**Landasan Teori**

**Inget 3.8nya gaada**

**3.1. Demensia**

Demensia adalah penurunan kemampuan berpikir, mengingat, dan bernalar yang mengganggu aktivitas sehari-hari seseorang. Tingkat keparahannya bervariasi, mulai dari tahap ringan, di mana pengaruhnya masih kecil, hingga tahap berat, di mana seseorang perlu bantuan penuh dari orang lain untuk melakukan kegiatan dasar sehari-hari. Penyebab demensia dapat berbeda tergantung perubahan yang terjadi pada otak. Beberapa bentuk demensia antara lain *Alzheimer*, *Lewy body dementia*, *frontotemporal disorder*s, dan *vascular dementia* (National Institute on Aging, 2023).

**3.2. Alzheimer’s Disease**

Alzheimer merupakan penyebab terbanyak dari kasus demensia, menyumbang mayoritas dari seluruh kasus demensia yang terjadi. Penyakit ini menyebabkan penurunan kemampuan kognitif yang signifikan, seperti kemampuan berpikir, mengingat, dan bernalar, yang akhirnya mengganggu aktivitas sehari-hari penderitanya. Penurunan ini terjadi karena sel-sel saraf (neuron) di bagian otak yang terlibat dalam fungsi kognitif telah rusak dan tidak lagi berfungsi normal (Sianturi, 2021).

Para dokter menggunakan berbagai metode untuk menentukan apakah seseorang mengalami kehilangan ingatan akibat Alzheimer. Pertama, mereka akan mengajukan pertanyaan tentang kesehatan pasien, obat yang dikonsumsi, pola makan, riwayat medis, aktivitas sehari-hari, dan perubahan perilaku kepada pasien atau orang terdekatnya. Selanjutnya, mereka melakukan tes ingatan dan menilai kemampuan pasien dalam memecahkan masalah, fokus, perhatian, menghitung, dan berbahasa. Tes medis standar juga dilakukan untuk mengidentifikasi kemungkinan penyebab lain dari gejala tersebut. Terakhir, dokter mungkin akan merekomendasikan pemindaian otak (National Institute on Aging, 2023).

Penyusutan ukuran (artrofi) pada bagian otak seperti hippocampus, pembesaran ruang kosong (ventrikel), dan penyusutan korteks adalah tanda-tanda yang sensitif untuk Alzheimer (Pini et al., 2016). Untuk memeriksa hal ini, dokter menggunakan teknologi pencitraan canggih seperti CT scan, MRI, atau PET scan. Teknologi-teknologi ini membantu dokter melihat perubahan struktur otak untuk mengonfirmasi kehadiran dan perkembangan Alzheimer, serta memastikan bahwa gejala yang dialami tidak disebabkan oleh masalah kesehatan lain (Gunawardena et al., 2017).

**3.3. Kecerdasan Buatan**

Kecerdasan Buatan (AI) adalah sistem komputer yang dapat menjalankan tugas-tugas yang biasanya memerlukan kecerdasan manusia. Teknologi ini memungkinkan sistem untuk membuat keputusan dengan menganalisis dan menggunakan data yang tersedia. Proses AI melibatkan pembelajaran, penalaran, dan koreksi diri, mirip dengan cara manusia menganalisis sebelum membuat keputusan (Kurniawan et al., 2023). Kecerdasan buatan dapat diaplikasikan ke berbagai bidang diantaranya adalah *gaming, Natural Language Processing, Expert System, Vision System, Speech Recognition,* dan *Handwriting Recognition* (Shoumi et al., 2022).

**3.4. Pembelajaran Mesin (*Machine Learning*)**

Machine learning adalah bagian dari domain kecerdasan buatan yang difokuskan pada penggunaan algoritma dan metode tertentu untuk memungkinkan komputer melakukan prediksi, pengenalan pola, dan klasifikasi dari data yang diberikan (Lestari & Rahayu, 2023). Inti dari pembelajaran mesin adalah kemampuan sistem untuk belajar dari pengalaman. Setelah diberikan sejumlah contoh pelatihan, sistem harus mampu membuat model umum yang bisa digunakan untuk mengambil keputusan pada kasus baru dengan akurasi yang memadai. Berdasarkan pendekatan ini, ada tiga metode pembelajaran dalam pembelajaran mesin (Amaratunga, 2020):

1. *Supervised Learning*

Sistem diberi contoh-contoh berlabel (set pelatihan) dan diminta membuat model yang bisa diterapkan pada kasus baru. Set pelatihan ini berfungsi sebagai data acuan yang digunakan oleh sistem untuk mengenali pola dan hubungan antara input dan output. Dengan menggunakan algoritma pembelajaran terawasi, sistem secara bertahap belajar untuk membuat prediksi atau keputusan berdasarkan data baru yang belum pernah dilihat sebelumnya. Proses ini melibatkan optimasi model untuk meminimalkan kesalahan prediksi, sehingga model yang dihasilkan dapat memberikan hasil yang akurat dan andal saat dihadapkan pada data baru di masa mendatang. Pembelajaran terawasi sangat berguna dalam berbagai aplikasi, seperti klasifikasi gambar, pengenalan suara, dan prediksi penyakit berdasarkan data medis. Beberapa metode pada algoritma ini adalah *decision tree*, *random forest*, *k-nearest neighbor*, dan *logistic regression* (Shoumi et al., 2022).

1. *Unsupervised Learning*

Sistem diberi contoh-contoh tanpa label dan diminta menemukan pola dalam data tersebut. Ini cocok untuk menemukan pola tersembunyi. Tanpa label, sistem harus mengidentifikasi kelompok atau struktur dalam data secara mandiri, sering kali menggunakan teknik seperti pengelompokan (clustering) atau pengurangan dimensi (dimensionality reduction). Metode ini sangat berguna dalam analisis data eksploratif, di mana peneliti ingin menemukan wawasan atau anomali yang sebelumnya tidak terdeteksi dalam dataset besar. Beberapa contoh algoritmanya antara lain *k-means clustering* dan algoritma apriori (Shoumi et al., 2022).

