

BAB II

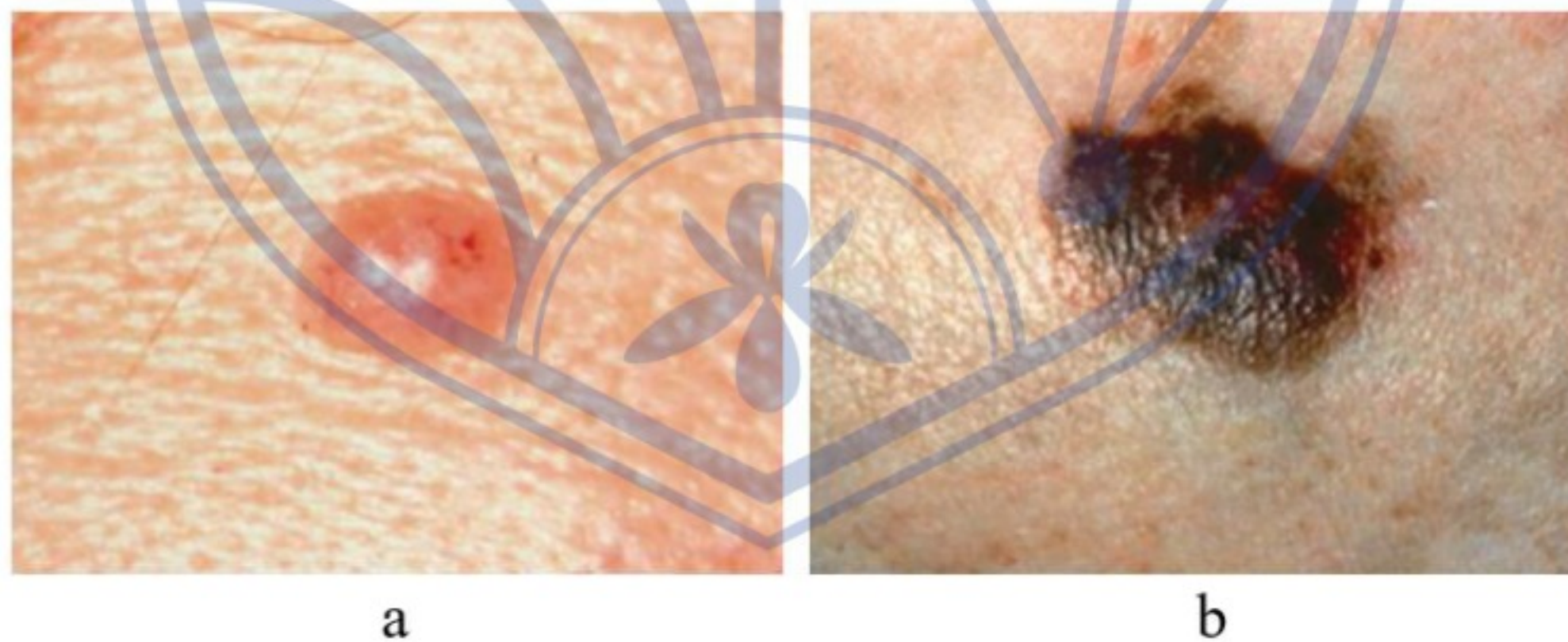
KAJIAN LITERATUR

2.1 Melanoma

Melanoma berasal dari bahasa Yunani, yaitu kata “*melas*” yang berarti gelap dan “*oma*” yang berarti tumor. Melanoma adalah jenis kanker kulit yang berbahaya dan berkembang ketika sel-sel pemberi warna pada kulit yang disebut melanosit mengalami perubahan abnormal dan tumbuh secara tidak terkendali [14] [15].

