histogram, *scatterplot*, dan grafik lainnya. Karena kemampuannya dalam menganalisis dan menggambarkan data, matplotlib sangat sering digunakan dalam proses analisis data.
5. Pandas, memberikan kemampuan kepada pengguna untuk menangani kumpulan data yang besar. Perpustakaan ini menyediakan alat untuk membaca dan menulis data, membersihkan dan mengubah data, serta berbagai fungsi lainnya.
6. Scikit-*learn*, terdiri dari berbagai fitur dan metode yang secara khusus dibuat untuk membantu pengguna dalam kebutuhan pembelajaran mesin. Menggunakan perpustakaan Numpy, terutama dalam operasi *array*. Selain itu, *library* ini berfokus pada penerapan dan evaluasi model dan menyediakan berbagai algoritma dan alat yang memudahkan proses pembelajaran mesin dari *preprocessing* hingga evaluasi model [16].
7. OpenCV (*open source computer vision library*), merupakan sebuah pustaka perangkat lunak yang ditujukan untuk pengolahan citra dinamis. OpenCV versi pertama diluncurkan pada tahun 1999. OpenCV menunjang banyak program, bisa menunjang seperti Linux ataupun Windows dan saat ini sudah bisa menunjang Android serta MacOS. *Library* OpenCV amat banyak digunakan untuk pemrosesan citra karena mempunyai sangat banyak algoritma, dimana algoritma itu dapat dipakai untuk mengolah citra mulai dari deteksi wajah, pengenalan wajah, *object tracking*, dan sebagainya. OpenCV juga merupakan *library open source* yang digunakan untuk bahasa C++, Python, Java, dan Matlab [17]. OpenCV dirancang khusus untuk memproses citra dengan tujuan memberikan kemampuan pengolahan visual komputer yang menyerupai kemampuan visual manusia. Selain itu OpenCV juga menyediakan modul-modul yang digunakan untuk mendeteksi objek menggunakan metode *computer vision*. *Library* OpenCV ini akan dimanfaatkan untuk pengambilan data bersamaan dengan pemanfaatan jupyter notebook.

2.2.5 Jupyter Notebook

Gambar 2.11 Logo Jupyter Notebook [16]

Jupyter Notebook adalah gagasan dari *Project* Jupyter, yang merupakan organisasi nirlaba yang didirikan oleh Fernando Pérez. Itu dibuat dengan tujuan mengembangkan perangkat lunak sumber terbuka dan menyediakan layanan yang memungkinkan berbagai bahasa berinteraksi satu sama lain secara efektif komputasi. Jupyter Notebook adalah aplikasi berbasis *web open source* yang memungkinkan pengguna membuat, mengedit, menjalankan, dan membagikan kode mereka dengan mudah. Aplikasi ini mendapatkan namanya dari bahasa utama yang didukungnya yaitu Julia, Python, dan R. Jupyter Notebook dikembangkan pada tahun 2014 sebagai *spin-off* dari aslinya IPython, yang merupakan *shell* perintah yang digunakan untuk melakukan pengkodean interaktif. Dengan dirilisnya Jupyter Notebook, IPython mendapati dirinya bersaing dengan Jupyter Notebook itu, sampai batas tertentu. Situs resmi *Project* Jupyter menyatakan bahwa Jupyter Notebook dapat mendukung lebih dari empat puluh bahasa pemrograman. Setiap proyek disimpan sebagai *notebook* yang terdiri dari beberapa sel kode, grafik, teks, dan persamaan yang dapat diubah dengan mudah [16].

Jupyter Notebook memiliki fitur utama berikut:

1. Setiap Notebook Jupyter adalah dokumen JSON. JSON adalah sebuah format data independen bahasa yang berasal dari JavaScript. Ia menggunakan teks yang dapat dibaca manusia untuk mengirimkan data berisi *array* atau pasangan atribut-nilai.
2. Setiap Notebook Jupyter biasanya disimpan dengan ekstensi .ipynb.
3. Notebook Jupyter memiliki gaya yang mirip dengan lainnya antarmuka yang berasal bertahun-tahun sebelumnya, termasuk *Maple* dan *Mathematica* (dari tahun 1980-an) dan *SageMath* (dari tahun 2000-an).

4. Notebook Jupyter dirilis dengan modifikasi Lisensi BSD, yang memberikan pengguna minimum keterbatasan dalam penggunaan dan distribusi perangkat lunak.
5. Notebook Jupyter dapat dengan mudah dibagikan kepada orang lain melalui email, Dropbox, GitHub, dan Jupyter Notebook.
6. Jupyter Notebook saat ini sepenuhnya gratis digunakan, dan ini dimaksudkan agar tetap gratis bagi siapa saja yang menggunakannya gunakan kapan saja.

Dari beberapa fitur tersebut maka penulis memutuskan untuk menggunakan Jupyter Notebook untuk mengambil data. Selain itu sejak dirilis, jupyter notebook terbukti menjadi alat yang ampuh untuk pemrograman, terutama untuk programmer tingkat tinggi. Memiliki kelancaran dan antarmuka yang mudah digunakan, sangat bagus bagi siapa saja yang baru mengenal pemrograman, serta sistem sangat memudahkan pengguna untuk penyimpanan kode.