1. *Reinforcement Learning*

Sistem mengambil tindakan dan diberi hadiah (*reward*) atau hukuman (*punishment*) berdasarkan seberapa baik tindakan tersebut sesuai dengan situasi yang diberikan. Sistem belajar tindakan mana yang menghasilkan hadiah terbanyak dalam berbagai situasi. Salah satu algoritma pembelajaran mesin ini adalah *markov decision process* (Shoumi et al., 2022).

**3.5. *Computer Vision***

Computer Vision adalah teknologi otomatis yang digunakan oleh komputer untuk menganalisis gambar dan video agar dapat memahami dan mendapatkan informasi tentang objek tertentu. Ini adalah kemampuan komputer untuk melihat dan mengenali gambar, dengan ketepatan yang setara atau bahkan lebih baik dari penglihatan manusia (Mulya et al., 2023). Tugas utama visi komputer adalah tentang bagaimana membuat komputer memahami gambar digital serta data visual dari dunia nyata. Dalam menganalisis informasi dari masukan tersebut dan membuat keputusan melibatkan beberapa proses yaitu sebagai berikut (Arnita et al., 2022).

1. Pengolahan Sinyal/Citra

Pengolahan citra digital adalah manipulasi dan interprestasi digital dari citra dengan bantuan komputer. *Input* dari pengolahan citra adalah gambar, sedangkan *output*-nya adalah citra hasil pengolahan. Citra secara umum merupakan suatu gambar, foto ataupun berbagai tampilan dua dimensi yang menggambarkan suatu visualisasi objek.

1. Klasifikasi Citra

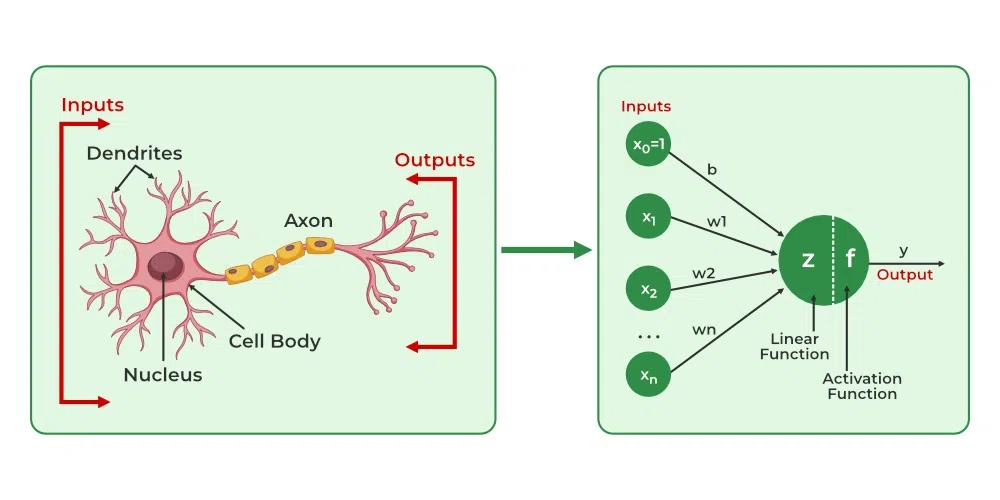
Klasifikasi citra adalah proses analisis data secara numeris yang bertujuan untuk mengidentifikasi objek atau mengenali pola. Pengenalan pola objek (spectral pattern recognition) mengevaluasi informasi objek berdasarkan ciri-ciri dalam citra penginderaan jauh untuk menginterpretasi citra digital. Sistem pengenalan pola ini melibatkan pengkategorian piksel dalam citra berdasarkan hubungan spasial antar piksel tersebut.

*Computer vision* mempunyai keterkaitan dengan beberapa bidang yaitu, *image processing* (pengolahan citra) dan *machine vision* (visi mesin). Ada kesamaan yang signifikan dalam berbagai teknik dan aplikasi yang mencakup tiga bidang ini. Hal ini menunjukan teknik dasar yang digunakan dan dikembangkan kurang lebih sama. Secara luas *computer vision* berhubungan dan diterapkan dengan bidang lain seperti *artificial intelligence (AI),* robotika, otomasi industry, pengolahan sinyal, optik fisik, *neurobiology*, dan lain-lain.

*Computer vision* sekarang ini telah sering digunakan untuk berbagai hal, contohnya saja mendeteksi wajah pada gambar (*face detection*, mengenali wajah (*facial expression recognition*) dan dalam prakteknya sering digunakan bersama dengan jaringan saraf tiruan (*artificial neural network*) (Dompeipen & Sompie, 2020).

**3.6. Jaringan Saraf Tiruan**

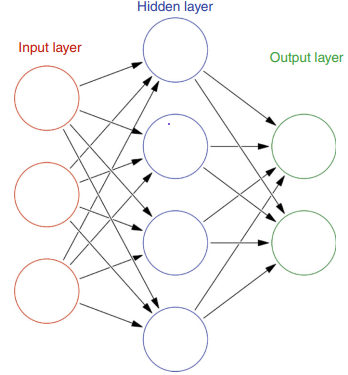
Jaringan saraf tiruan (*Artificial Neural Networks/ANN*) terinspirasi oleh model awal pemrosesan sensorik oleh otak. Jaringan saraf tiruan dapat dibuat dengan mensimulasikan jaringan neuron model dalam komputer. Dengan menerapkan algoritma yang meniru proses neuron asli, kita dapat membuat jaringan 'belajar' untuk menyelesaikan berbagai jenis masalah. Neuron ini menerima input dari sejumlah unit lain atau sumber eksternal, memberi bobot pada setiap input, dan menjumlahkannya (Krogh, 2008).