Melanoma biasanya ditandai dengan munculnya bintik atau tahi lalat yang terlihat tidak normal pada kulit, yang terkadang disertai rasa gatal atau pendarahan. Melanoma dapat berkembang dengan dua cara, yaitu menyebar ke samping di permukaan kulit atau tumbuh semakin dalam ke lapisan kulit. Pada tahap awal perkembangannya, melanoma umumnya masih terlihat datar dan tipis di permukaan kulit. Namun seiring berjalannya waktu, bentuknya berubah menjadi lebih tebal, menonjol, dan dapat dirasakan saat diraba [16].

Melanoma dapat dideteksi menggunakan metode penilaian visual, yaitu kriteria ABCDE. Kriteria ABCDE berfokus pada 5 jenis karakteristik utama tahi lalat atau lesi, yaitu pertumbuhan lesi yang asimetris (*Asymmetry*), tepi lesi yang tidak rata atau tidak jelas (*Border irregularity*), lesi memiliki lebih dari satu warna atau gradasi warna (*Color variation*), lesi berukuran lebih besar dari 6 mm (*Diameter*), serta terjadinya perubahan ukuran, bentuk, atau warna seiring waktu (*Evolving*) [15].



Gambar 2.1 (a) Bukan Melanoma (b) Melanoma [17]

2.2 Pengolahan Citra

Pengolahan citra merupakan sekumpulan metode dan algoritma komputasi yang digunakan untuk memanipulasi, menganalisis, dan meningkatkan kualitas suatu gambar

digital. Dalam proses ini, citra diubah menjadi bentuk data numerik sehingga dapat diolah oleh komputer untuk memperoleh informasi tertentu. Proses ini memungkinkan citra untuk dianalisis oleh sistem komputer, sehingga dapat membantu manusia dalam memahami dan mengambil keputusan berdasarkan informasi yang terdapat pada citra tersebut [18].

Secara umum, pengolahan citra terbagi ke dalam enam kategori utama berdasarkan tujuan dan mekanismenya, yaitu [19]:

1. Peningkatan kualitas citra (*image enhancement*)

Operasi ini difokuskan untuk memperbaiki aspek visual citra agar lebih nyaman dilihat atau lebih siap untuk diolah pada tahap berikutnya. Contoh tekniknya meliputi manipulasi kontras, penyesuaian kecerahan, hingga penajaman detail tertentu agar informasi dalam gambar terlihat lebih jelas.

2. Restorasi citra (*image restoration*)

Restorasi bertujuan untuk mengembalikan citra yang mengalami degradasi ke kondisi aslinya. Proses ini biasanya menangani masalah teknik seperti penghapusan derau (*noise*), perbaikan gambar yang kabur (*blur*), atau korelasi terhadap distorsi geometris.

3. Kompresi citra (*image compression*)

Tujuan utama dari operasi ini adalah untuk efisiensi penyimpanan dan transmisi. Kompresi bekerja dengan meminimalkan ukuran memori *file* gambar namun tetap berupaya mempertahankan kualitas visual agar tidak mengalami penurunan yang drastis.

4. Segmentasi citra (*image segmentation*)

Proses ini melibatkan pembagian atau pemisahan citra menjadi beberapa wilayah yang homogen. Pemisahan dilakukan berdasarkan kemiripan karakteristik tertentu, seperti kesamaan warna, pola tekstur, atau tingkat intensitas cahaya.

5. Analisis citra (*image analysis*)

Tahap ini dilakukan untuk mengekstraksi informasi penting dari sebuah gambar. Caranya adalah dengan mengidentifikasi fitur-fitur unik yang ada pada citra untuk kemudian diolah menjadi data atau pengetahuan yang bermanfaat.

6. Rekonstruksi citra (*image reconstruction*)

Operasi ini spesifik digunakan untuk membangun kembali citra yang utuh dari sekumpulan data mentah atau hasil tangkapan sensor yang belum berbentuk gambar sempurna. Teknik ini sangat krusial di bidang medis, seperti pada pengolahan data dari pemindaian sinar-X, *CT scan*, maupun MRI.

