

BAB II

LANDASAN TEORI

2.1 Penelitian Terkait

Berikut adalah penelitian-penelitian terdahulu yang digunakan sebagai referensi dalam mendukung topik yang diteliti. Penelitian-penelitian ini memberikan landasan teoritis dan empiris terkait pendekatan, metode, serta teknologi yang relevan dalam pengembangan sistem klasifikasi penyakit paru-paru pada citra X-ray menggunakan teknik *deep learning*. Dengan mengkaji berbagai karya ilmiah sebelumnya, peneliti dapat mengidentifikasi keunggulan, keterbatasan, serta peluang pengembangan lebih lanjut yang menjadi dasar dalam merumuskan metodologi dan arah penelitian ini.

Tabel 2.1 Tabel Penelitian Terkait

Jurnal 1	
Judul	Augmenting Radiological Diagnostics with AI for Tuberculosis and COVID-19 Disease Detection: Deep Learning Detection of Chest Radiographs [12].
Peneliti	Manjur Kolhar, Ahmed M. Al Rajeh, dan Raisa Nazir Ahmed Kazi.
Jurnal	<i>Journal of Diagnostics</i>
Tahun	2024
DOI	10.3390/diagnostics14131334
Metode	VGG16, ResNet50.
Pembahasan dan Hasil	Hasil penelitian menunjukkan bahwa model ResNet50 secara signifikan lebih unggul dibandingkan VGG16 dalam mendeteksi penyakit paru-paru seperti COVID-19 dan tuberkulosis. ResNet50 mencapai tingkat akurasi validasi mencapai 97,5%. Sementara itu, VGG16 juga menunjukkan performa yang baik selama pengujian, dengan <i>precision</i> 0,99 dan <i>recall</i> 0,93 untuk deteksi tuberkulosis.
Jurnal 2	
Judul	Multiclass Lung Disease Classification with Ensembled CNN and

	LSTM Modeling [13].
Peneliti	Shagun Sharma, Kalpna Guleria.
Jurnal	<i>AIP Conf. Proc</i>
Tahun	2024
DOI	10.1063/5.0229036
Metode	Ensemble CNN-LSTM
Pembahasan dan Hasil	Hasil penelitian menunjukkan bahwa model ensemble CNN-LSTM berhasil meningkatkan klasifikasi penyakit paru-paru dengan akurasi tinggi sebesar 96,2%. Model ini mampu menggabungkan fitur spasial (geografis) dan temporal dari citra untuk mendeteksi berbagai penyakit paru, termasuk COVID-19 dan tuberkulosis, secara cepat dan akurat.
Jurnal 3	
Judul	An explainable artificial intelligence model for multiple lung diseases classification from chest X-ray images using fine-tuned transfer learning [14].
Peneliti	Eram Mahamud, Nafiz Fahad, Md Assaduzzaman, S.M. Zain, Kah Ong Michael Goh, dan Md. Kishor Morol.
Jurnal	<i>Decision Analytics Journal</i>
Tahun	2024
DOI	10.1016/j.dajour.2024.100499
Metode	InceptionV3, EfficientNetB0, LeNet, Proposed Model (<i>Customized CNN</i>).
Pembahasan dan Hasil	Hasil penelitian menunjukkan bahwa <i>proposed model</i> mencapai akurasi tertinggi sebesar 99.20%, dengan performa superior dalam hal <i>precision</i> , <i>recall</i> , dan <i>F1-score</i> , serta menunjukkan kemampuan generalisasi yang baik, serta interpretabilitas yang ditingkatkan menggunakan teknik <i>Explainable AI</i> .
Jurnal 4	
Judul	Explainable Lung Disease Classification from Chest X-Ray Images Utilizing Deep Learning and XAI [15].
Peneliti	Tanzina Taher Ifty, Saleh Ahmed Shafin, Shoeb Mohammad Shahriar, dan Tashfia Towhid.
Jurnal	<i>Journal of Electrical Engineering and System Science</i>

Tahun	2024
DOI	10.48550/arXiv.2404.11428
Metode	Xception, InceptionV3, DenseNet121, DenseNet201, ResNet50, VGG19, EfficientNetB7.
Pembahasan dan Hasil	Hasil penelitian menunjukkan bahwa model Xception mencapai akurasi tertinggi hingga 96.2% dengan <i>cross-validation</i> , dan teknik <i>XAI</i> seperti <i>Grad-CAM</i> dan <i>LIME</i> berhasil memberikan penjelasan visual yang meningkatkan interpretabilitas prediksi.
Jurnal 5	
Judul	MixNet-LD: An Automated Classification System for Multiple Lung Diseases Using Modified MixNet Model [16].
Peneliti	Ayesha Ahoor, Fahim Arif, Muhammad Zaheer Sajid, Imran Qureshi, Fakhar Abbas, Sohail Jabbar, Qaisar Abbas.
Jurnal	<i>Journal of Diagnostics</i>
Tahun	2023
DOI	10.3390/diagnostics132031
Metode	MixNet CNN Dense Blocks (dalam arsitektur CNN), SVM.
Pembahasan dan Hasil	Hasil penelitian menunjukkan bahwa sistem <i>MixNet-LD</i> mampu mencapai akurasi tinggi sebesar 98,5% dalam mengklasifikasi berbagai penyakit paru-paru, termasuk COVID-19, pneumonia, tuberkulosis, dan kanker paru-paru, dengan reliabilitas yang konsisten pada dataset yang berbeda.
Jurnal 6	
Judul	Image Classification of Various Lung Diseases Using Transfer Learning and Machine Learning Techniques [17].
Peneliti	Neelam Sharma, Neeraj Garg, Bhavay Garg, Shivesh Ranjan, dan Piyush Goyal.
Jurnal	<i>SSRN Electronic Journal</i>
Tahun	2023
DOI	10.2139/ssrn.4628662
Metode	ResNet18, DenseNet121, VGG15, Inception-ResNet V2, Xception EfficientNet, MobileNet.

Pembahasan dan Hasil	Hasilnya menunjukkan bahwa model ResNet18, DenseNet121, dan EfficientNet mencapai akurasi tertinggi sekitar 90%, dengan performa yang baik dalam mengklasifikasikan berbagai penyakit paru-paru dari citra CT.
Jurnal 7	
Judul	DeepLungNet: An effective DL-based approach for lung disease classification using CRIs [18].
Peneliti	Naeem Ullah, Mehrez Marzougui, Ijaz Ahmad, Samia Allaoua Chelloug.
Jurnal	<i>Journal of Electronics</i>
Tahun	2023
DOI	10.3390/electronics12081860
Metode	<i>DeepLungNet</i> (CNN Custom proposed model), SVM, VGG16, ResNet50, InceptionV3, MobileNetV2, EfficientNetB4, Xception.
Pembahasan dan Hasil	Hasil penelitian menunjukkan bahwa model <i>DeepLungNet</i> berhasil mengklasifikasikan penyakit paru-paru dari citra X-ray dengan akurasi hingga 97,47% pada klasifikasi 5 kelas (COVID-19, TB, <i>Lung Opacity</i> , pneumonia, normal), dan akurasi 95,57% pada klasifikasi 6 kelas (<i>Viral Pneumonia</i> , <i>Bacterial Pneumonia</i> , COVID-19, TB, <i>Lung Opacity</i> , normal).
Jurnal 8	
Judul	Hybrid CLAHE-CNN Deep Neural Networks for Classifying Lung Diseases from X-ray Acquisitions [19].
Peneliti	Fairouz Hussein, Ala Mughaid, Shadi AlZu'bi, Subhieh M. El-Salhi, Belal Abuhaija, Laith Abualigah, dan Amir H. Gandomi.
Jurnal	<i>Journal of Electronics</i>
Tahun	2022
DOI	10.3390/electronics11193075
Metode	SVM, VGG19, CNN.
Pembahasan dan Hasil	Eksperimen menunjukkan bahwa kombinasi CLAHE + CNN menghasilkan akurasi terbaik, yaitu hingga 91%, dibandingkan dengan model lainnya. Selain itu, CNN menunjukkan performa yang lebih tinggi dibandingkan SVM dan VGG19.
Jurnal 9	

Judul	The Use of CLAHE for Improving an Accuracy of CNN Architecture for Detecting Pneumonia [20].
Peneliti	Elbert Alfredo Tjoa, I Putu Yowan Nugraha Suparta, Rita Magdalena, dan Nor Kumalasari CP
Jurnal	<i>SHS Web of Conferences</i>
Tahun	2022
DOI	10.1051/shsconf/202213903026
Metode	CNN AlexNet, YENS-Net (Proposed Customized CNN)
Pembahasan dan Hasil	Penelitian ini membandingkan tiga pendekatan: CNN AlexNet, CLAHE + AlexNet, dan YENS-Net. Hasil terbaik diperoleh oleh CNN AlexNet dengan akurasi sebesar 91.11%. CLAHE digunakan sebagai metode pra-pemrosesan untuk meningkatkan kualitas citra, namun tidak menunjukkan peningkatan signifikan dibandingkan CNN AlexNet tanpa CLAHE.
Jurnal 10	
Judul	Automated Pneumonia Diagnosis Using a 2D Deep Convolutional Neural Network with Chest X-Ray Images [21].
Peneliti	Kamila Kassylkassova, Batyrkhan Omarov, Gulnur Kazbekova, Zhadra Kozhamkulova, Mukhit Maikotov, dan Zhanar Bidakhmet
Jurnal	International Journal of Advanced Computer Science and Applications (IJACSA)
Tahun	2023
DOI	10.14569/IJACSA.2023.0140281
Metode	2D Deep CNN
Pembahasan dan Hasil	Penelitian ini mengusulkan model CNN 2D untuk diagnosis otomatis pneumonia dari citra X-ray paru-paru. Dataset yang digunakan terdiri dari 5.856 citra. Model CNN terdiri dari empat lapisan konvolusional dan dua lapisan <i>dense</i> . Hasil pengujian menunjukkan akurasi sebesar 96% dan <i>error rate</i> sebesar 4%, yang menunjukkan efektivitas model dalam klasifikasi dua kelas (pneumonia dan normal).