2.2.6 Google Colab

Gambar 2.12 Logo Google Colab [18]

Pengaplikasian *deep learning* kini makin meningkat dan hadir dalam berbagai aspek kehidupan sehari-hari. Namun, nyatanya dalam mengaplikasikannya memerlukan komputasi berat dengan data besar dan biasanya memanfaatkan GPU untuk mempercepat prosesnya. Sayangnya, penggunaan perangkat keras GPU dalam lingkungan riset dan pengembangan menghadapi beberapa tantangan. Risiko seperti penggunaan yang tidak optimal, depresiasi perangkat keras, dan kegagalan sistem merupakan kekhawatiran utama. Selain itu, biaya pemeliharaan, energi, dan sumber daya manusia untuk mengoperasikan perangkat keras ini juga signifikan.

Dalam upaya untuk mempermudah akses dan pembelajaran *deep learning*, google memperkenalkan Google Colaboratory atau Colab, sebuah layanan *cloud* gratis yang menyediakan lingkungan komputasi dengan GPU yang kuat dan sudah

dikonfigurasi sepenuhnya dengan pustaka AI utama. Colab terintegrasi dengan google drive, membuatnya sangat mudah diakses dan digunakan oleh peneliti dan pelajar tanpa biaya.

Pada penelitian tahun 2019 oleh Carneiro T et al., [18] mengenai "*Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Aplications*", mendapatkan hasil penelitian yang menunjukkan bahwa *runtime* Colaboratory dapat mempercepat pelatihan model *deep learning*, bahkan mengungguli kinerja 20 inti fisik pada server Linux. Penelitian juga menyertakan beberapa studi kasus yang menunjukkan bagaimana Google Colab digunakan dalam berbagai aplikasi *deep learning*, dari klasifikasi gambar hingga pengenalan suara. Studi kasus ini menyoroti efektivitas Colab dalam mempercepat pengembangan dan pelatihan model. Namun, penelitian ini juga mencatat beberapa batasan seperti keterbatasan alokasi memori dan durasi sesi yang terbatas.

Berikut kelebihan dan kekurangan Google Colab:

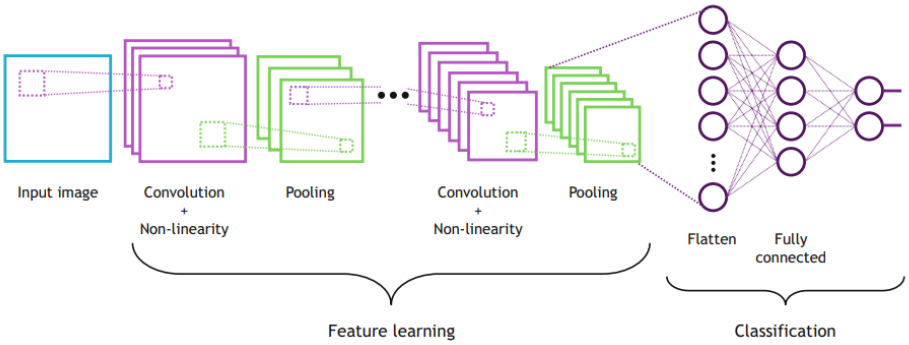
- Kelebihan:
    1. Akses gratis ke GPU: Pengguna dapat mengakses GPU tanpa biaya, yang sangat menguntungkan bagi yang memiliki anggaran terbatas.
    2. Integrasi dengan google drive: Colab terintegrasi dengan google drive, memungkinkan penyimpanan dan pengelolaan file yang mudah.
    3. Kemudahan kolaborasi: Colab memungkinkan kolaborasi *real-time*, mirip dengan google docs, yang sangat memudahkan dalam proyek-proyek tim.
- Kekurangan:
    1. Batasan sumber daya: Karena alokasi memori dan durasi sesi yang terbatas, pengguna mungkin mengalami keterbatasan dalam mengerjakan proyek-proyek besar atau kompleks.
    2. Ketergantungan pada koneksi internet: Sebagai *platform* berbasis *cloud*, kinerja Colab sangat tergantung pada kualitas koneksi internet pengguna.

2.2.7 *Convolutional Neural Network* (CNN)

Pada tahun 1998, dalam sebuah paper yang ditulis oleh Bengio, Le Cun, Bottou, dan Haffner, Convolutional Neural Network (CNN) diperkenalkan untuk pertama kalinya. Mereka memperkenalkan LeNet-5, yang merupakan CNN

pertama mereka, yang mampu mengklasifikasikan angka yang ditulis secara manual dengan tangan. CNN adalah jenis jaringan saraf tiruan yang digunakan untuk pengenalan dan pemrosesan gambar, serta untuk memproses data yang memiliki topologi jala atau grid-like. Penggunaan istilah Convolutional Neural Network (CNN) memiliki kekhasan karena menggambarkan jaringan yang menggunakan operasi matematika yang disebut konvolusi. Konvolusi, atau operasi konvolusional, adalah teknik yang menggunakan filter untuk mengekstraksi pola dari peta fitur (*feature maps*), dan filter ini diterapkan pada setiap bagian dari input sebelumnya [13].