Gambar 3.1 Hubungan Neuron Biologis dan Buatan (Harkiran78, 2023)

Sistem jaringan saraf tiruan terinspirasi oleh struktur dan fungsi neuron manusia, di mana neuron model dalam jaringan saraf tiruan meniru cara neuron asli berkomunikasi dan memproses informasi di otak. Jika keduanya dibandingkan, maka masukan dalam ANN merepresentasikan dendrite, *nodes* (fungsi linear & aktivasi) adalah inti sel, *weights* adalah sinapsis, dan axon adalah keluaran dari sistem ANN. Nodes akan menerima nilai masukan dan akan memproses menggunakan fungsi aktivasi untuk menghasilkan nilai keluaran. Dalam jaringan saraf tiruan, fungsi aktivasi adalah fungsi matematika yang memungkinkan jaringan untuk mempelajari pola dan hubungan dalam data, serta memetakan input menjadi output pada (Harkiran78, 2023).

Secara umum jaringan saraf tiruan terdiri dari *input layer, hidden layer,* dan *output layer*. Dalam setiap *input* dan *hidden layer* terdapat neuron-neuron yang berbeda antara satu sama lain (Sumin & Prihantono, 2020). Neuron-neuron pada satu lapisan akan dihubungkan dengan lapisan-lapisan lainnya. Informasi yang didapatkan pada sebuah neuron akan disampaikan ke semua lapisan-lapisan yang ada, mulai dari lapisan masukan sampai dengan lapisan keluaran melalui lapisan tersembunyi (hidden layer).



Gambar 3.2 Arsitektur jaringan saraf tiruan (Zhang et al., 2019)

**3.7. Deep Learning**

Pembelajaran mendalam (Deep Learning) adalah subset dari pembelajaran mesin yang berfokus pada algoritma yang terinspirasi oleh pemahaman kita tentang cara kerja otak untuk memperoleh pengetahuan. Ini juga disebut sebagai pembelajaran terstruktur mendalam atau pembelajaran hierarkis. Pembelajaran mendalam membangun ide jaringan saraf tiruan dan mengembangkannya dengan memperdalam jaringan tersebut agar mampu mengonsumsi sejumlah besar data. Melalui jaringan yang lebih dalam, model pembelajaran mendalam memiliki kemampuan untuk mengekstraksi fitur dari data mentah dan "mempelajari" fitur-fitur tersebut sedikit demi sedikit di setiap lapisan, hingga membentuk pengetahuan tingkat tinggi tentang data tersebut (Amaratunga, 2020). Beberapa komponen penting dalam implementasi deep learning diantaranya adalah fungsi aktivasi, optimizer, dan loss function.

**3.7.1. Fungsi Aktivasi**

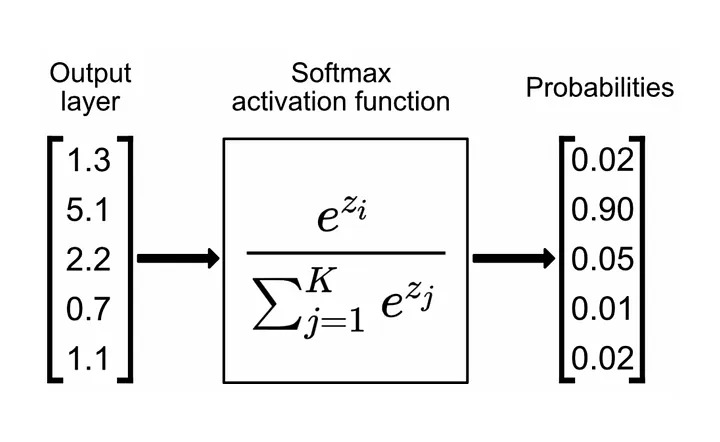
Dalam ranah machine learning dan deep learning, fungsi aktivasi memainkan peran penting dalam kemampuan jaringan saraf untuk membuat keputusan dan prediksi yang kompleks (Belagatti, 2024). Fungsi aktivasi digunakan untuk menghitung bobot masukan dan bias (jika ada) serta memutuskan apakah suatu neuron dapat diaktifkan atau tidak (Panneerselvam, 2024). Fungsi aktivasi yang sering digunakan diantaranya softmax dan ReLU.

1. Softmax

Fungsi softmax, yang biasanya diterapkan pada lapisan akhir model jaringan saraf untuk klasifikasi, mengubah skor output mentah (dikenal sebagai logits) menjadi probabilitas. Ini dilakukan dengan mengambil eksponensial dari setiap output dan menormalkan nilai-nilai tersebut dengan membaginya dengan jumlah dari semua eksponensial. Proses ini memastikan nilai output berada dalam rentang 0 hingga 1 dan jumlah totalnya sama dengan 1, sehingga nilai-nilai tersebut dapat diinterpretasikan sebagai probabilitas (Belagatti, 2024). Fungsi softmax dapat dinyatakan dengan persamaan berikut (Alzubaidi et al., 2021):

(3.1)

Pada persamaan tersebut, ekspresi merepresentasikan *output* saraf ke-i yang belum dinormalisasi pada lapisan sebelumnya. Ekspresi adalah jumlah seluruh *output* saraf pada lapisan sebelumnya. Hasil dari persamaan berikut adalah nilai dengan rentang 0−1. Berikut ilustrasi perhitungannya:



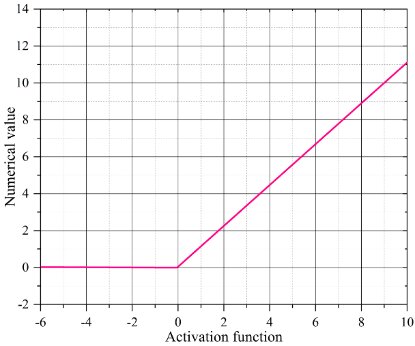
Gambar 3.3 Ilustrasi hasil perhitungan softmax(Belagatti, 2024)