Berdasarkan Tabel 2.1, berbagai pendekatan telah dikembangkan untuk mendeteksi dan mengklasifikasikan penyakit paru-paru dari citra X-ray, dengan tren yang semakin mengarah pada pemanfaatan model *deep learning* dan teknik interpretabilitas. Kolhar *et al.* (2024) dalam *Journal of*

Diagnostics menunjukkan bahwa arsitektur ResNet50 memberikan performa lebih unggul dibandingkan VGG16 dalam mendeteksi COVID-19 dan tuberkulosis, dengan akurasi validasi mencapai 97,5% [12]. Penelitian lain oleh Sharma dan Guleria (2024) mengusulkan model ensemble berbasis CNN dan LSTM yang mampu menggabungkan fitur spasial dan temporal dari citra X-ray, menghasilkan akurasi sebesar 96,2% [13].

Mahamud *et al.* (2024) dalam *Decision Analytics Journal* menyempurnakan pendekatan CNN dengan memanfaatkan *transfer learning* dan teknik *Explainable AI* (XAI), sehingga model yang dikembangkan mampu mencapai akurasi tertinggi sebesar 99,20% [14]. Senada dengan itu, Ifty *et al.* (2024) juga menyoroti pentingnya interpretabilitas model melalui penggunaan *Grad-CAM* dan *LIME*, serta menunjukkan bahwa model Xception yang digunakan berhasil mencapai akurasi 96,2% [15]. Pendekatan lain ditunjukkan oleh Ayesha Ahoor *et al.* (2023) yang mengembangkan sistem MixNet-LD berbasis *Dense Blocks* dan *Grad-CAM*, dengan fokus pada klasifikasi multikelas termasuk kanker paru-paru, dan memperoleh akurasi tinggi sebesar 98,5% [16].

. Sementara itu, Sharma *et al.* (2024) melakukan komparasi antara model *deep learning* dan *machine learning*, dan menemukan bahwa EfficientNet, ResNet18, serta DenseNet121 termasuk arsitektur dengan performa terbaik, dengan akurasi berkisar 90% [17]. Dalam *Journal of Electronics*, Naeem Ullah *et al.* (2023) memperkenalkan model *DeepLungNet* yang berhasil mengklasifikasikan hingga enam jenis penyakit paru-paru dengan akurasi sebesar 97,47%, sekaligus menunjukkan kemampuan generalisasi yang kuat [18].

Sejalan dengan hasil-hasil penelitian terdahulu, pemanfaatan arsitektur *deep learning* seperti EfficientNet, ResNet, dan DenseNet, serta teknik interpretabilitas seperti Grad-CAM, telah berkontribusi signifikan dalam meningkatkan performa dan transparansi model klasifikasi penyakit

paru-paru. Pendekatan *transfer learning* juga terbukti efektif, khususnya pada kasus dengan keterbatasan data, yang menjadi salah satu landasan penting dalam penelitian ini yang menggunakan EfficientNetB3, ResNet50, dan DenseNet201 sebagai model komparatif utama.

Meskipun berbagai penelitian telah menunjukkan keberhasilan model *deep learning* dalam klasifikasi penyakit paru-paru, masih terdapat ruang untuk peningkatan dari segi kualitas citra, cakupan dataset serta peningkatan akurasi deteksi. Oleh karena itu, penelitian ini bertujuan untuk meningkatkan akurasi klasifikasi penyakit paru-paru dengan menerapkan CLAHE sebagai teknik pra-pemrosesan citra, serta menambah variasi dataset guna memperkuat kemampuan generalisasi model.

2.2 Teori Penelitian

2.2.1 Penyakit Paru-paru

Paru-paru merupakan organ vital dalam sistem pernapasan yang berperan utama dalam proses pertukaran gas, yaitu oksigen dan karbondioksida. Organ ini sangat rentan terhadap berbagai gangguan akibat infeksi mikroorganisme, paparan polutan, dan kebiasaan merokok. Penyakit paru-paru menjadi salah satu penyebab utama masalah kesehatan global, terutama di wilayah dengan kualitas udara yang buruk dan akses layanan kesehatan yang terbatas. Beberapa penyakit paru-paru yang umum dijumpai antara lain tuberkulosis (TB), pneumonia, serta COVID-19.

Penyakit-penyakit ini dapat menimbulkan gejala serius seperti batuk berkepanjangan, sesak napas, nyeri dada, dan kelelahan, serta berpotensi menyebabkan komplikasi yang membahayakan jiwa jika tidak segera ditangani. Selain itu, infeksi saluran pernapasan juga tercatat sebagai penyebab kematian utama pada bayi dan kelompok

rentan lainnya, menjadikan gangguan paru-paru sebagai masalah kesehatan yang signifikan di berbagai lapisan usia.

Beragam penyakit dapat menyerang paru-paru, masing-masing dengan penyebab, gejala, dan tingkat keparahan yang berbeda. Pemahaman yang mendalam terhadap karakteristik dari setiap penyakit paru menjadi penting untuk mendukung diagnosis dan penanganan yang tepat. Berikut ini akan dibahas beberapa jenis penyakit paru-paru yang paling umum dan memiliki dampak besar terhadap kesehatan masyarakat, yaitu tuberkulosis, pneumonia, dan COVID-19.

2.2.1.1 Tuberkulosis

Penyakit tuberkulosis, yang dikenal pula dengan sebutan TB, disebabkan oleh bakteri *Mycobacterium tuberculosis*. Meskipun paru-paru menjadi organ yang paling sering terinfeksi, penyakit ini juga dapat menyerang bagian tubuh lain seperti ginjal, tulang, dan otak. Penyakit ini menular dan dapat menyebar melalui batuk atau polutan yang terkandung dalam udara [22]. Gejala tuberkulosis paru-paru biasanya termasuk batuk yang berkepanjangan, kadang-kadang disertai dengan dahak berdarah, demam, penurunan berat badan, kelelahan, dan keringat pada malam hari [23]. Tuberkulosis dapat menjadi penyakit serius yang membutuhkan pengobatan antibiotik yang lama. Untuk mencegah penyebaran infeksi dan mencegah komplikasi yang lebih serius, TB harus diidentifikasi dan diobati segera mungkin. Penggunaan kombinasi antibiotik selama periode waktu yang cukup panjang biasanya merupakan bagian dari program pengobatan TB [24].

2.2.1.2 Pneumonia

Pneumonia adalah peradangan paru-paru yang sering kali disebabkan oleh infeksi bakteri, virus, jamur, atau bahan kimia yang terkandung dalam polutan udara. Infeksi ini dapat mengisi

alveoli, kantong udara di paru-paru, dengan cairan atau nanah, yang membuat sulit bernapas. Pneumonia dapat menyerang satu atau kedua paru-paru [25]. Pneumonia memiliki gejala seperti batuk yang disertai dengan dahak berwarna, demam, menggigil, nyeri dada yang dapat menjadi lebih parah saat bernapas atau batuk, napas cepat, dan kelelahan. Usia, kesehatan umum, dan penyebab infeksi adalah beberapa faktor yang dapat memengaruhi gejala [26]. Walaupun pneumonia dapat menyerang semua kelompok usia, anak-anak, lansia, dan orang dengan sistem kekebalan yang lemah memiliki risiko lebih tinggi. Pemeriksaan fisik, analisis darah, dan seringkali pemeriksaan citra seperti sinar-X paru-paru adalah bagian dari diagnosis pneumonia. Orang tua, anak-anak, dan individu dengan sistem kekebalan yang lemah memiliki risiko lebih besar terkena pneumonia dibandingkan dengan kelompok usia lainnya [26]. Diagnosis pneumonia memerlukan pemeriksaan fisik, analisis darah, dan seringkali pemeriksaan citra, seperti sinar-X bagian dada [27].