CNN sering digunakan untuk memproses gambar atau pemandangan karena CNN dirancang khusus untuk memproses data yang memiliki struktur berbentuk kotak. Proses yang umum dilakukan pada gambar meliputi deteksi dan segmentasi objek. Gambar yang digunakan sebagai input dalam CNN biasanya direpresentasikan sebagai *array* dengan dua dimensi atau lebih.



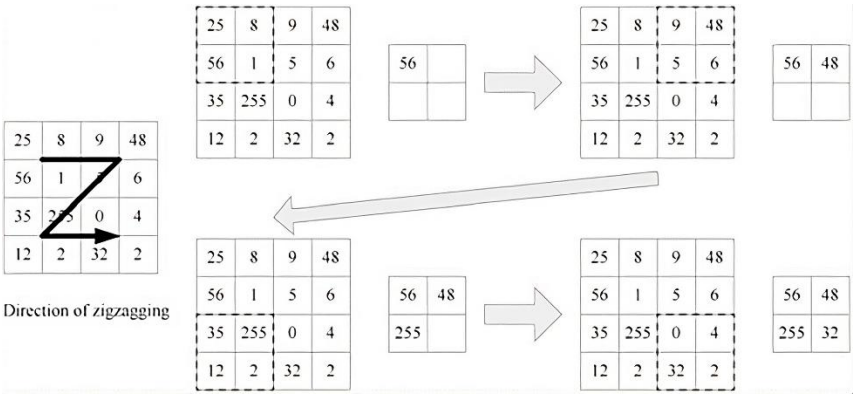
Gambar 2.13 Arsitektur Dasar *Convolutional Neural Network* (CNN) [19]

Berdasarkan gambar 2.13 ada beberapa *layer* yang ada pada CNN dan untuk penjelasan tiap *layer* adalah sebagai berikut:

1. Lapisan konvolusi. Dalam arsitektur CNN, salah satu komponen yang paling penting adalah *convolution layer*. Lapisan konvolusi sangat penting dan berperan vital dalam kinerja CNN. Istilah 'konvolusi' adalah menggabungkan dan/atau mengkombinasikan dua fungsi untuk membentuk fungsi ketiga. Lapisan konvolusi kadang-kadang disebut sebagai filter atau kernel, dan dapat dioperasikan pada data input untuk menghasilkan apa yang disebut peta fitur (*feature map*). CNN memiliki beberapa lapisan konvolusi dimana lapisan pertama mendeteksi fitur tingkat rendah dari gambar input, seperti warna,

orientasi gradien, dan tepi (*edges*). Lapisan-lapisan berikutnya fokus pada pendeteksian fitur tingkat menengah seperti bentuk-bentuk dalam gambar. Lapisan terakhir bertujuan untuk mendeteksi objek secara keseluruhan [20], [21].

2. *Pooling layer*, tugas utama dari lapisan *pooling* adalah melakukan *sub-sampling* pada peta fitur. Peta-peta ini dihasilkan setelah operasi konvolusi dilakukan. Dengan kata lain, pendekatan ini mengurangi ukuran peta fitur yang besar menjadi peta fitur yang lebih kecil. Secara bersamaan, metode ini mempertahankan sebagian besar informasi dominan (atau fitur) di setiap langkah tahap *pooling*. Mirip dengan operasi konvolusi, baik langkah (*stride*) maupun kernel diberi ukuran terlebih dahulu sebelum operasi *pooling* dilakukan. Terdapat berbagai metode pooling yang dapat diterapkan pada berbagai lapisan *pooling* dalam CNN. Beberapa metode ini termasuk *tree pooling*, *gated pooling*, *average pooling*, *min pooling*, *max pooling*, *global average pooling* (GAP), dan *global max pooling*. Di antara semua metode ini, yang paling populer dan sering digunakan adalah *max pooling*, *min pooling*, dan global *average pooling* (GAP) [20].



Gambar 2.14 Contoh Operasi *Pooling (Max Pooling)* [12]

3. *Fully connected layer*, lapisan ini sering disebut sebagai *dense layer*, biasanya berada di akhir setiap arsitektur CNN. Di dalam lapisan ini, setiap *neuron* terhubung dengan semua neuron dari lapisan sebelumnya, mengikuti pendekatan yang disebut *Fully Connected* (FC). Dense layer berfungsi sebagai pengklasifikasi dalam CNN. Metode ini didasarkan pada prinsip-prinsip jaringan saraf multilapis konvensional (*multiple-layer perceptron*), yang merupakan jenis jaringan saraf tiruan (ANN) dengan aliran data yang hanya

maju (*feed-forward*). *Input* untuk lapisan FC berasal dari lapisan pooling atau konvolusi terakhir dan berbentuk vektor, yang dibuat dari peta fitur setelah proses *flattening*. *Output* dari lapisan FC mewakili hasil akhir dari CNN [13][20]. Karena *output* yang dihasilkan adalah hasil akhir, nilai tersebut dapat dikategorikan berdasarkan tingkat akurasi klasifikasi sebagai berikut:

a. Klasifikasi sangat baik (*excellent classification*) dengan akurasi antara 0,90 dan 1,00.
b. Klasifikasi baik (*good classification*) dengan akurasi antara 0,80 dan 0,90.
c. Klasifikasi cukup (*fair classification*) dengan akurasi antara 0,70 dan 0,80.
d. Klasifikasi kurang baik (*poor classification*) dengan akurasi antara 0,60 dan 0,70.
e. Klasifikasi gagal (*failure classification*) dengan akurasi antara 0,50 dan 0,60.