1. ReLU

ReLU (Rectified Linear Unit) adalah fungsi aktivasi yang paling umum digunakan dalam model deep learning. Fungsi ini mengembalikan nilai 0 jika menerima input negatif, tetapi untuk setiap nilai positif (x), fungsi ini mengembalikan nilai (x) tersebut (Mkale, 2022). Keunggulan utama ReLU dibandingkan dengan fungsi aktivasi lainnya adalah beban komputasinya yang lebih ringan. Fungsi aktivasi ReLU dapat dituliskan sebagai berikut (Alzubaidi et al., 2021):

(3.2)

Berikut ilustrasi kurva dari fungsi ReLU:



Gambar 3.4 Kurva fungsi aktivasi ReLU (Jia, 2023)

**3.7.2. Optimizer**

Optimisasi (*optimizer*) adalah algoritma atau metode yang digunakan untuk meminimalkan fungsi kerugian (*loss function*) atau untuk memaksimalkan efisiensi produksi. *Optimizer* adalah fungsi matematika yang bergantung pada parameter yang dapat dipelajari model, yaitu bobot (*weights*) dan bias. *Optimizer* membantu menentukan bagaimana mengubah bobot dan laju pembelajaran jaringan saraf untuk mengurangi kerugian (Musstafa, 2021). Beberapa aspek yang diatur dan disesuaikan oleh *optimizer* adalah bobot dan laju pembelajaran. Bobot merupakan parameter yang menyimpan informasi tentang hubungan antar neuron. Laju pembelajaran menentukan seberapa besar perubahan yang diterapkan pada bobot di setiap iterasi.

Satu algoritma yang menonjol dalam hal melatih jaringan saraf tiruan adalah optimisasi Adam. Adam, singkatan dari *Adaptive Moment Estimation*, adalah algoritma dengan laju pembelajaran adaptif yang dirancang untuk mempercepat proses pelatihan jaringan saraf dalam-dalam dan mencapai konvergensi dengan cepat. Algoritma ini menyesuaikan laju pembelajaran setiap parameter berdasarkan riwayat gradiennya, sehingga membantu jaringan saraf belajar dengan efisien secara keseluruhan (Agarwal, 2023).

**3.7.3. Loss Function**

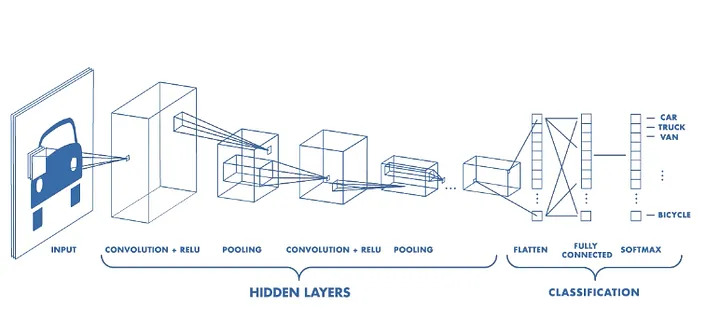
Dalam machine learning (ML), fungsi kerugian (loss function) digunakan untuk mengukur kinerja model dengan menghitung deviasi prediksi model dari prediksi yang benar, atau "ground truth". Jika prediksi model akurat, kerugiannya kecil. Jika prediksi tidak akurat, kerugiannya besar (Bergmann & Stryker, 2024). Jenis loss function yang biasanya digunakan dalam tugas klasifikasi adalah Cross-Entropy. Kerugian Cross-Entropy dapat dinyatakan dengan persamaan berikut:

(3.3)

Pada persamaan tersebut, simbol p menunjukkan probabilitas asli. Sedangkan q adalah probabilitas perkiraan yang dihasilkan oleh model.

**3.9. *Convolutional Neural Network* (CNN)**

CNN, atau *Convolutional Neural Network*, adalah jenis model *deep learning* yang dirancang khusus untuk memproses data dengan pola grid, seperti gambar. Model ini bekerja dengan menggunakan lapisan konvolusi untuk mengekstraksi fitur-fitur penting dari data input. Dengan kemampuan ini, CNN sangat efektif dalam tugas-tugas seperti pengenalan objek, klasifikasi gambar, dan analisis visual lainnya, karena mampu menangkap pola spatial dan temporal dalam data dengan lebih baik dibandingkan model-model deep learning lainnya (Yamashita et al., 2018).

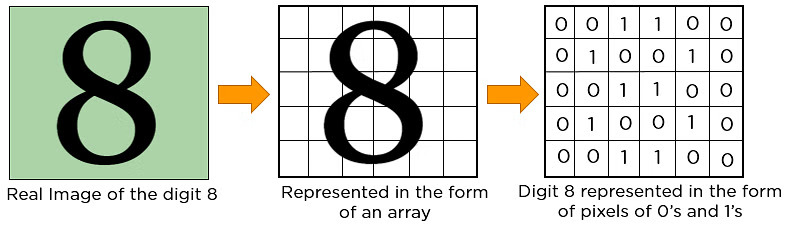


Gambar 3.5 Ilustrasi arsitektur CNN secara umum (Mishra, 2020)

CNN adalah sebuah sistem matematis yang terdiri dari tiga jenis lapisan utama: lapisan konvolusi, lapisan *pooling*, dan lapisan sepenuhnya terhubung (*fully-connected*). Lapisan konvolusi bertugas untuk menangkap fitur dari gambar dengan menggunakan filter yang bergerak di atas gambar, mendeteksi pola seperti tepi atau tekstur. Setelah itu, lapisan *pooling* mengurangi ukuran peta fitur sambil tetap menjaga informasi penting, biasanya dengan cara mengambil nilai maksimum atau rata-rata dari bagian-bagian peta fitur. Terakhir, lapisan sepenuhnya terhubung menghubungkan fitur-fitur yang telah diproses ke neuron-neuron pada lapisan output, menghasilkan hasil akhir seperti klasifikasi atau prediksi. Dengan gabungan ketiga lapisan ini, CNN bisa memproses data gambar, mengekstrak fitur penting, dan memberikan hasil yang akurat (Yamashita et al., 2018).