2.2.1.3 COVID-19

COVID-19 adalah penyakit infeksi saluran pernapasan yang disebabkan oleh virus *SARS-CoV-2*. Virus ini menyerang saluran pernapasan atas dan bawah, termasuk saluran *bronkial* dan *alveoli* di paru-paru, yang dapat menyebabkan peradangan dan gangguan pertukaran oksigen [28]. Gejala umum COVID-19 meliputi batuk kering, demam, sesak napas, kelelahan, dan nyeri dada. Pada beberapa kasus, infeksi dapat berkembang menjadi pneumonia atau bahkan *Acute Respiratory Distress Syndrome* (ARDS), yang merupakan kondisi serius dan membutuhkan perawatan intensif [28].

Selain gejala akut, sebagian pasien mengalami gejala jangka panjang yang dikenal sebagai *long COVID*, termasuk batuk berkepanjangan, sesak napas, dan kelelahan yang berlangsung selama berminggu-minggu atau berbulan-bulan setelah infeksi awal [29]. COVID-19 juga dapat menyebabkan komplikasi serius pada sistem pernapasan, terutama pada individu dengan kondisi kesehatan mendasar seperti penyakit jantung, paru-paru, atau gangguan imun [30].

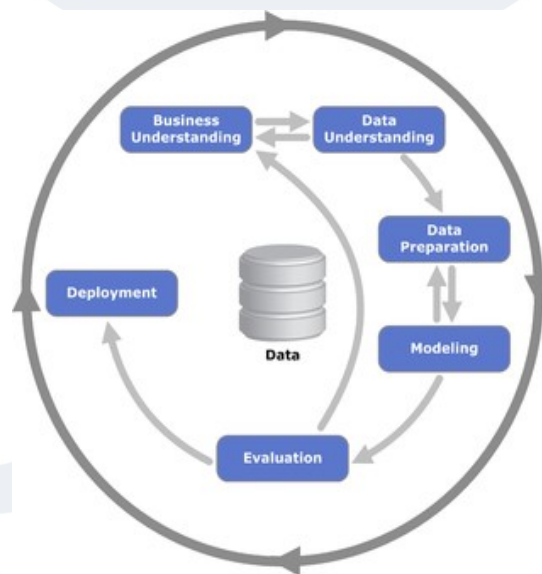
2.3 Framework dan Algoritma

Dalam proses pengembangan sistem klasifikasi berbasis kecerdasan buatan, pemilihan *framework* dan algoritma yang tepat menjadi langkah fundamental yang menentukan keberhasilan analisis dan akurasi model. *Framework* digunakan untuk memberikan panduan sistematis dalam tahapan pengolahan data, mulai dari pemahaman masalah hingga implementasi model, sedangkan algoritma berperan dalam proses pembelajaran dan prediksi berdasarkan data yang tersedia. Salah satu *framework* yang banyak digunakan dalam proyek-proyek *data mining* dan analisis prediktif adalah CRISP-DM (*Cross-Industry Standard Process for Data Mining*). *Framework* ini menyediakan pendekatan terstruktur yang membantu mengelola dan mengeksekusi setiap tahapan secara efisien. Berikut adalah penjelasan mengenai CRISP-DM dan relevansinya dalam penelitian ini.

2.3.1 Framework CRISP-DM

Model *Cross-Industry Standard Process for Data Mining* (CRISP-DM) merupakan suatu kerangka kerja untuk proses *data mining* yang menguraikan siklus hidup dalam kegiatan tersebut. Gambar 2.1 merupakan tahapan CRISP-DM yang terdiri dari enam fase utama, yaitu *Business Understanding*, *Data Understanding*, *Data*

Preparation, Modeling, Evaluation, dan Deployment, yang membentuk alur kerja iteratif dan fleksibel. CRISP-DM diakui sebagai standar utama dalam eksplorasi *data mining* dan telah membuktikan kemampuannya dalam memberikan efisiensi dalam hal biaya dan waktu [31]. Penggunaannya yang luas dalam berbagai domain membuktikan bahwa *framework* ini dapat diadaptasi dengan baik pada beragam konteks penelitian, termasuk dalam bidang medis. Pada penelitian terdahulu mengenai analisis bedah kanker paru-paru yang menggunakan *framework* CRISP-DM, untuk mengeksplorasi data dari 501 pasien yang menjalani operasi kanker paru-paru dengan tujuan penyembuhan. Model terpilih menghasilkan performa yang memuaskan, dengan nilai sensitivitas, spesifisitas, presisi global, prediktif positif, dan prediktif negatif yang cukup signifikan, sehingga relevan dengan penelitian ini yang akan menggunakan *framework* CRISP-DM [32].



Gambar 2.1 Framework CRISP-DM [31]

CRISP-DM memiliki enam fase iteratif sebagai berikut:

1. Business Understanding

Pada tahap *Business Understanding*, fokus diberikan pada identifikasi faktor-faktor krusial terkait dengan penelitian bisnis jangka panjang. Analisis situasi bisnis dilakukan untuk mendapatkan gambaran tentang ketersediaan sumber daya, sementara tujuan dari data mining ditentukan dalam fase ini. Langkah ini menjadi dasar untuk merancang pendekatan penelitian yang sesuai dengan konteks bisnis yang sedang dihadapi [32].

2. Data Understanding

Data Understanding merupakan tahap dalam siklus analisis data yang berfokus pada pemahaman mendalam terhadap data. Data dieksplorasi, dideskripsikan, dan disiapkan untuk memastikan kualitas dan keutuhan. Analisis statistik digunakan untuk mendeskripsikan informasi dari data, sambil menentukan atribut yang relevan untuk pengolahan lebih lanjut [32].

3. Data Preparation

Data preparation merupakan langkah persiapan dan penataan data sebelum memasuki tahap analisis. Proses ini melibatkan pembersihan, transformasi, dan pengorganisasian data agar siap digunakan dalam pemodelan atau analisis lebih lanjut. Fokusnya adalah memastikan data memiliki keakuratan, konsistensi, dan kesiapan yang diperlukan untuk mendapatkan hasil analisis yang optimal [32].

4. Modelling

Modelling adalah tahap memilih teknik pemodelan yang sesuai dengan karakteristik masalah dan data. Selanjutnya, model dikembangkan menggunakan teknik tersebut dengan memanfaatkan data yang telah disiapkan sebelumnya, dan proses *tuning* dilakukan untuk meningkatkan kinerja model [32].

5. Evaluation

Evaluation merupakan langkah yang bertujuan untuk mengevaluasi kinerja model atau hasil analisis secara mendalam. Di dalam proses ini dilakukan penggunaan metrik evaluasi yang sesuai untuk mengukur keberhasilan model atau solusi yang dikembangkan, seperti *accuracy*, *precision*, *recall*, atau metrik lainnya. Evaluasi membantu memastikan bahwa model atau solusi dapat memenuhi tujuan bisnis yang ditetapkan dan memberikan nilai tambah yang diinginkan. Jika hasil evaluasi tidak memuaskan, dapat diperlukan langkah-langkah tambahan seperti penyesuaian model atau pemilihan ulang fitur untuk meningkatkan kualitas hasil [32].

6. Deployment

Deployment adalah proses mengimplementasikan model atau solusi yang dikembangkan agar dapat beroperasi secara aktif di lingkungan produksi. Proses ini mencakup penerapan model ke dalam sistem yang dapat digunakan oleh pengguna akhir atau organisasi. Selain itu, aspek-aspek seperti integrasi dengan sistem yang ada, pengelolaan sumber daya, dan pelatihan pengguna mungkin perlu diperhatikan untuk memastikan penerapan yang sukses. *Deployment* memungkinkan hasil dari analisis atau pemodelan menjadi operasional dan memberikan dampak pada keputusan bisnis atau proses yang sesuai dengan tujuan awal proyek [32].

2.3.2 Data Augmentation

Data augmentation adalah suatu teknik dalam *deep learning* yang bertujuan untuk meningkatkan jumlah dan keragaman data pelatihan. Ini dilakukan dengan menghasilkan variasi dari data yang sudah ada, sehingga model dapat melihat berbagai situasi dan pola yang mungkin

terjadi dalam dunia nyata [33]. Dalam pengolahan citra, *data augmentation* sering melibatkan transformasi seperti rotasi, pergeseran, perubahan skala, *flipping*, dan perubahan warna pada citra asli. Misalnya, dalam *data augmentation* untuk citra, sebuah citra dapat dirotasi beberapa derajat, diperkecil atau diperbesar, di-*flip* secara horizontal atau vertikal, atau diberikan variasi dalam warna. Proses ini menciptakan versi baru dari citra tersebut, tetapi tetap mempertahankan label kelas yang sama. Dengan cara ini, model dapat melihat variasi yang lebih luas dari objek atau pola yang ada dalam dataset pelatihan [34].