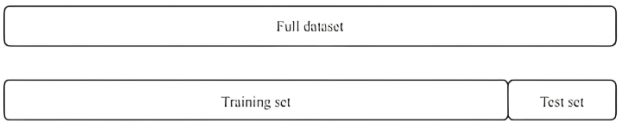
Selain 3 *layer* di atas, ada juga *layer* penting lainnya untuk membuat model atau arsitektur CNN salah satunya yaitu *flatten layer* merupakan komponen terakhir dari sebuah CNN. *Layer* ini mengubah struktur data dari format 2D atau *multi*-dimensi lainnya yang dihasilkan oleh lapisan konvolusi atau *pooling* menjadi format 1D. Hal ini dilakukan karena lapisan *fully connected* membutuhkan *input* dalam bentuk vektor satu dimensi [22]. Oleh karena itu, *flatten layer* ini merupakan elemen esensial dalam arsitektur CNN. Tanpa *layer* ini, peta fitur yang dihasilkan oleh lapisan konvolusi dan *pooling* tidak dapat dihubungkan dengan efektif ke lapisan *fully connected* yang digunakan untuk klasifikasi akhir.

Istilah lainnya selain *layer* yang berkaitan untuk pembuatan model CNN adalah epoch dan *learning rate*. Istilah epoch yang digunakan dalam *machine learning* dan *deep learning* adalah untuk menggambarkan satu siklus penuh melalui seluruh dataset pelatihan. Satu epoch pelatihan mengacu pada satu kali penyapuan melalui seluruh set pelatihan. Pelatihan memungkinkan jaringan untuk menangani set pelatihan yang sangat besar, dan juga untuk memperbarui bobot saat pengamatan baru masuk. Sebagai contoh, dengan dataset yang terdiri dari 1000 contoh dan penggunaan ukuran batch 100, maka satu epoch akan terdiri dari 10 iterasi (1000/100). Sedangkan, *learning rate* merupakan parameter yang menentukan ukuran langkah dalam proses pembaruan bobot jaringan selama

pelatihan. *Learning rate* dengan simbolnya berupa Γ_r untuk pembelajaran *batch* biasanya dianggap sebagai konstanta, dan juga dapat dioptimalkan melalui pencarian garis yang meminimalkan fungsi kesalahan pada setiap pembaruan. Dengan pembelajaran Γ_r harus berkurang menjadi nol saat iterasi $r \rightarrow \infty$. Fungsinya adalah untuk mengontrol seberapa besar pembaruan yang dilakukan pada bobot jaringan pada setiap iterasi [23].

2.2.8 Pembagian Dataset

Hal penting lainnya dalam pembuatan model adalah dataset. Pembagian dataset menjadi subset seperti *training* set dan *test* set sangat penting dalam melatih model *machine learning* ataupun *deep learning* karena beberapa alasan utama. Pertama, hal ini membantu mencegah *overfitting*, di mana model dapat menghafal data pelatihan tanpa mampu menggeneralisasi ke data baru. Kedua, dengan memisahkan *test* set dari data *train*, kita dapat memperoleh estimasi yang lebih akurat tentang kinerja model pada data baru yang belum pernah dilihat sebelumnya. Pembagian dataset juga memungkinkan kita untuk mengukur kemampuan generalisasi model, yaitu seberapa baik model menangani data yang tidak ada dalam training set, memberikan indikasi tentang performa model di luar dataset yang telah dilihat. Ini juga membantu dalam mendeteksi bias dan variansi dalam model, di mana jika model berkinerja baik pada *training* set tetapi buruk pada *test* set, ini menunjukkan adanya variansi yang tinggi [12].



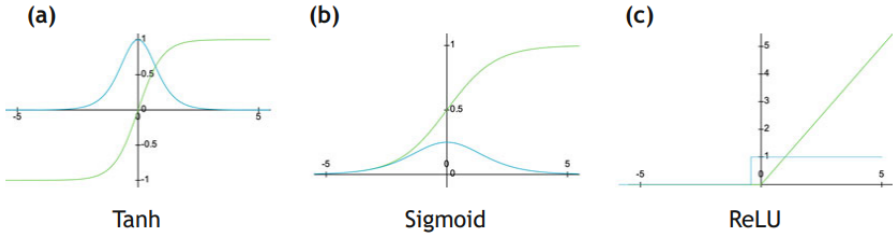
Gambar 2.15 Ilustrasi Pembagian Dataset Untuk *Training* dan *Test* [12]