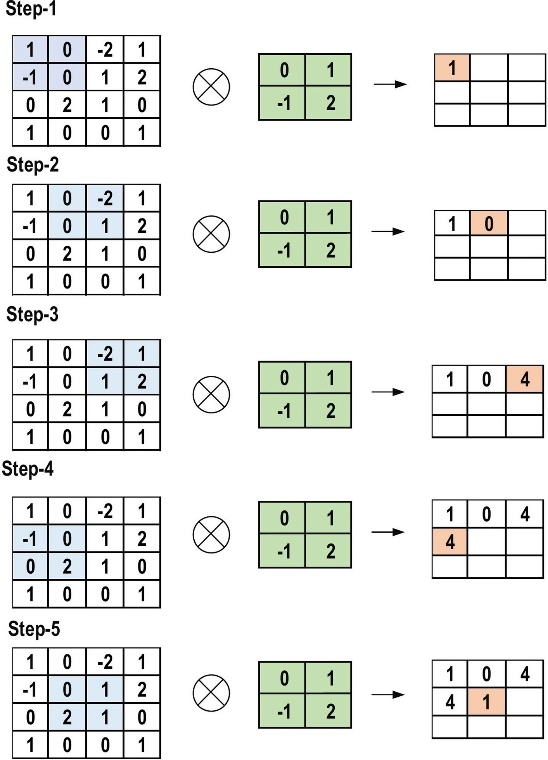
**3.10.1. *Convolution Layer***

Lapisan konvolusi (*convolution layer*) adalah bertugas untuk menangkap fitur dari gambar dengan menggunakan filter (kernel) yang bergerak di atas gambar, mendeteksi pola seperti tepi atau tekstur (Lina, 2019). Hasil dari proses konvolusi ini disebut sebagai *activation map* atau *feature map*.



Gambar 3.6 Komputer melihat gambar sebagai suatu array representasi piksel (Biswal, 2023)

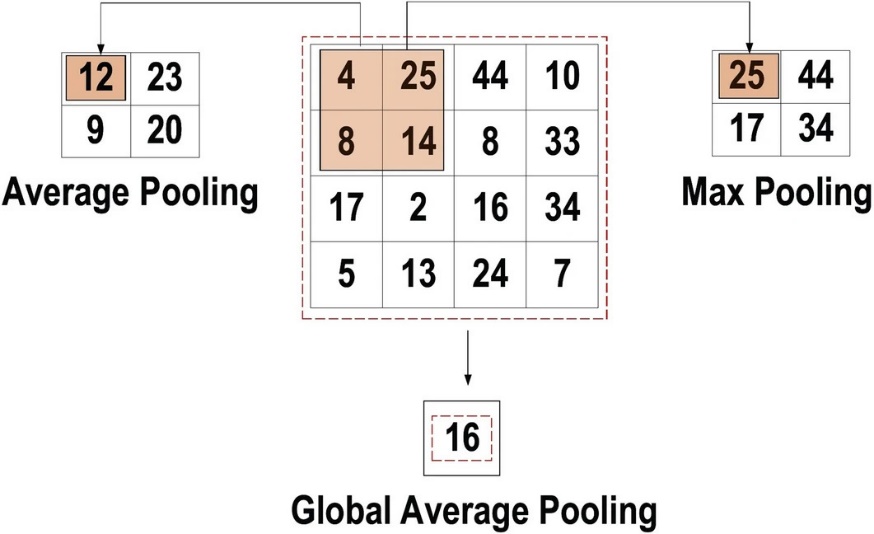
Ukuran kernel konvolusi dapat disesuaikan dengan kebutuhan spesifik dalam suatu tugas pemrosesan citra atau sinyal. Proses konvolusi dapat diilustrasikan dengan gambar di bawah ini.



Gambar 3.7 Proses konvolusi (Alzubaidi et al., 2021)

**3.10.2. *Pooling Layer***

*Pooling layer* bertugas untuk memperkecil ukuran *feature maps* yang dihasilkan dari proses konvolusi. Lapisan ini tetap mempertahankan sebagian besar informasi dominan pada setiap langkah *pooling*. Beberapa metode *pooling* yang paling sering digunakan diantaranya *max* *pooling*, *average* *pooling*, dan GAP (*global average pooling*) (Alzubaidi et al., 2021). Ketiga operasi tersebut memiliki perbedaan dalam cara mengambil nilai dari *feature map*. *Max pooling* mengambil nilai tertinggi dari wilayah kecil *feature map*, *average poolin*g mengambil nilai rata-rata dari wilayah kecil *feature map*, sedangkan GAP mengambil nilai rata-rata dari keseluruhan *feature map*.



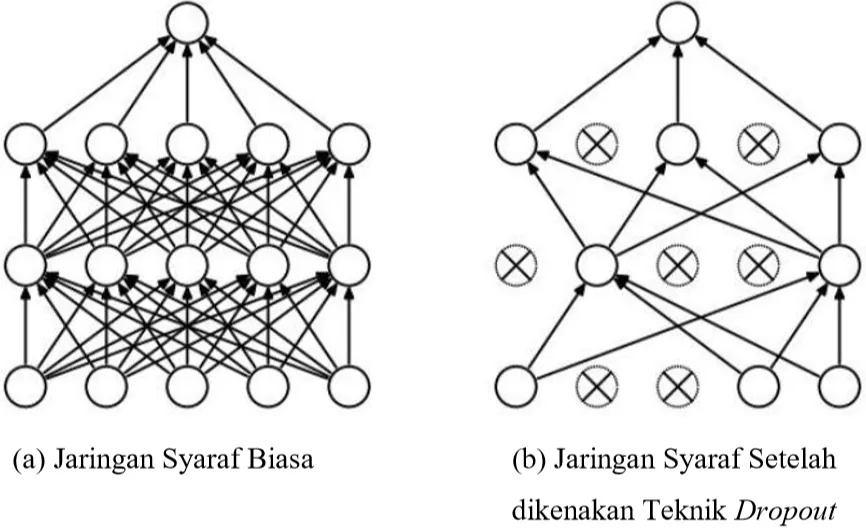
Gambar 3.8 Ilustrasi operasi max pooling, average pooling, dan global average pooling (Alzubaidi et al., 2021)