2.3 Kecerdasan Artifisial / *Artificial Intelligence* (AI)

Kecerdasan artifisial (*artificial intelligence*) merupakan cabang ilmu komputer yang berfokus pada pengembangan mesin dan sistem cerdas yang mampu meniru kecerdasan manusia [20] [21]. *Artificial intelligence* bekerja dengan memahami lingkungannya melalui pengumpulan, interpretasi, pemrosesan dan penalaran data sehingga sistem tersebut dapat menghasilkan prediksi, rekomendasi, maupun keputusan terbaik berdasarkan tujuan yang telah ditetapkan sebelumnya [22] [23] [24]. Dengan terus berkembangnya berbagai teori, metode, dan teknologi didalamnya, *artificial intelligence* memungkinkan mesin untuk merespons dan bertindak secara cerdas layaknya manusia dalam menghadapi berbagai situasi yang kompleks [20] [21].

Sebagai bidang ilmu yang luas, *artificial intelligence* melahirkan berbagai cabang ilmu yang saling berkaitan. Adapapun beberapa cabang ilmu dalam *artificial intelligence* antara lain [25]:

1. Pengolahan bahasa alami / *Natural Language Processing* (NLP)

NLP merupakan cabang ilmu kecerdasan buatan yang berfokus mempelajari bagaimana bahasa alami dapat diolah sehingga komputer mampu berkomunikasi menggunakan bahasa manusia. Cabang ini berfokus pada kemampuan komputer untuk memproses *input* berupa bahasa alami dan menghasilkan *output* dalam bahasa alami pula.

2. *Computer vision*

Computer vision adalah bagian dari kecerdasan buatan yang mempelajari pemrosesan gambar secara presisi. Cabang ini mencakup teknik untuk memperoleh, memproses, menganalisis dan memahami gambar, sehingga sistem dapat membangun makna dari gambar hingga ke objek fisik.

3. Sistem navigasi dan robotika

Robotika merupakan cabang ilmu kecerdasan buatan yang mempelajari desain robot, di mana robot dirancang memiliki kemampuan untuk melakukan berbagai kegiatan dan berinteraksi dengan lingkungannya menggunakan motor, sensor, dan sistem navigasi, termasuk kemampuan untuk membedakan wajah manusia melalui teknologi visi.

4. *Game*

Kecerdasan buatan dalam bidang permainan memungkinkan perancang membuat aturan yang dapat mengendalikan karakter lawan secara mandiri dan mampu beradaptasi selama permainan berlangsung.

5. Sistem pakar

Sistem pakar merupakan cabang ilmu kecerdasan buatan yang mempelajari cara membangun sistem yang mampu menyelesaikan suatu permasalahan secara mandiri.

Kecerdasan buatan telah diterapkan secara luas di berbagai sektor kehidupan sehari-hari. Dalam bidang pendidikan, AI dimanfaatkan untuk membangun sistem pembelajaran yang dapat menyesuaikan materi dengan kebutuhan setiap siswa, sekaligus membantu tenaga pengajar dalam memantau perkembangan belajar siswa dan mengelola tugas administratif secara otomatis. Di lingkungan rumah tangga, AI berperan dalam mengotomatisasikan berbagai kegiatan domestik seperti pengendalian suhu, pencahayaan, dan sistem keamanan rumah yang dilakukan melalui aplikasi digital atau perintah suara. Dalam bidang pekerjaan, AI berkontribusi dalam menyederhanakan alur bisnis, melakukan otomatisasi analisis data, serta mendorong peningkatan efisiensi operasional dan produktivitas [26]. Dalam bidang kesehatan, AI menunjukkan potensi yang sangat besar melalui kemampuannya mengolah data medis yang kompleks guna mendukung proses diagnosis hingga personalisasi pengobatan berdasarkan riwayat kesehatan pasien. Pemanfaatan AI dalam bidang keuangan mencakup berbagai kegiatan seperti pengelolaan aset, manajemen risiko, pemrosesan klaim asuransi, dan pendeteksian penipuan. Selain itu, AI juga dapat digunakan untuk memprediksi tren harga saham dengan lebih akurat [27].