Pada pembuatan model, proporsi data *train* ada di kisaran 70-80% dari total data, sementara data *test* berkisar 20-30%. *Training* set adalah kumpulan sampel yang digunakan untuk melatih model. Selama proses pelatihan, algoritma akan mencoba meningkatkan akurasi prediksi model berdasarkan sampel-sampel dalam *training* set. Ini menyebabkan model berkinerja lebih baik pada *training* set dibandingkan dengan data yang tidak diketahui. Untuk mengukur kemampuan generalisasi model, biasanya dipilih secara acak sebagian dari seluruh dataset

sebagai *test* set sebelum pelatihan dimulai, seperti yang ditunjukkan pada gambar 2.15. Sampel-sampel dalam *test* set tidak digunakan dalam pelatihan, sehingga mereka tetap tidak diketahui oleh model. Dengan demikian, kinerja model pada *test* set dapat diperkirakan mencerminkan kinerja model pada data yang belum pernah dilihat sebelumnya [12].

2.2.9 *Non-Linear Function*

Salah satu komponen yang juga penting dari *deep neural networks* adalah fungsi *non-linear*, juga disebut fungsi aktivasi. Fungsi ini mengubah sinyal masukan *linear* dari sebuah node menjadi keluaran *non-linear* untuk memudahkan pembelajaran polinomial orde tinggi. Kinerja *non-linear* dari lapisan aktivasi ini berarti pemetaan dari masukan ke keluaran akan bersifat *non-linear*; selain itu, lapisan-lapisan ini memberikan kemampuan kepada CNN untuk belajar hal-hal yang lebih rumit. Fungsi aktivasi juga memiliki kemampuan untuk diferensiasi, yang merupakan fitur yang sangat signifikan, karena memungkinkan untuk menggunakan propagasi kesalahan (*error backpropagation*) untuk melatih jaringan. Berikut adalah jenis-jenis fungsi aktivasi yang paling umum digunakan dalam CNN dan *deep neural networks* lainnya [19], [20].



Gambar 2.16 Perbedaan Grafik *Non-Linear Function* (Tanh, Sigmoid, dan ReLU) [19]

1. Sigmoid, masukan dari fungsi aktivasi ini adalah angka real, sementara keluarannya dibatasi antara nol dan satu. Kurva fungsi sigmoid berbentuk S dan dapat direpresentasikan secara matematis oleh persamaan berikut:

$$f(x)_{sigm} = \frac{1}{1 + e^{-x}} \tag{2.8}$$

2. Tanh, fungsi aktivasi ini mirip dengan fungsi sigmoid, karena masukannya adalah angka *real*, tetapi keluarannya dibatasi antara -1 dan 1. Representasi matematisnya dapat dilihat dalam persamaan:

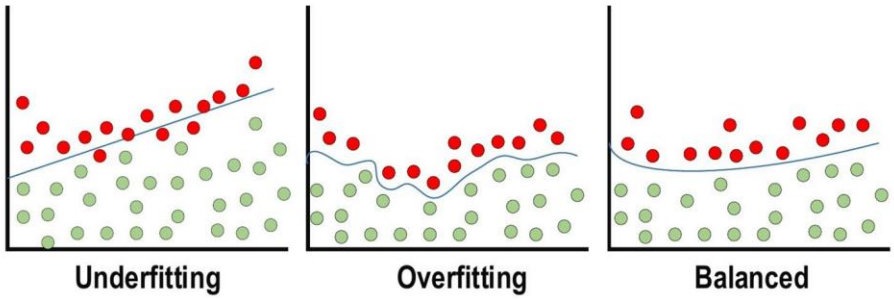
$$f(x)_{tanh} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (2.9)$$

3. ReLU adalah fungsi aktivasi yang paling umum digunakan dalam CNN. Fungsi ini mengubah semua nilai masukan menjadi angka positif. Salah satu keunggulan utama ReLU adalah beban komputasinya yang lebih rendah dibandingkan dengan fungsi aktivasi lainnya. Representasi matematis dari ReLU dinyatakan dalam bentuk persamaan:

$$f(x)_{ReLU} = max(0,x) \quad (2.10)$$

2.2.10 *Overfitting* dan *Underfitting*

Model DL (*Deep Learning*) memiliki kemungkinan yang sangat tinggi untuk mengalami *overfitting* atau bahkan *underfitting* pada tahap pelatihan karena sejumlah besar parameter yang terlibat yang berkorelasi secara kompleks. Situasi semacam ini mengurangi kemampuan model untuk mencapai kinerja yang baik pada data uji [20].



Gambar 2.17 Ilustrasi Grafik *Underfitting*, *Overfitting*, dan *Balanced* [20]

- Definisi 1: *Underfitting* terjadi ketika model gagal mencapai *loss* yang cukup rendah pada set pelatihan.
- Definisi 2: *Overfitting* terjadi ketika kesenjangan antara *loss* pelatihan dan *loss* uji terlalu besar.