Data augmentation diterapkan untuk mencegah *overfitting*, yang terjadi ketika model hanya cocok dengan data pelatihan dan gagal bekerja dengan baik pada data yang belum pernah dilihat. Dengan menambahkan variasi pada data pelatihan, model menjadi lebih mampu belajar secara efektif dan optimal, meningkatkan kinerja pada data pengujian yang belum pernah dilihat sebelumnya. *Data augmentation* merupakan praktik umum dalam pelatihan model, terutama ketika dataset pelatihan terbatas [35].

Proses *data augmentation* dalam penelitian ini dilakukan dengan memanfaatkan modul *Keras* pada tahap pelatihan model. Augmentasi ini diterapkan secara *online* (selama proses *training*) agar model lebih *robust* terhadap variasi *input*. Tabel 2.2 merupakan rangkaian transformasi augmentasi yang digunakan meliputi:

Tabel 2.2 Layer Data Augmentasi

Layer	Deskripsi	Fungsi
<i>InputLayer</i> (224, 224, 3) [36]	Menentukan dimensi <i>input</i> citra sebagai masukan <i>RGB</i> dengan ukuran 224×224 piksel	Menyesuaikan <i>input</i> dengan standar arsitektur CNN.

Layer	Deskripsi	Fungsi
<i>RandomFlip</i> ("horizontal") [36]	Melakukan pembalikan citra secara horizontal secara acak	Membuat model lebih tahan terhadap orientasi objek yang berbeda.
<i>RandomTranslation</i> (0.2, 0.2) [36]	Menerapkan translasi acak hingga 20% secara horizontal dan vertikal	Membantu model mengenali objek yang posisinya bergeser dari tengah
<i>RandomZoom</i> (0.2) [36]	Melakukan pembesaran atau pengecilan citra secara acak hingga 20%	Memberikan ketahanan model terhadap variasi skala objek

Semua transformasi ini dilakukan secara *real-time* selama proses pelatihan, sehingga setiap *epoch* model akan melihat variasi baru dari data yang sama, yang pada akhirnya membantu mengurangi *overfitting* dan meningkatkan kinerja generalisasi model pada data uji yang belum pernah dilihat sebelumnya.

2.3.3 Deep Learning

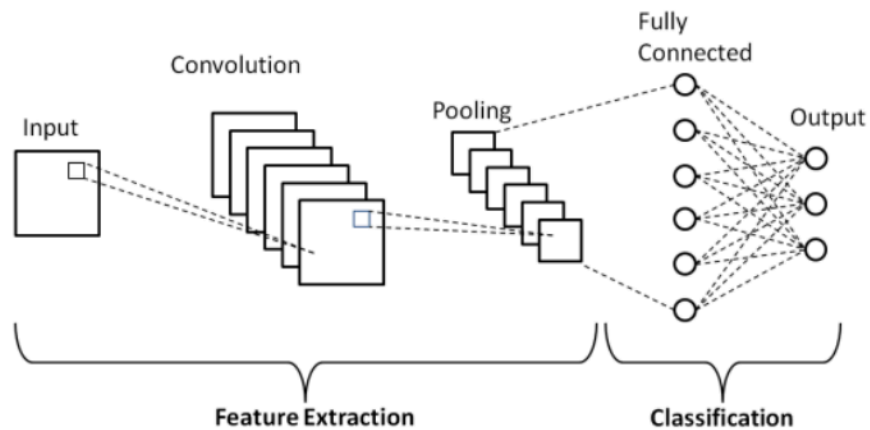
Deep learning adalah cabang dari *Artificial Intelligence* (AI) yang fokus pada pengembangan dan penggunaan algoritma jaringan saraf tiruan yang disebut jaringan saraf dalam (*neural networks*) untuk meniru cara manusia belajar [37]. Model *deep learning* terdiri dari lapisan-lapisan yang dapat memahami dan memproses data secara hierarkis, memungkinkan untuk mempelajari representasi fitur yang kompleks. Fokus utama dari *deep learning* adalah penggunaan jaringan saraf tiruan, yang terinspirasi dari struktur jaringan saraf manusia, untuk mengidentifikasi pola yang kompleks dan mewakili informasi secara hierarkis. *Deep learning* telah menunjukkan hasil signifikan dalam

berbagai bidang, termasuk pengenalan citra dan suara, pemrosesan bahasa alami, serta identifikasi pola-pola kompleks [38].

2.3.4 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan algoritma populer yang sering dipakai untuk mengolah data berbentuk citra [39]. Algoritma ini menggunakan operasi matematis untuk belajar dari suatu jaringan yang disebut "*Convolution Layer*". *Convolution layer* merupakan komponen penting dari CNN, yang memainkan peran vital dalam ekstraksi fitur dengan memanfaatkan koneksi lokal dan bobot karakteristik pada objek. *Convolution layer* terdiri dari operasi linear dan non-linear yang bekerja bersama-sama untuk mengekstrak pola-pola bermakna dari *input* data [40].

Kemampuan *convolution layer* untuk mengekstraksi fitur-fitur penting menjadi alasan utama mengapa CNN sangat sesuai untuk data citra. Beberapa filter dalam *convolution layer* mempelajari fitur-fitur yang berbeda melalui berbagai bobot. Secara matematis, filter ini dapat direpresentasikan sebagai matriks berukuran $n \times n$, dengan setiap elemen merepresentasikan suatu bobot tertentu. Proses penyingkapan diilustrasikan dalam Gambar 2.2. Setiap pasang elemen dalam filter dikalikan dan dijumlahkan untuk menghasilkan satu nilai *output*. Proses ini diulang untuk setiap filter, menghasilkan "*feature map*" yang menyoroti fitur-fitur khusus dalam data masukan. Dengan menggunakan beberapa filter dengan bobot yang berbeda, *convolution layer* secara efektif mempelajari set fitur yang beragam, memungkinkannya mengidentifikasi dan membedakan berbagai objek atau pola dalam *input* data.

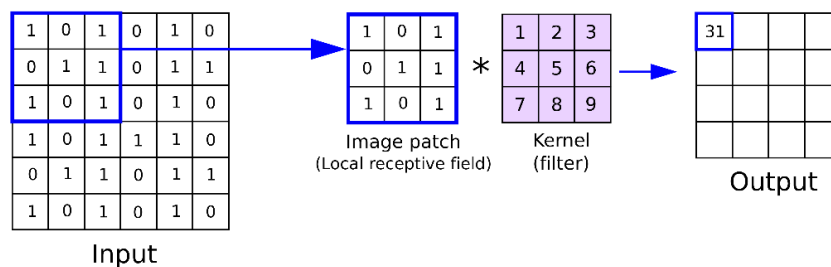


Gambar 2.2 Arsitektur CNN [41]

Berdasarkan penelitian terdahulu, CNN terbukti sangat cocok digunakan dalam bidang radiologi karena keunggulannya dalam analisis citra medis. CNN memanfaatkan *layer* konvolusi yang memungkinkan ekstraksi otomatis fitur-fitur kritis pada citra medis, seperti lesi atau nodul pada citra *Computed Tomography* (CT) dan *Magnetic Resonance Imaging* (MRI) [42]. Kelebihan utamanya terletak pada kemampuannya untuk menangani data 2D dan 3D, memungkinkan adaptasi yang *seamless* pada berbagai jenis pemeriksaan radiologi. Selain itu, CNN dapat dengan sukses membaca data deret waktu, seperti pada pemeriksaan dinamis dengan kontras dinamis pada CT/MRI. Fleksibilitas CNN juga tercermin dalam kemampuannya mengklasifikasikan berbagai jenis lesi, seperti nodul paru-paru atau massa hati. Dalam beberapa kasus, penggunaan model CNN 3D memungkinkan pemahaman konteks spasial 3D yang diperlukan untuk diagnosis yang lebih akurat. Dengan kombinasi fitur-fitur ini, CNN menjadi alat yang sangat efektif dalam mendukung deteksi dini dan diagnosis penyakit melalui analisis citra medis di bidang radiologi [42]. Dari sisi struktur *layer* yang terdapat pada CNN, secara umum CNN terdiri dari 3 jenis *layer*, yaitu:

1. Convolutional Layer

Convolutional layer adalah komponen kunci dalam CNN pada *deep learning*. Lapisan ini dirancang khusus untuk mengekstraksi fitur dari data *input*, terutama data spasial seperti citra. *Convolutional layer* menggunakan operasi konvolusi untuk menerapkan sejumlah filter atau *kernel* pada *input*, yang membantu mendeteksi pola atau fitur khusus dalam data. Dengan cara ini, *convolutional layer* dapat memahami hierarki fitur yang semakin kompleks seiring dengan kedalaman lapisan dalam arsitektur CNN. *Convolutional layer* memainkan peran penting dalam pengenalan citra, identifikasi objek, dan tugas-tugas pengolahan citra lainnya dalam bidang *deep learning*. [42].