**3.10.3. *Batch Normalization***

*Batch normalization* merupakan suatu metode dalam *deep learning* yang membuat pelatihan model menjadi lebih cepat dan stabil (Huber, 2020). Fungsi utama dari Batch Normalization adalah untuk menormalkan input dari setiap lapisan dalam jaringan saraf sehingga output dari setiap lapisan tidak terlalu bervariasi dan lebih mudah diatur.

**3.10.4. *Dropout Regularization***

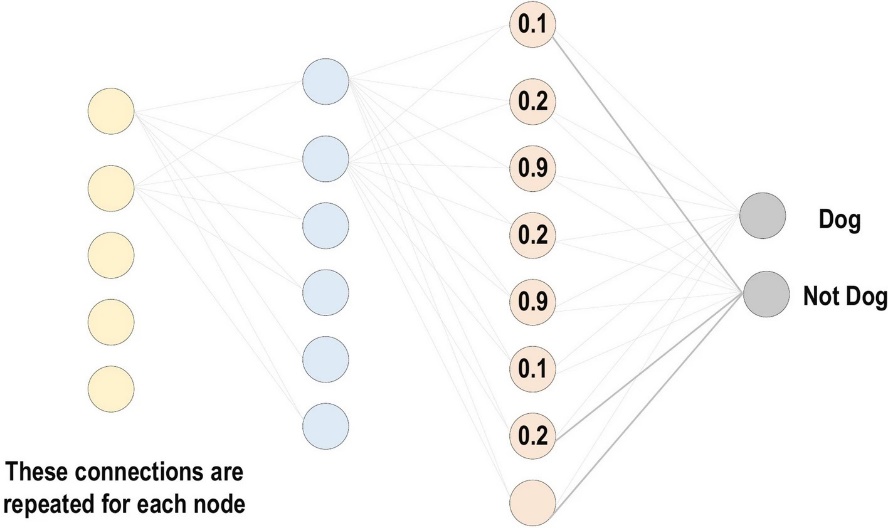
*Dropout* adalah suatu teknik regularisasi jaringan saraf yang bekerja dengan cara memilih beberapa neuron secara acak untuk tidak digunakan dalam proses pelatihan. Teknik ini bertujuan untuk mencegah terjadinya overfitting dan mempercepat proses pelatihan (Lina, 2019). Neuron yang dihilangkan dapat berasal dari lapisan *hidden* maupun *visible*. Melatih model menggunakan *dropout* menghasilkan lebih sedikit kesalahan generalisasi akibat model yang terlalu menghafal data pelatihan (Srivastava, 2013).



Gambar 3.9 Ilustrasi dropout regularization (Lina, 2019)

**3.10.5. *Fully-Connected Layer***

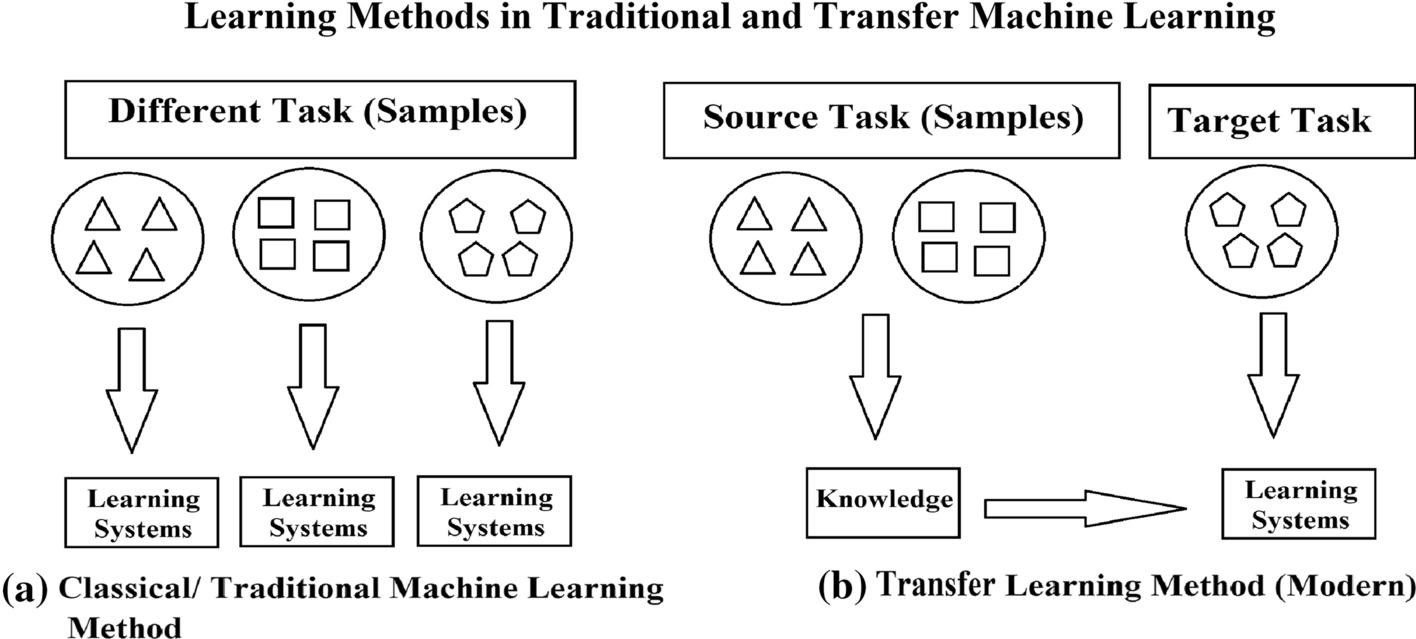
Lapisan sepenuhnya terhubung (*fully-connected layer*), adalah lapisan di dalam jaringan saraf di mana setiap neuron terhubung dengan semua neuron di lapisan sebelumnya (Alzubaidi et al., 2021). Ini berarti bahwa setiap neuron di lapisan ini menerima informasi dari seluruh neuron di lapisan sebelumnya, memungkinkan integrasi lengkap dari semua fitur yang telah diekstraksi oleh lapisan sebelumnya. Lapisan ini biasanya terletak di akhir arsitektur model. Pada lapisan ini terjadi proses klasifikasi gambar sesuai kelas berdasarkan identifikasi fitur pada lapisan-lapisan sebelumnya.