2.4 Machine Learning

Machine learning merupakan cabang dari kecerdasan buatan yang memungkinkan komputer untuk belajar secara mandiri dari data tanpa perlu diprogram secara eksplisit. *Machine learning* menggunakan algoritma untuk mengolah data, mengenali pola yang terdapat di dalam data, menghasilkan kesimpulan dari analisis data, dan menjadikan kesimpulan tersebut sebagai dasar pengambilan keputusan dengan intervensi manusia yang minimal [27] [28]. *Machine learning* menggunakan model yang dapat beradaptasi agar hasil prediksinya semakin akurat dari waktu ke waktu. Meskipun dapat diterapkan pada *dataset* dengan berbagai ukuran, jumlah data yang lebih besar akan membantu dalam meningkatkan kualitas pelatihan model. Dalam penerapannya, berbagai fitur dimasukkan ke dalam model komputasi untuk menganalisis data, misalnya dengan mengelompokkan data yang mirip atau memperkirakan hasil tertentu [29].

Metode dalam *machine learning* umumnya dikelompokkan menurut keberadaan dan jenis label pada data. Berdasarkan label, *machine learning* terbagi menjadi empat kategori, yaitu [21]:

1. *Supervised learning*

Supervised learning merupakan metode *machine learning* yang menggunakan data berlabel sepenuhnya untuk membangun modelnya. Model ini belajar mengenali hubungan antara data masukan dan keluaran sehingga mampu menghasilkan prediksi yang akurat untuk data baru. Beberapa algoritma yang umum digunakan antara lain *Logistic Regression*, *Decision Tree*, *Support Vector Machines (SVM)*, dan *Naïve Bayes*.

2. *Unsupervised learning*

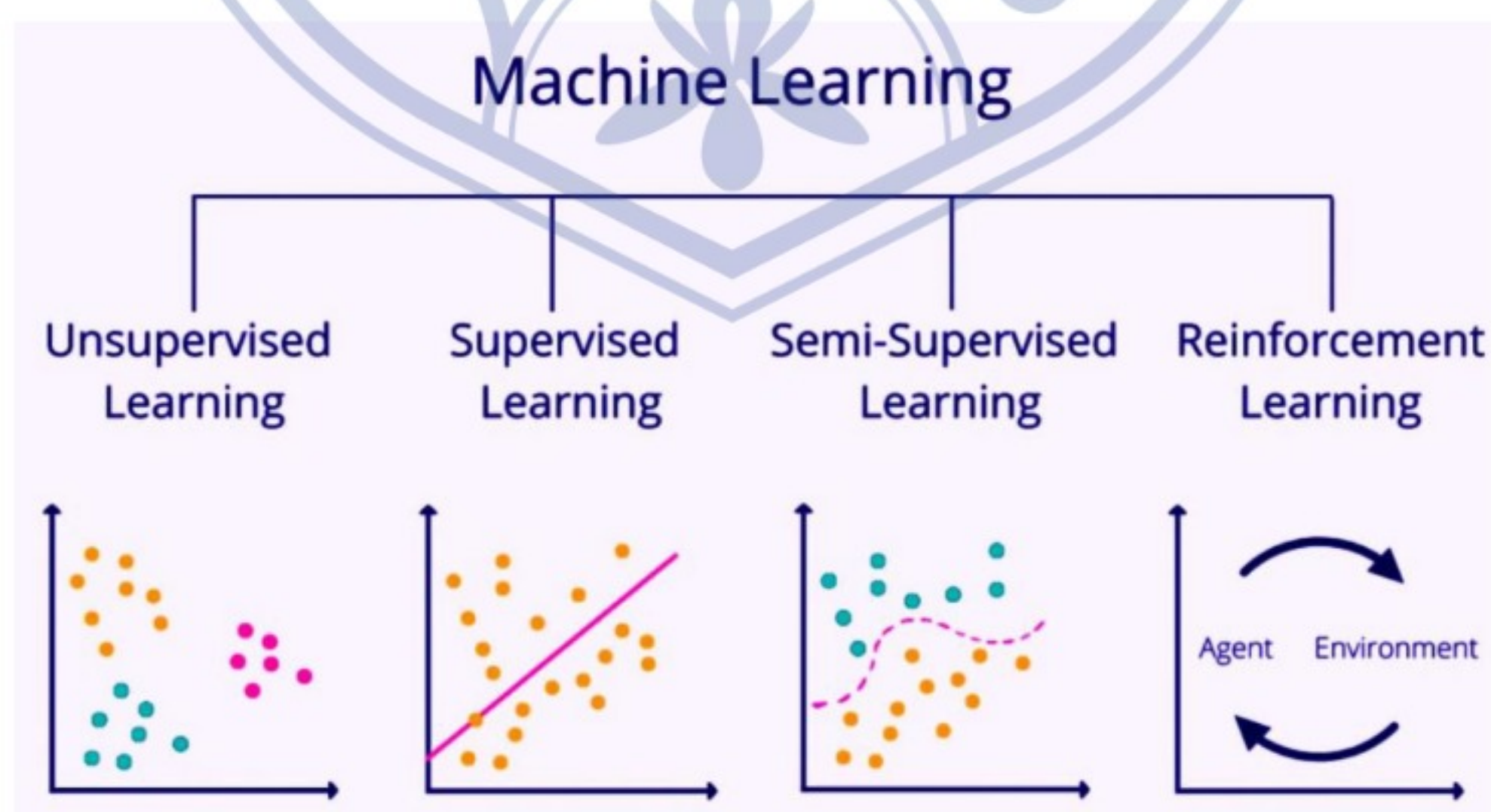
Unsupervised learning bekerja dengan data yang tidak memiliki label. Metode ini dimanfaatkan untuk menemukan pola tersembunyi dalam data melalui proses pengelompokkan, evaluasi hubungan antar variabel, dan penyederhanaan dimensi data. Algoritma yang umum digunakan mencakup *K-means Clustering* dan *Principal Component Analysis (PCA)*.