Gambar 2.3 Convolutional Layers [39]

Ilustrasi pada Gambar 2.3 memperlihatkan simulasi perhitungan pada *convolutional layer* saat memproses citra *input* berukuran 5x5 piksel dengan tiga *channel* warna (RGB) [42]. Dalam simulasi ini, digunakan *kernel* berukuran 3x3x3. Proses ini memberikan gambaran visual tentang bagaimana setiap elemen dalam *kernel* berinteraksi dengan piksel-piksel dalam citra, menciptakan *output* yang mencerminkan pengaruh *kernel* terhadap data citra.

2. Activation Layer

Activation layer adalah lapisan yang memperkenalkan non-linearitas ke dalam model [43]. *Activation layer* berperan penting dalam CNN untuk memodelkan hubungan yang kompleks dan non-linear dalam data. *Layer* aktivasi menerima *input* dari lapisan

sebelumnya, mengaplikasikan fungsi aktivasi, dan menghasilkan *output* yang kemudian diteruskan ke lapisan berikutnya. Fungsi aktivasi, seperti *ReLU* (*Rectified Linear Unit*) atau *sigmoid*, menentukan bagaimana *neuron* akan merespons terhadap stimulus. Penggunaan *layer* aktivasi memungkinkan jaringan saraf untuk mempelajari representasi yang lebih kompleks dan abstrak dari data [43]. Dalam CNN, terdapat beragam fungsi aktivasi, termasuk *ReLU*, jika diberikan *input* positif, *ReLU* mengembalikan nilai *input* tersebut; jika *input* negatif, *ReLU* mengembalikan nilai 0 [43]. Fungsi *ReLU* memperkenalkan non-linearitas ke dalam model, yang memungkinkan CNN untuk memahami dan memodelkan hubungan yang kompleks dan non-linear dalam data. Keunggulan *ReLU* termasuk kemudahan dalam pelatihan (tidak ada saturasi pada nilai tinggi seperti pada fungsi *sigmoid* atau tangen hiperbolik) dan efisiensi komputasional [44]. Berikut adalah implementasi *ReLU* dalam fungsi *sigmoid*, dihitung menggunakan rumus (1) [45].

$$ReLU(x) = \max(x, 0) \quad (1)$$

Rumus 2.1 Fungsi Aktivasi ReLU (Rectified Linear Unit)

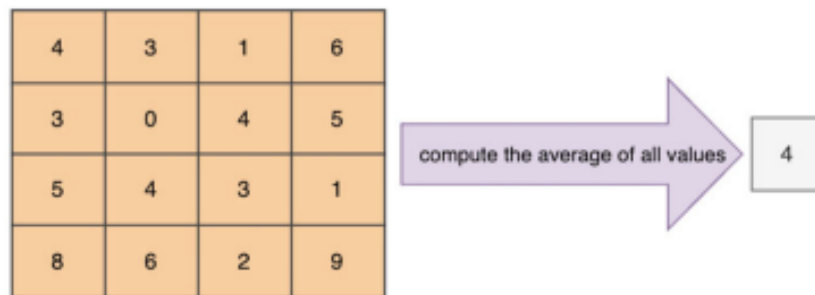
Fungsi *sigmoid* digunakan sebagai fungsi aktivasi pada lapisan tertentu, terutama pada lapisan *output* ketika model harus menghasilkan prediksi biner, seperti dalam masalah klasifikasi biner. Fungsi *sigmoid* mengubah nilai-nilai kontinu menjadi rentang antara 0 dan 1, yang dapat diinterpretasikan sebagai probabilitas. Dengan kata lain, jika *output* dari lapisan *sigmoid* mendekati 0, itu menunjukkan probabilitas rendah, sementara nilai mendekati 1 menunjukkan probabilitas tinggi [45]. Perhitungan nilai *output* ini dapat dilakukan menggunakan suatu persamaan matematika, yaitu rumus (2) [46].

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (2)$$

Rumus 2.2 Fungsi Aktivasi Sigmoid

3. Pooling Layer

Fungsi dari *pooling layer* adalah untuk mereduksi ukuran spasial representasi *output* citra dari lapisan konvolusi yang ada di tahap sebelumnya. Lapisan pooling memiliki tugas utama untuk mengurangi jumlah parameter dan komputasi dalam jaringan, serta menyederhanakan representasi data. *Pooling layer* bekerja dengan memilih nilai representatif di dalam suatu wilayah tertentu (misalnya, maksimum atau rata-rata) dan menggunakan nilai tersebut sebagai representasi untuk wilayah tersebut. *Pooling layer* yang digunakan dalam penelitian ini adalah *Global Average Pooling*, yang bekerja dengan menghitung rata-rata dari setiap peta fitur (*feature map*) untuk mereduksi dimensi spasial tanpa kehilangan informasi kontekstual [47].



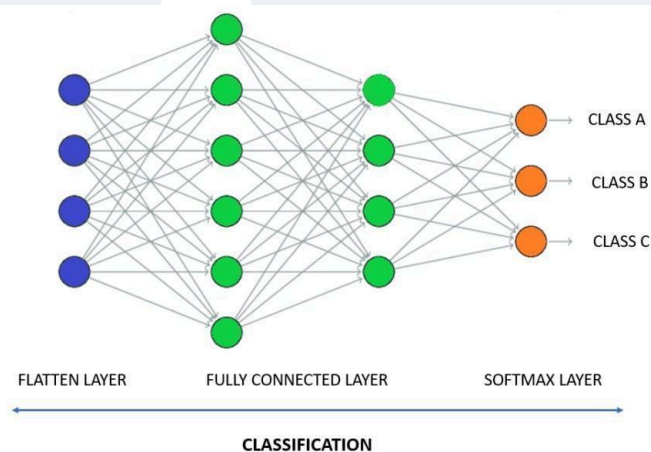
Gambar 2.4 Global Average Pooling [47]

Ilustrasi pada Gambar 2.4 menyajikan simulasi langkah kerja dari proses *global average pooling* pada citra *input* berukuran 4×4. Dalam metode ini, nilai *output* diperoleh dengan menghitung rata-rata dari seluruh nilai piksel dalam citra *input*. Tujuannya adalah untuk mereduksi dimensi data dengan mempertahankan informasi global dalam bentuk nilai rata-rata.

4. Fully Connected Layer

Lapisan *Fully Connected* merupakan bagian dari jaringan yang menghubungkan setiap *neuron* secara penuh dengan semua *neuron* pada

lapisan sebelumnya dan lapisan setelahnya. Dalam *fully connected layer*, setiap *input* dari lapisan sebelumnya dikaitkan dengan bobot (*weight*) yang unik, dan hasilnya kemudian dijumlah dan dioperasikan oleh fungsi aktivasi. Lapisan ini berperan penting dalam proses pengambilan keputusan akhir pada model *deep learning*, karena menggabungkan informasi dari seluruh fitur yang telah diekstraksi sebelumnya. Selain itu, parameter yang dimiliki oleh lapisan ini cenderung banyak, sehingga memerlukan teknik regularisasi untuk menghindari *overfitting*. Lapisan ini juga sering dipasangkan dengan teknik dropout untuk meningkatkan generalisasi model.



Gambar 2.5 Fully-connected Layers [48]

Struktur *fully-connected layer*, seperti yang terlihat pada gambar 2.5, terdiri dari tiga sub-lapisan yang memiliki peran khas. Lapisan *input* berperan sebagai penerima data awal, yaitu menerima masukan berupa nilai-nilai piksel dari citra pada saat proses identifikasi citra, dan mengirimkannya ke lapisan berikutnya. Di tengah-tengah, terdapat *hidden layer* yang berfungsi sebagai pusat pemrosesan, berada di antara lapisan *input* dan *output*. Neuron di *hidden layer* menjalankan operasi matematika kritis dan menghasilkan *output* yang selanjutnya disalurkan ke *output layer*. *Output layer*, sebagai lapisan terakhir, bertanggung

jawab menghasilkan prediksi atau *output* sesuai dengan tugas yang diberikan, misalnya, labeling kelas dalam klasifikasi citra [48].

2.3.5 Transfer Learning

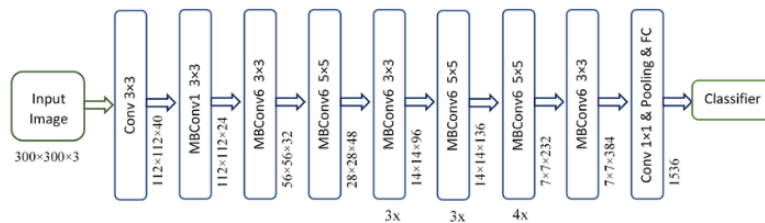
Transfer Learning adalah pendekatan dalam *machine learning* yang memanfaatkan model yang telah dilatih pada tugas tertentu (biasanya dengan dataset besar seperti *ImageNet*) digunakan kembali sebagai titik awal untuk menyelesaikan tugas baru yang memiliki keterbatasan data. Dalam CNN, *transfer learning* memungkinkan pemanfaatan fitur-fitur visual yang telah dipelajari oleh model *pretrained* untuk mempercepat dan meningkatkan performa pelatihan pada tugas klasifikasi citra baru, seperti deteksi penyakit pada citra medis [49].