Seperti yang dinyatakan dalam definisi 1 dan 2, *underfitting* terjadi ketika *Neural Network* (NN) tidak mampu menangkap atau merepresentasikan karakteristik penting dari data pelatihan dengan baik. Hal ini menyebabkan model tidak dapat memetakan target dengan akurat, menghasilkan tingkat akurasi yang rendah dan nilai *loss* yang tinggi pada data pelatihan, validasi, dan pengujian. Beberapa faktor yang dapat menyebabkan *underfitting* termasuk arsitektur NN yang terlalu sederhana, dengan jumlah lapisan tersembunyi atau parameter yang dapat

dilatih terlalu sedikit, sehingga NN tidak memiliki kapasitas yang cukup untuk menangkap kompleksitas data. Selain itu, *underfitting* juga bisa terjadi jika NN belum dilatih cukup lama untuk memahami karakteristik atau pola dalam data..

Sebaliknya, *overfitting* terjadi ketika NN terlalu memfokuskan pada data pelatihan sehingga belajar terlalu spesifik pada pola dalam data pelatihan dan gagal untuk umum ke data baru. Ini sering menyebabkan performa yang baik pada data pelatihan tetapi buruk pada data validasi atau pengujian. *Overfitting* dapat disebabkan oleh beberapa faktor, termasuk arsitektur NN yang terlalu kompleks dengan jumlah parameter yang sangat besar dibandingkan dengan jumlah data pelatihan, sehingga model terlalu menyesuaikan diri dengan fitur-fitur spesifik dari set pelatihan dan mengabaikan fitur-fitur umum. Selain itu, jika karakteristik data pelatihan berbeda signifikan dari data validasi atau pengujian, NN mungkin tidak mampu mempelajari pola yang generalis untuk semua set data, mengakibatkan *overfitting* [24].

2.2.11 *Regularization to* CNN

Regularisasi dalam konteks pembelajaran mesin dan *deep learning* merujuk pada teknik-teknik yang digunakan untuk mencegah model dari *overfitting* dan *underfitting*. Model disebut *overfitted* ketika ia berkinerja sangat baik pada data pelatihan namun gagal pada data uji (data yang tidak terlihat sebelumnya. Sebaliknya, model yang disebut *underfitted* terjadi ketika model tidak belajar cukup dari data pelatihan. Model disebut "*just-fitted*" jika berkinerja baik pada kedua data pelatihan dan data uji [20]. Berbagai konsep intuitif digunakan untuk membantu regularisasi guna menghindari *overfitting* dalam penelitian ini adalah;

1. *Dropout*, teknik yang banyak digunakan untuk meningkatkan generalisasi model. Selama setiap epoch pelatihan, *neuron-neuron* dipilih secara acak untuk dihapus (*dropped*). Dengan melakukan ini, kekuatan seleksi fitur didistribusikan secara merata di seluruh kelompok *neuron*, dan juga memaksa model untuk belajar berbagai fitur independen.
2. Normalisasi *Batch* (*Batch Normalization*), metode ini memastikan kinerja dari aktivasi *output*. Kinerja ini mengikuti distribusi *Gaussian* satuan. Dengan mengurangi *mean* dan membagi dengan deviasi standar, *output* pada setiap

lapisan dinormalisasi. Meskipun ini dapat dianggap sebagai tugas pra-pemrosesan pada setiap layer dalam jaringan, metode ini juga dapat dibedakan dan diintegrasikan dengan jaringan lain.

2.2.12 *Optimizer Selection*

Selain fungsi aktivasi, fungsi *loss* adalah salah satu elemen utama dari *neural network*. Ini adalah fungsi yang merepresentasikan kesalahan untuk prediksi yang diberikan [19]. Pada proses pembelajaran CNN (*Convolutional Neural Networks*), terdapat dua masalah utama dalam proses ini: masalah pertama adalah pemilihan algoritma pembelajaran (*optimizer*), sedangkan masalah kedua adalah penggunaan berbagai peningkatan (seperti AdaDelta, Adagrad, dan momentum) bersama dengan algoritma pembelajaran untuk meningkatkan hasil [20].