Gambar 3.10 Ilustrasi fully-connected layer (Alzubaidi et al., 2021)

**3.10. *Transfer Learning***

*Transfer learning* (TL) adalah teknik yang melatih model saat ini dengan menggunakan model yang telah dilatih sebelumnya (*pre-trained model*) pada tugas-tugas terkait yang serupa (Hosna et al., 2022). Dalam TL, *pretrained model* berfungsi sebagai dasar untuk memulai pelatihan pada tugas baru, memungkinkan model untuk memanfaatkan pengetahuan yang sudah ada sehingga pelatihan menjadi lebih efisien dan cepat.

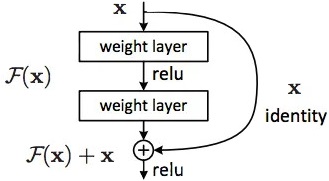


Gambar 3.11 Traditional Machine Learning vs Transfer Learning (Hosna et al., 2022)

Dalam *transfer learning*, model yang telah dilatih sebelumnya (*pre-trained model*) dapat digunakan sebagai titik awal untuk melatih model pada tugas baru yang serupa. Proses ini melibatkan penggunaan seluruh atau sebagian dari model yang sudah ada, tergantung pada teknik pemodelan yang dipilih. Setelah diadopsi, model ini dapat disesuaikan atau disempurnakan lebih lanjut berdasarkan data *input*-*output* yang spesifik untuk tugas baru tersebut. Pendekatan ini memungkinkan pemanfaatan pengetahuan yang telah diperoleh dari tugas sebelumnya, sehingga mempercepat dan meningkatkan efisiensi pelatihan model pada tugas baru.

**3.11. *Residual Network* (ResNet)**

Residual Network (ResNet) adalah salah satu arsitektur deep learning yang sangat populer, diperkenalkan oleh Kaiming He dan rekan-rekannya dari Microsoft Research pada tahun 2015 (He et al., 2015). ResNet muncul sebagai solusi untuk masalah degradasi performa pada jaringan neural yang sangat dalam, di mana penambahan lapisan justru mengurangi akurasi. Inti dari ResNet adalah konsep *residual learning*, yang memungkinkan jaringan untuk mempelajari perbedaan atau *residual* antara *input* dan *output* yang diharapkan, alih-alih mempelajari fungsi langsung.



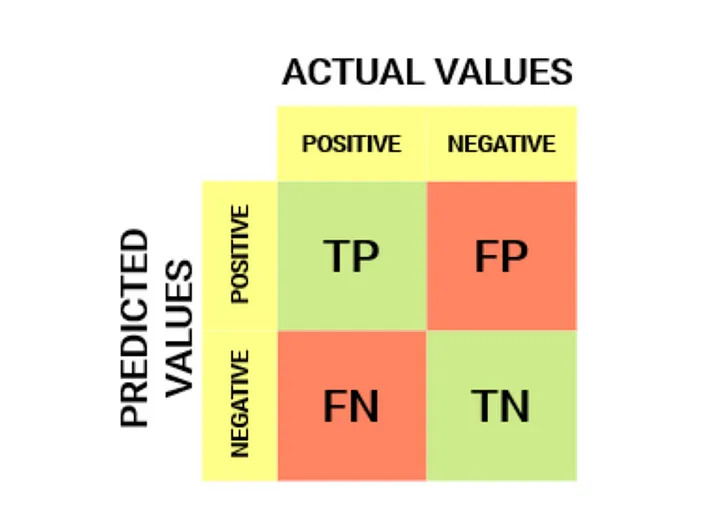
Gambar 3.12 Blok residual learning dengan identity mapping (He et al., 2015)

Dalam Residual Network (ResNet), konsep utama yang perlu dipahami adalah *Skip Connection* dengan pemetaan identitas (*identity mapping*). Pemetaan identitas ini adalah fitur penting yang tidak memiliki parameter dan berfungsi untuk menambahkan output dari lapisan sebelumnya langsung ke lapisan berikutnya (Shorten, 2019). Dengan menggunakan skip connection, jaringan dapat meneruskan informasi dari lapisan sebelumnya tanpa perubahan, memungkinkan jaringan untuk mempelajari fungsi identitas. Ini membantu dalam pelatihan jaringan yang lebih dalam dengan meningkatkan kemampuan untuk memanfaatkan informasi yang sudah ada dari lapisan sebelumnya (Banjara, 2023).