Transfer learning sangat berguna dalam domain medis karena sering kali data terbatas, dan pelatihan model dari nol dapat menyebabkan *overfitting*. Pendekatan ini memungkinkan model untuk memanfaatkan pengetahuan sebelumnya dan menyesuaikannya dengan tugas spesifik, sehingga meningkatkan akurasi dan efisiensi dalam analisis citra medis [50].

2.3.5.1 EfficientNetB3

EfficientNetB3 merupakan salah satu varian dari arsitektur EfficientNet yang dikembangkan oleh Tan dan Le (2019). Arsitektur EfficientNetB3 diilustrasikan pada Gambar 2.6 terdiri dari beberapa MBConv Blocks (*Mobile Inverted Bottleneck Convolution*) dikenal karena kemampuannya dalam menyeimbangkan akurasi dan efisiensi komputasi melalui teknik *compound scaling*, yang secara bersamaan mengatur kedalaman, lebar, dan resolusi *input* jaringan. EfficientNetB3 menawarkan

performa tinggi dengan jumlah parameter yang lebih sedikit dibandingkan arsitektur CNN tradisional, menjadikannya pilihan ideal untuk aplikasi klasifikasi citra medis yang memerlukan akurasi tinggi dan efisiensi komputasi [51].

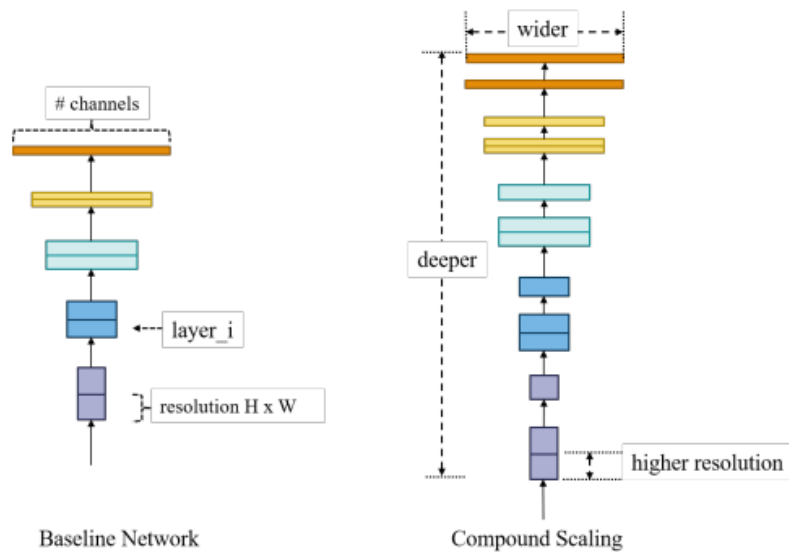


Gambar 2.6 Arsitektur EfficientNetB3 [52]

Teknik *compound scaling* dalam EfficientNetB3 secara proporsional meningkatkan ketiga dimensi utama jaringan tersebut dengan menggunakan sebuah koefisien tunggal (ϕ). Hal ini bertujuan untuk mencapai keseimbangan optimal antara kompleksitas komputasi dan performa klasifikasi. Dengan menerapkan metode ini, EfficientNetB3 mampu menghasilkan peningkatan akurasi yang signifikan tanpa memerlukan biaya komputasi yang besar, berbeda dengan pendekatan konvensional yang cenderung memperbesar hanya salah satu dimensi saja. Keunggulan tersebut menjadikan EfficientNetB3 sangat efektif digunakan pada dataset dengan kompleksitas tinggi seperti citra radiografi paru-paru, di mana kualitas hasil prediksi sangat dipengaruhi oleh kemampuan model dalam mengekstraksi fitur secara optimal [53].

Selain itu, arsitektur EfficientNetB3 yang telah dioptimasi dengan teknik *compound scaling* juga memungkinkan model untuk menangani variasi ukuran dan resolusi citra secara lebih adaptif. Hal ini sangat penting dalam analisis citra medis, seperti

X-ray paru-paru, yang seringkali memiliki kualitas dan karakteristik visual yang beragam. Dengan kapasitas ekstraksi fitur yang lebih baik, EfficientNetB3 mampu membedakan detail-detail halus yang menjadi indikator penting dalam diagnosis berbagai penyakit paru-paru. Tidak hanya itu, efisiensi penggunaan sumber daya komputasi yang dimiliki model ini membuatnya sangat sesuai diterapkan pada sistem deteksi otomatis di lingkungan klinis, termasuk pada perangkat keras dengan keterbatasan kapasitas komputasi. Dengan demikian, pemanfaatan EfficientNetB3 diharapkan dapat mempercepat proses diagnosis sekaligus meningkatkan keakuratan hasil analisis citra radiografi paru-paru [53].

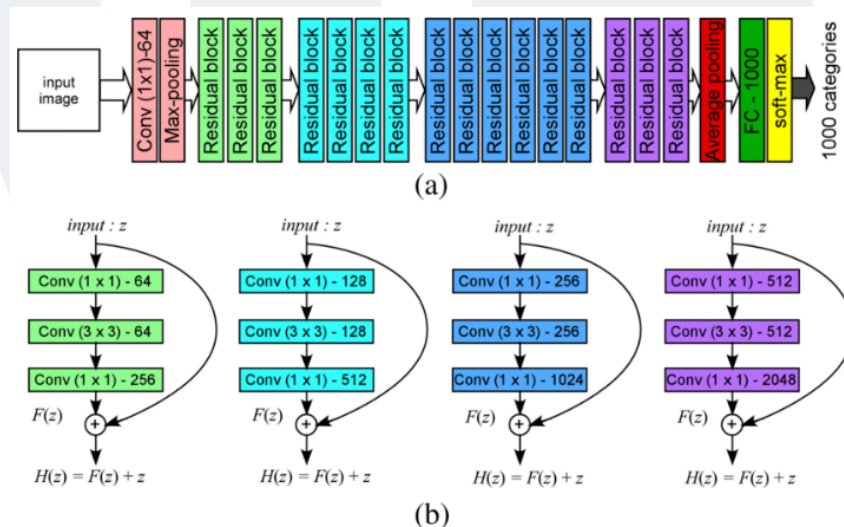


Gambar 2.7 Metode Compound Scaling [53]

2.3.5.2 ResNet50

ResNet50 (*Residual Network 50-layer*) adalah arsitektur *deep learning* berbasis CNN yang dikembangkan oleh He *et al.* pada 2015. Ciri khas dari ResNet adalah penggunaan *residual*

blocks (shortcut connections) yang memungkinkan informasi melewati beberapa *layer* tanpa mengalami degradasi. Hal ini mengatasi masalah *vanishing gradient* pada jaringan sangat dalam, sehingga memungkinkan pelatihan arsitektur yang lebih dalam dengan akurasi lebih baik. ResNet50 terdiri dari 50 *layer* dan sangat populer digunakan dalam klasifikasi citra, termasuk aplikasi medis seperti deteksi penyakit paru-paru dari X-ray atau *CT scan* [54].



Gambar 2.8 Arsitektur ResNet50 [55]

Arsitektur ResNet50, sebagaimana ditampilkan pada Gambar 2.8, dimulai dengan lapisan konvolusi awal berukuran 7×7 dengan 64 filter diikuti oleh *max-pooling*. Selanjutnya, jaringan ini terdiri dari rangkaian residual blocks yang terbagi menjadi empat tahap utama. Tahap pertama (hijau muda) terdiri dari beberapa residual blocks dengan filter berukuran 64 hingga 256, tahap kedua menggunakan filter 128 hingga 512, tahap ketiga dengan filter 256 hingga 1024, dan tahap terakhir dengan filter 512 hingga 2048 [55].

Setiap *residual block* terdiri dari tiga layer konvolusi (disebut sebagai *bottleneck layers*) dengan ukuran *kernel* 1×1 , 3×3 , dan 1×1 . *Shortcut connection* pada *residual block* menghubungkan input secara langsung ke *output* dari *block* tersebut untuk menghindari degradasi informasi selama propagasi. Setelah rangkaian residual blocks selesai, dilakukan proses *global average pooling* untuk mengurangi dimensi fitur, diikuti oleh *fully connected layer* (1000 neuron) dan fungsi aktivasi *softmax* untuk menghasilkan probabilitas klasifikasi. Dengan demikian, ResNet50 mampu mencapai performa klasifikasi tinggi dengan jumlah parameter yang relatif efisien [55].