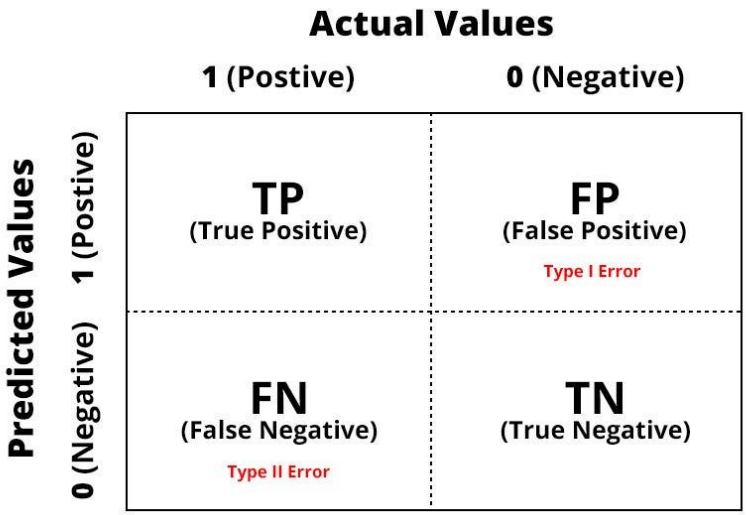
Ada beberapa alternatif *optimizer* yang sering digunakan saat ini, yaitu: *Batch Gradient Descent*, *Stochastic Gradient Descent*, *Stochastic Gradient with Momentum*, *AdaGrad*, *RMSProp*, *Mini-batch Gradient Descent*, *Momentum*, *Adaptive Moment Estimation* (Adam), dan lainnya. Dalam penelitian ini, *optimizer* yang akan digunakan adalah Adam. Adam sendiri merupakan teknik optimisasi atau algoritma pembelajaran lain yang banyak digunakan, sehingga mewakili tren terbaru dalam optimisasi *deep learning*. Ini direpresentasikan oleh matriks Hessian, yang menggunakan turunan orde kedua. Adam adalah strategi pembelajaran yang dirancang khusus untuk melatih *deep neural networks*. Lebih efisien dalam penggunaan memori dan membutuhkan daya komputasi yang lebih sedikit adalah dua keunggulan dari Adam. Mekanisme Adam adalah menghitung LR adaptif untuk setiap parameter dalam model. Ini menggabungkan kelebihan dari Momentum dan RMSprop. Ini menggunakan gradien kuadrat untuk menyesuaikan laju pembelajaran seperti RMSprop dan mirip dengan momentum dengan menggunakan rata-rata bergerak dari gradien [20].

Penggunaan Adam sebagai *optimizer* dalam penelitian ini, penulis pertimbangkan setelah mengetahui hasil penelitian oleh Al Rival M et al., [25] pada tahun 2022, dengan topik penelitiannya "Klasifikasi Isyarat Bahasa Indonesia Menggunakan Metode *Convolutional Neural Network*". Penelitian tersebut mengenai membandingkan penerapan arsitektur AlexNet dan VGG-16 serta

membandingkan optimizer Adam dan SGD pada kedua arsitektur. Didapatkan hasilnya bahwa penggunaan optimizer Adam lebih unggul dari pada SGD pada kedua arsitektur yaitu AlexNet dan VGG-16. Oleh karena itu, penulis memutuskan Adam sebagai *optimizer* pada penelitian ini.

2.2.13 *Confusion Matrix*

Confusion matrix digunakan untuk menilai performa algoritma *machine learning* atau *deep learning* dalam klasifikasi data. Tabel ini memberikan gambaran terperinci mengenai berapa banyak data yang diklasifikasikan dengan benar dan salah oleh model. Dari hasil klasifikasi yang ditampilkan dalam tabel ini, kita dapat menghitung berbagai matrik evaluasi seperti akurasi, presisi, dan *recall* [26]. Berikut adalah contoh tabel *confusion matrix*:



Gambar 2.18 Contoh Tabel *Confusion Matrix* [26]

Dalam *confusion matrix*, ada empat hasil keputusan yang mungkin terjadi: *True Positive*, *True Negative*, *False Positive*, dan *False Negative*, seperti yang ditunjukkan pada gambar 2.18. Arti dari 4 parameter tersebut, yaitu:

- *True Positive* (TP): Kondisi ini terjadi ketika data sebenarnya benar dan model juga memprediksikannya sebagai benar.
- *True negative* (TN): Ini adalah situasi di mana data sebenarnya salah dan model juga memprediksikannya sebagai salah.
- *False positive* (FP): Ini terjadi ketika data sebenarnya salah, tetapi model memprediksikannya sebagai benar.

- *False negative* (FN): Kondisi ini muncul ketika data sebenarnya benar, tetapi model memprediksikannya sebagai salah.

Dari nilai-nilai yang terdapat dalam *confusion matrix*, kita dapat menghitung beberapa matrik evaluasi seperti akurasi, *recall*, dan presisi. Akurasi adalah rasio antara jumlah prediksi yang benar dengan total jumlah data yang diprediksi. *Recall* adalah rasio antara jumlah prediksi benar positif dan total jumlah data yang sebenarnya benar. Presisi adalah rasio antara jumlah prediksi benar positif dan total jumlah hasil yang diprediksi sebagai positif [27].

- Nilai akurasi dapat dihitung dengan menggunakan persamaan berikut.

$$\text{Akurasi} = \frac{\boldsymbol{The\ number\ of\ true\ predictions}}{\boldsymbol{The\ number\ of\ predictions}} \cdot 100\%$$

$$= \frac{\mathbf{TP + TN}}{\mathbf{TP + FP + FN + TN}} \cdot 100\% \quad (2.4)$$

- Nilai *recall* dapat dihitung dengan menggunakan persamaan berikut.