ResNet telah mengalami banyak pengembangan dan pembaruan, dimulai dengan ResNet-18 yang memiliki 18 lapisan, varian ini terus berkembang hingga ResNet-152 dengan 152 lapisan. Pengembangan juga menghasilkan varian lain seperti ResNeXt, *Wide* ResNet, dan ResNet *with Attention Mechanism* (Chaure, 2024). Setiap varian menawarkan peningkatan kapasitas untuk menangani data yang lebih kompleks sambil menjaga efisiensi pelatihan, menjadikan ResNet sebagai fondasi penting dalam berbagai aplikasi *deep learning*.

**3.12. Metrik Evaluasi**

Metrik evaluasi (*Evaluation Metrics*) adalah ukuran atau indikator yang digunakan untuk menilai kinerja dan efektivitas model pembelajaran mesin atau algoritma. Metrik ini memberikan gambaran seberapa baik model tersebut bekerja dalam memprediksi atau mengklasifikasikan data, dan digunakan untuk membandingkan hasil antar model atau memilih model terbaik. Dalam tugas klasifikasi data, metrik evaluasi digunakan dalam dua tahap, yaitu tahap pelatihan dan pengujian (Hossin & Sulaiman, 2015). Contoh metrik evaluasi meliputi akurasi (*accuracy*), *presisi*, *recall*, dan *f1-score*. Metrik evaluasi tersebut bisa dihitung menggunakan nilai-nilai dari *confusion matrix*.



Gambar 3.13 Confusion Matrix (Karra, 2020)

*True positive* (TP) terjadi ketika data yang sebenarnya bernilai positif diprediksi dengan benar sebagai positif. *True negative* (TN) adalah kondisi di mana model memprediksi data yang bernilai negatif dengan tepat sebagai negatif. *False positive* (FP) terjadi ketika data yang sebenarnya negatif diprediksi sebagai positif. *False negative* (FN) adalah situasi di mana data yang seharusnya positif diprediksi sebagai negatif. Dengan menggunakan kombinasi dari 4 nilai tersebut, dapat dihitung metrik evaluasi di bawah ini.

**3.12.1. Akurasi**

Secara umum, akurasi (*accuracy*) merupakan perbandingan antara prediksi benar terhadap jumlah total contoh yang dievaluasi. Nilai akurasi dapat dihitung dengan menggunakan persamaan berikut (Hossin & Sulaiman, 2015):

(3.4)

**3.12.2. *Precision***

*Precision* digunakan untuk mengukur pola positif yang diprediksi dengan benar dari total pola yang diprediksi dalam kelas positif. Secara matematis, *precision* dapat ditulis dalam bentuk persamaan berikut (Hossin & Sulaiman, 2015):

(3.5)

**3.12.3. *Recall***

*Recall* merupakan perbandingan antara jumlah prediksi positif yang benar dengan jumlah keseluruhan yang diprediksi benar. *Recall* dapat diungkapkan secara matematis melalui persamaan berikut (Hossin & Sulaiman, 2015):

(3.6)

**3.12.4. *F1-score***

*F1-score* digunakan untuk menghitung rata-rata harmonik antara nilai *precision* dan *recall*. Dalam bentuk matematis, *f1-score* dapat dituliskan dengan persamaan berikut (Hossin & Sulaiman, 2015):

(3.7)

**3.14. Python**

Python adalah bahasa pemrograman tingkat tinggi yang mulai dikembangkan pada tahun 1989 oleh *programmer* asal belanda, Guido van Rossum (Moltzau, 2019). Bahasa pemrograman tingkat tinggi adalah bahasa yang dirancang untuk lebih mudah dipahami dan digunakan oleh manusia, dengan sintaksis dan struktur yang lebih mirip dengan bahasa alami atau logika matematis. Bahasa ini menyediakan tingkat abstraksi yang tinggi dari perangkat keras, memungkinkan programmer untuk fokus pada logika dan fungsionalitas tanpa harus mengelola detail teknis rendah seperti manajemen memori secara langsung.



Gambar 3.14 Python (Ramdhani, 2019)

Python adalah bahasa pemrograman *open-source* yang populer dan serbaguna, digunakan dalam berbagai aplikasi mulai dari pengembangan web, analisis data, kecerdasan buatan, dan otomatisasi skrip . Python juga didukung oleh ribuan pustaka (*library*) yang kuat, yang memungkinkan pengembang untuk dengan mudah mengimplementasikan solusi di berbagai bidang, mulai dari ilmu data dengan pandas dan NumPy, hingga pengembangan web dengan Django dan Flask, serta machine learning dengan TensorFlow dan Keras.

TensorFlow adalah sebuah pustaka *open-source* yang digunakan dalam pembelajaran mesin dan kecerdasan buatan yang dikembangkan oleh Google. TensorFlow dapat digunakan untuk banyak tugas, termasuk pelatihan dan *deployment* model pembelajaran mesin (Kılınç, 2023). Salah satu keunggulan TensorFlow adalah kemampuannya berjalan pada berbagai *platform* seperti *desktop*, perangkat *mobile*, dan lingkungan komputasi *cloud*. TensorFlow juga didukung oleh komunitas besar pengguna dan pengembang, yang mempermudah pencarian bantuan dan akses ke berbagai sumber daya secara daring.

Keras adalah API untuk mengembangkan jaringan saraf tiruan. Aplikasi dari Keras sangat luas di mana kita dapat membangun jaringan saraf tiruan untuk klasifikasi gambar, pemrosesan bahasa alami, pengenalan suara, dan prediksi *time series* (Wahidi, 2021). TensorFlow dan Keras bekerja sama untuk memudahkan pengembangan model machine learning. Keras, sebagai antarmuka tingkat tinggi, menyediakan API yang sederhana dan intuitif untuk membangun dan melatih model neural network, sementara TensorFlow berfungsi sebagai backend yang kuat untuk komputasi dan eksekusi model. Kolaborasi ini memungkinkan pengguna untuk memanfaatkan kemudahan penggunaan Keras dengan kekuatan dan fleksibilitas TensorFlow, menggabungkan kemudahan pengembangan dengan performa tinggi dalam aplikasi machine learning.