2.3.5.3 DenseNet201

DenseNet201 merupakan salah satu arsitektur *deep learning* lanjutan berbasis CNN yang memiliki keunggulan utama berupa koneksi padat antar lapisan, dengan setiap lapisan menerima input dari semua lapisan sebelumnya. Pendekatan ini meningkatkan efisiensi propagasi fitur dan gradien, memungkinkan model belajar lebih dalam tanpa mengalami degradasi performa. Dalam beberapa tahun terakhir, DenseNet201 telah banyak digunakan dalam penelitian deteksi penyakit paru-paru dari citra X-ray dan *CT scan*. Salah satu studi oleh A. Alshoraihy (2023) mengembangkan model DenseNet201 yang ditingkatkan untuk mendeteksi pneumonia pada anak-anak, dan berhasil mencapai akurasi tinggi dengan pendekatan multi-skala konvolusi [56]. Penelitian lain oleh Upasana Chutia *et al.* (2024) menggunakan DenseNet201 dalam diagnosis multi-penyakit paru-paru seperti COVID-19, pneumonia, dan tuberkulosis, serta mengintegrasikan teknik *explainable AI* untuk meningkatkan

transparansi hasil model [57]. Selain itu, S. S. Rajput *et al.* (2023) menunjukkan bahwa DenseNet201 mampu mengklasifikasikan penyakit paru-paru dengan akurasi hingga 98,34%, mengungguli model CNN lain dalam studi yang dilakukan [58]

Layers	Output Size	DenseNet-201(k = 32)
Convolution	112 × 112	7 × 7 conv, stride 2
Pooling	56 × 56	3 × 3 max pool, stride 2
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv
	28 × 28	2 × 2 average pool, stride 2
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv
	14 × 14	2 × 2 average pool, stride 2
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Transition Layer (2)	14 × 14	1 × 1 conv
	7 × 7	2 × 2 average pool, stride 2
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$
Classification Layer	1 × 1	7 × 7 global average pool

Gambar 2.9 Arsitektur DenseNet201 [59]

Arsitektur DenseNet201 pada Gambar 2.9 terdiri dari beberapa tahap utama, dimulai dengan lapisan konvolusi awal berukuran 7×7 dengan *stride* 2, yang menghasilkan *output* berukuran 112×112, diikuti oleh operasi *max pooling* berukuran 3×3 dengan *stride* 2. Setelah lapisan awal ini, jaringan dibagi menjadi empat *blok dense* yang dipisahkan oleh *transition layers*. Setiap *dense block* berisi lapisan-lapisan konvolusi 1×1 sebagai lapisan *bottleneck* yang diikuti oleh lapisan konvolusi 3×3, yang diulang sebanyak 6, 12, 48, dan 32 kali berturut-turut untuk setiap bloknya. *Transition layers* berfungsi untuk mereduksi dimensi fitur dengan konvolusi 1×1 diikuti oleh *average pooling* berukuran 2×2. Setelah melewati rangkaian *blok dense* dan

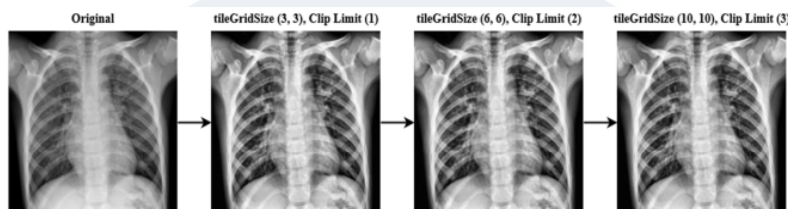
transition layers, jaringan mengimplementasikan *global average pooling* berukuran 7×7 sebelum fitur diklasifikasikan menggunakan *fully connected layer* pada akhir jaringan. Arsitektur ini memungkinkan DenseNet201 untuk secara efektif menggabungkan informasi fitur dari semua lapisan sebelumnya secara efisien, menghasilkan performa klasifikasi yang tinggi dengan jumlah parameter yang relatif rendah dibandingkan model CNN konvensional [59].

2.3.6 Contrast Limited Adaptive Histogram Equalization (CLAHE)

CLAHE merupakan singkatan dari *Contrast Limited Adaptive Histogram Equalization* merupakan teknik pengolahan citra yang digunakan untuk meningkatkan kontras suatu citra dengan mendistribusikan ulang tingkat intensitas secara adaptif [60]. Metode ini sangat efektif untuk meningkatkan kualitas visual pada gambar yang awalnya tampak kusam atau memiliki pencahayaan tidak merata. Bagian "*limited*" berarti bahwa teknik ini mencegah penguatan *noise* pada daerah dengan kontras rendah. CLAHE sering digunakan dalam bidang visi komputer dan pengolahan citra medis untuk meningkatkan visibilitas detail pada citra.

Dengan menerapkan pemerataan histogram pada blok-blok kecil citra, CLAHE mampu mengatasi tantangan kontras rendah tanpa menunjukkan *noise* berlebihan. Pendekatan adaptifnya memungkinkan operasi pada berbagai bagian citra, sementara pembatasan kontras menjaga agar peningkatan tidak berlebihan. CLAHE sering digunakan dalam penglihatan komputer dan pengolahan citra medis karena kemampuannya dalam meningkatkan visibilitas detail yang berguna untuk deteksi tepi, pengenalan objek, serta analisis citra medis seperti sinar-X atau MRI [61]. Salah satu kelebihan CLAHE adalah kemampuannya dalam mempertahankan detail halus pada area penting

citra, sehingga sangat berguna dalam proses diagnosis berbasis citra medis.



Gambar 2.9 Ilustrasi CLAHE [61]

Sebagai metode yang efektif, CLAHE memberikan kontrol yang baik terhadap kontras, sehingga menjadi solusi efektif untuk meningkatkan kualitas visual citra. Untuk menghitung histogram, pertama-tama kita perlu menghitung rata-rata jumlah piksel per wilayah dengan rumus sebagai berikut :

$$NA = (NX \times NY) / NG \quad (3)$$

Rumus 2.3 Rata-rata jumlah piksel per wilayah.

Rumus (3) menggunakan Na sebagai rata-rata piksel, Nx sebagai jumlah piksel sepanjang dimensi X, dan NY sebagai jumlah piksel sepanjang dimensi Y, dan NG adalah jumlah *grayscale*. Selanjutnya, kita dapat mendefinisikan batas klip seperti pada rumus (4) untuk membatasi histogram.

$$NCL = NA \times NNCL \quad (4)$$

Rumus 2.4 Penentuan nilai batas klip (clip limit)

Pada rumus (4.), NCL adalah batas klip dan NNCL adalah batas klip yang dinormalisasi antara 0 dan 1. Setelah itu, untuk setiap piksel, batas klip diterapkan untuk tinggi histogram menggunakan rumus (5).

$$Hi = \begin{cases} NCL & \text{if } \frac{Ni \geq NCL}{else} \\ Ni & \end{cases} \quad i = 1, 2, \dots, l - 1 \quad (5)$$

Rumus 2.5 Penerapan batas klip pada histogram

Pada rumus (5), H_i adalah tinggi histogram piksel ke- i , N_i adalah histogram ubin ke- i , dan L adalah jumlah tingkat *grayscale*. Jumlah total piksel yang di-clip dapat dihitung menggunakan rumus (6).

$$NC = (NX \times NY) - \sum_{i=0}^{L-1} H_i \quad (6)$$

Rumus 2.6 Jumlah piksel yang diklip.

Pada rumus (6), NC adalah jumlah piksel yang di-clip. Setelah menghitung NC , kita juga harus mendistribusikan kembali piksel yang di-clip tersebut. Piksel dapat didistribusikan baik secara merata maupun tidak merata. Untuk menghitung jumlah piksel yang akan didistribusikan ulang, kita dapat menggunakan rumus (7).

$$NR = NC/L \quad (7)$$

Rumus 2.7 Redistribusi piksel hasil kliping

Pada rumus (7), NR adalah jumlah piksel yang akan didistribusikan ulang. Setelah itu, histogram yang di-clip dinormalisasi menggunakan rumus (8).

$$H_i = \begin{cases} NCL N_i + NR & \text{if } \frac{N_i + NR}{else} \geq NCL \\ i = 1, 2, \dots, l - 1 \end{cases} \quad (8)$$

Rumus 2.8 Normalisasi histogram pasca-redistribusi

Jumlah piksel yang belum didistribusikan dihitung menggunakan Rumus (6) dan Rumus (7). Sampai semua piksel didistribusikan ulang, Rumus (8) diulang. Sekarang, akhirnya, histogram kumulatif dari wilayah kontekstual dapat diungkapkan dengan Rumus (9).