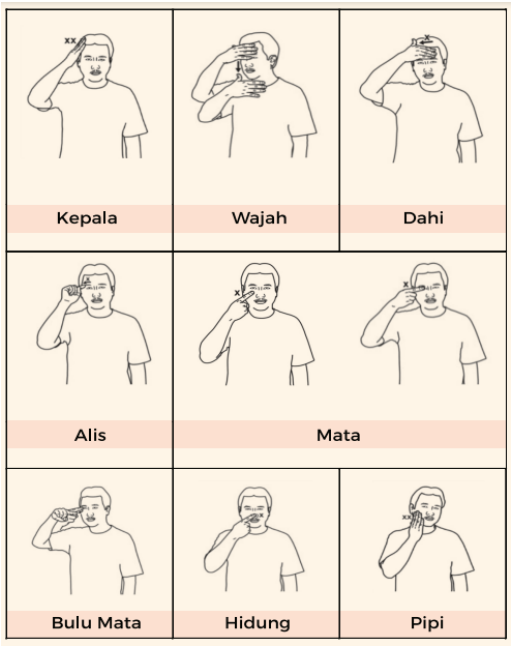
$$\textit{Recall} = \frac{\mathbf{TP}}{\mathbf{TP + FN}} \quad (2.5)$$

- Nilai presisi dapat dihitung dengan menggunakan persamaan berikut.

$$\text{Presisi} = \frac{\mathbf{TP}}{\mathbf{TP + FP}} \quad (2.6)$$

2.2.14 Bahasa Isyarat

Manusia secara alamiah merupakan makhluk sosial yang selalu berinteraksi, berkomunikasi, dan bersosialisasi dengan sesama manusia. Salah satu medium utama komunikasi antarmanusia adalah bahasa. Sebagian besar masyarakat berkomunikasi secara lisan, menggunakan bahasa *verbal*. Namun, tidak semua individu menggunakan bahasa lisan dalam komunikasi mereka. Contohnya adalah individu dengan disabilitas seperti tuna rungu dan tuna wicara. Mereka menggunakan bahasa isyarat sebagai media komunikasi mereka.



Gambar 2.19 Contoh Bahasa Isyarat Indonesia (BISINDO) [28]

Bahasa isyarat adalah metode komunikasi yang digunakan oleh individu dengan kebutuhan khusus seperti tuna rungu dan tuna wicara. Berbeda dengan komunikasi *verbal* yang menggunakan suara, bahasa isyarat melibatkan gerakan manual, bahasa tubuh, dan ekspresi wajah. Penggunaan bahasa isyarat oleh penyandang tuna rungu dan tuna wicara saat berinteraksi dengan orang tanpa disabilitas adalah sebuah masalah sosial yang sering terjadi. Bahasa komunikasi dapat dibagi menjadi dua jenis: bahasa *verbal*, yang melibatkan penggunaan kata-kata lisan atau tertulis, dan bahasa *nonverbal*, yang melibatkan ekspresi wajah, gerakan tubuh, dan gerakan tangan atau jari, seperti yang terjadi dalam penggunaan bahasa isyarat [29].

2.2.15 Tunarungu-wicara

Pada dasarnya, manusia yang lahir memiliki hak hidup yang sama, baik dalam hukum, Pendidikan, maupun interaksi sosial di masyarakat. Namun, pada kenyataannya beberapa atau bahkan sebagian orang yang tidak mendapat manfaat dari hak-hak tersebut karena status sosial mereka hingga perbedaan fisiologis. Orang-orang yang masuk dalam kategori penyandang cacat (disabilitas) karena termasuk salah satu kelompok yang tidak dapat memperoleh manfaat yang sama dari hak yang disediakan untuk mereka.

Tunarungu adalah individu yang mengalami hilangnya kemampuan mendengar, baik secara parsial maupun total, karena kerusakan atau tidak berfungsinya alat pendengaran mereka. Hal ini mengakibatkan mereka tidak mampu menggunakan alat pendengarannya dalam aktivitas sehari-hari, dan memiliki dampak yang kompleks pada kehidupan mereka. Gangguan tuna wicara merupakan suatu kondisi realitas sosial yang tidak bisa dihindari di masyarakat. Orang-orang ini tidak mampu berkomunikasi dengan baik seperti manusia normal. Tunarungu-bicara adalah istilah yang dikaitkan satu sama lain. Keadaan ini merupakan hubungan khusus antara kemampuan menyimak (mendengar) dan kemampuan berbicara. Ciri khas anak tunarungu tidak bisa dilihat dari penampilan fisiknya. Sekilas, anak tunarungu dan anak normal tidak ada perbedaannya. Namun akibat dari ketulian, terdapat ciri khas yang dapat diamati, seperti dari segi kecerdasan, kemampuan berbahasa dan berbicara, serta perasaan emosional dan sosial [30]. Penyandang tunarungu mengalami kesulitan berkomunikasi. Oleh karena itu, penulis melakukan penelitian mengenai ”Klasifikasi Bahasa Isyarat Menggunakan CNN Pada Python Untuk Pembelajaran”. Hal tersebut sesuai dengan tujuan penulis untuk mengembangkan sistem yang cocok digunakan dalam konteks pembelajaran dan pengembangan keterampilan bahasa isyarat, sehingga dapat sedikit membantu kaum dengar mempelajari bahasa isyarat yang kedepannya diharapkan dapat melakukan komunikasi dengan penyandang tunarungu atau teman tuli untuk kehidupan interaksi sosial.