$$C_i = \frac{1}{(NX \times NY)} - \sum_{j=0}^i H_j \quad (9)$$

Rumus 2.9 Histogram kumulatif wilayah kontekstual

Setelah semua perhitungan selesai, histogram dari wilayah kontekstual disesuaikan dengan distribusi probabilitas yang serupa, atau sesuai dengan distribusi *Rayleigh* dan eksponensial, yang memberikan kecerahan dan kualitas visual yang telah ditentukan sebelumnya. Misalnya, terdapat piksel $P(x, y)$ dengan nilai s dan empat titik pusat yang termasuk dalam piksel yang berdekatan, yang dapat disebut sebagai $R1$, $R2$, $R3$, dan $R4$. Jumlah tertimbang dihitung untuk keempat wilayah kontekstual ini. Untuk citra *output*, piksel digabungkan dan eliminasi artefak antara piksel independen dilakukan menggunakan interpolasi bilinear. Nilai baru S yang disimbolkan sebagai S' dapat diperoleh menggunakan Rumus (10). Setelah langkah-langkah ini, akhirnya, citra yang ditingkatkan dapat diperoleh.

$$S' = (1 - y)((1 - x) \times R1(s) + X \times R2(s)) + y((1 - x) \times R3(s) + X \times R4(s)) \quad (10)$$

Rumus 2.10 Interpolasi bilinear untuk nilai piksel output

2.3.7 Evaluation Matrix

Penelitian ini menggunakan metrik evaluasi untuk menilai efektivitas model algoritma dalam pengolahan data, yang diperoleh dari *confusion matrix* dengan komponen utama yaitu *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN) [58]. *Confusion matrix* menggambarkan klasifikasi data uji dalam format tabel, menunjukkan berapa banyak prediksi yang benar (*true*) dan salah (*false*). Metrik-metrik yang dapat dihitung dari confusion matrix antara lain **akurasi**, **presisi**, **recall**, dan **F1-score**, yang masing-masing memberikan gambaran berbeda tentang performa model. Akurasi mengukur proporsi prediksi yang benar dari seluruh prediksi, sedangkan presisi dan recall menyoroti kemampuan model dalam mendeteksi kelas positif secara tepat. Dengan menggunakan

metrik-metrik ini, evaluasi model menjadi lebih komprehensif dan dapat mengidentifikasi potensi bias atau kelemahan dalam proses klasifikasi.

Tabel 2.3 Tabel Confusion Matrix

		<i>Observed</i>	
		<i>True (1)</i>	<i>False (0)</i>
Kelas Sebenarnya (Predicted Class)	<i>True (1)</i>	<i>True Positive</i> (TP)	<i>False Positive</i> (FP)
	<i>False (0)</i>	<i>False Negative</i> (FN)	<i>True Negative</i> (TN)

Dalam Tabel 2.1, terdapat *confusion matrix* yang menggambarkan nilai-nilai seperti *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). TP menggambarkan jumlah data positif yang sistem klasifikasi benar-benar identifikasi dengan benar, sementara TN mencerminkan jumlah data negatif yang sistem klasifikasi dengan tepat mengenali. Sebaliknya, FP mencakup data positif yang sistem klasifikasi keliru identifikasi, dan FN menunjukkan jumlah data negatif yang sistem klasifikasi keliru kenali [62]. *Confusion matrix* menghasilkan perhitungan untuk beberapa metrik evaluasi, termasuk tetapi tidak terbatas pada *accuracy*, *precision*, dan *recall*.

A. Accuracy

Accuracy merupakan persentase perbandingan antara jumlah data yang diklasifikasikan dengan benar (*True Positive* dan *True Negative*) dengan total seluruh data. Tingginya nilai *accuracy* dari model prediksi menunjukkan kualitas performa yang baik [62]. Persamaan untuk menghitung *accuracy* berdasarkan *confusion matrix* adalah sebagai berikut:

$$Accuracy = \frac{(TP+TN)}{TP+FP+FN+TN} \times 100\% \quad (11)$$

Persamaan 2.11 Rumus Perhitungan Accuracy

B. Precision

Precision adalah ukuran yang menunjukkan proporsi data yang diklasifikasikan sebagai positif dan benar-benar positif menurut sistem (*True Positive*) terhadap total prediksi positif [62]. *Precision* memberikan indikasi kemampuan model untuk mengidentifikasi data yang benar-benar relevan. Persamaan untuk menghitung *precision* berdasarkan *confusion matrix* dapat dijabarkan sebagai berikut:

$$Precision = \frac{TP}{TP+FP} \times 100\% \quad (12)$$

Persamaan 2.12 Rumus Perhitungan Precision

C. Recall

Recall menilai seberapa banyak data positif yang berhasil dikenali dengan benar oleh sistem dari keseluruhan data positif yang ada (*True Positive* dan *False Negative*) [62]. *Recall* mencerminkan kemampuan model untuk menemukan semua kasus yang relevan dalam data. Persamaan untuk menghitung *recall* berdasarkan *confusion matrix* dapat dijelaskan sebagai berikut:

$$Recall = \frac{TP}{TP+FN} \times 100\% \quad (13)$$

Persamaan 2.13 Rumus Perhitungan Recall

D. F1-Score

F1-score adalah metrik evaluasi yang digunakan untuk menilai keseimbangan antara *precision* dan *recall* dalam suatu model klasifikasi [62]. Persamaan untuk menghitung *F1-score* berdasarkan nilai *precision* dan *recall* adalah sebagai berikut:

$$F1-score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (14)$$

Persamaan 2.14 Rumus Perhitungan F1-score

2.4 Tools Penelitian

Pengembangan sistem klasifikasi penyakit paru-paru berbasis citra X-ray membutuhkan dukungan dari berbagai perangkat lunak dan platform pemrograman yang andal. Pemilihan *tools* yang tepat berperan penting dalam mempermudah proses pengolahan data, pelatihan model, visualisasi hasil, serta implementasi aplikasi berbasis web. Dalam penelitian ini, sejumlah perangkat lunak digunakan untuk mendukung seluruh alur kerja, mulai dari tahap awal pemrosesan data hingga *deployment* aplikasi. Beberapa *tools* utama yang digunakan meliputi *Python* sebagai bahasa pemrograman utama, *Streamlit* untuk membangun antarmuka aplikasi web interaktif, serta *GitHub* sebagai platform manajemen versi dan kolaborasi kode sumber. Penjelasan masing-masing *tools* disampaikan pada bagian berikut.

2.4.1 Python

Python, sebagai bahasa pemrograman, pertama kali diperkenalkan oleh Guido van Rossum pada tahun 1991, Dikenal sebagai bahasa pemrograman tingkat tinggi yang fleksibel dan multifungsi, sederhana untuk dipahami, dan bersifat terbuka (*open-source*), Python menjadi populer karena sintaksisnya yang bersih dan mudah dibaca, memudahkan pengembangan perangkat lunak. [63]. Bahasa Python memungkinkan penggunaan paradigma pemrograman objek, prosedural, serta fungsional, membuatnya dapat digunakan dalam berbagai jenis proyek, termasuk pengembangan web, analisis data, kecerdasan buatan, pengembangan perangkat lunak, dan lainnya. *Python* juga dikenal dengan ekosistemnya yang kaya, termasuk berbagai pustaka dan *framework* seperti *Pandas*, *NumPy*, *Matplotlib*,

Seaborn, *Scikit-learn*, dan *TensorFlow*, yang mendukung berbagai aplikasi pengembangan [64].

2.4.2 GitHub

GitHub adalah platform pengembangan perangkat lunak yang memungkinkan kolaborasi dalam pengembangan kode secara efisien. Dengan lebih dari 150 juta pengguna dan lebih dari 420 juta repositori, GitHub menjadi pusat komunitas *open-source* terbesar di dunia . Platform ini menyediakan berbagai fitur seperti kontrol versi, pelacakan isu, dan integrasi dengan alat pengembangan lainnya, yang memfasilitasi pengembangan perangkat lunak secara kolaboratif dan terstruktur [65]. Dalam penelitian ini, GitHub digunakan untuk menyimpan, mengelola, dan mendokumentasikan kode sumber dari model deteksi penyakit paru-paru yang dikembangkan. Penggunaan GitHub memungkinkan kolaborasi antara peneliti, pelacakan perubahan kode, serta distribusi kode kepada komunitas yang lebih luas untuk tujuan replikasi dan pengembangan lebih lanjut [65].

2.4.3 Streamlit

Streamlit adalah sebuah *framework open-source* untuk membuat aplikasi web interaktif dengan mudah menggunakan bahasa pemrograman Python [66]. *Framework* ini memungkinkan pengguna untuk membuat visualisasi data, dasbor, dan antarmuka pengguna yang interaktif dengan hanya beberapa baris kode. Keunggulan *Streamlit* termasuk kesederhanaan penggunaannya, otomatisasi tata letak, serta integrasi dengan berbagai pustaka visualisasi seperti *Matplotlib* dan *Plotly*. Pengguna dapat membuat aplikasi web dengan cepat tanpa harus menangani detail-detail kompleks dalam pengembangan web. Streamlit cocok digunakan untuk proyek-proyek kecil hingga menengah yang memerlukan antarmuka pengguna web yang sederhana dan efektif [66].