3. *Semi-supervised learning*

Semi-supervised learning merupakan pendekatan yang menggabungkan data berlabel dalam jumlah kecil dengan data tidak berlabel dalam jumlah besar. Metode ini terbukti mampu menghasilkan performa yang lebih baik dibandingkan *supervised learning* meskipun menggunakan lebih sedikit data berlabel.

4. *Reinforcement learning*

Reinforcement learning bekerja berdasarkan tujuan tertentu, dimana agen berinteraksi dengan lingkungan dan mendapatkan umpan balik berupa hadiah atau hukuman untuk melatih dirinya sendiri. Metode ini mampu membangun strategi keputusan yang dinamis dan dapat dikombinasikan dengan *deep learning* untuk menangani permasalahan yang lebih kompleks.

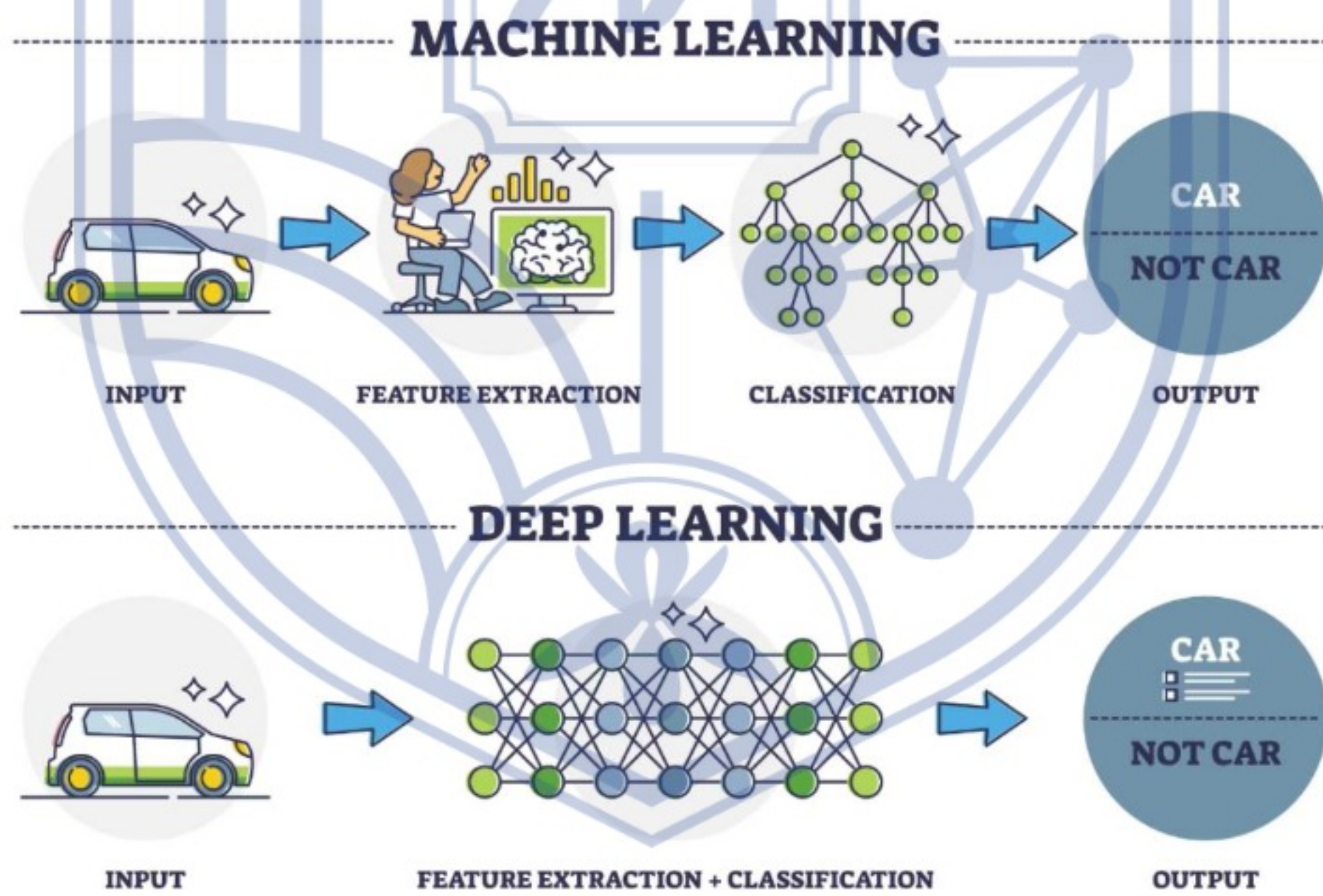


Gambar 2.2 Tipe *Machine Learning* [30]

2.5 Deep Learning

Deep learning merupakan cabang dari *machine learning* dalam bidang kecerdasan buatan yang menggunakan algoritma berbasis jaringan saraf tiruan yang terinspirasi dari mekanisme kerja otak manusia. *Deep learning* memiliki kemampuan unggul dalam mengolah data tidak terstruktur berskala besar seperti citra, teks, dan audio serta mampu mengenali pola dan menghasilkan prediksi tanpa intervensi manusia [31]. Pada klasifikasi citra medis, *deep learning* dapat belajar secara otomatis dari data citra untuk mengidentifikasi karakteristik visual yang mencurigakan tanpa memerlukan rekayasa fitur secara manual.

Machine learning tradisional seperti *Support Vector Machines* (SVM) kurang efektif digunakan untuk klasifikasi citra medis karena keterbatasannya dalam menangani data citra yang kompleks dan bervariasi, rentan terhadap *overfitting* serta kompleksitas komputasi yang tinggi [32]. Sebaliknya pendekatan berbasis *deep learning* terbukti menghasilkan akurasi yang lebih tinggi dibandingkan *machine learning* tradisional dalam tugas klasifikasi citra medis [33], sehingga menjadikannya pilihan yang lebih unggul dalam pengolahan *dataset* citra medis.



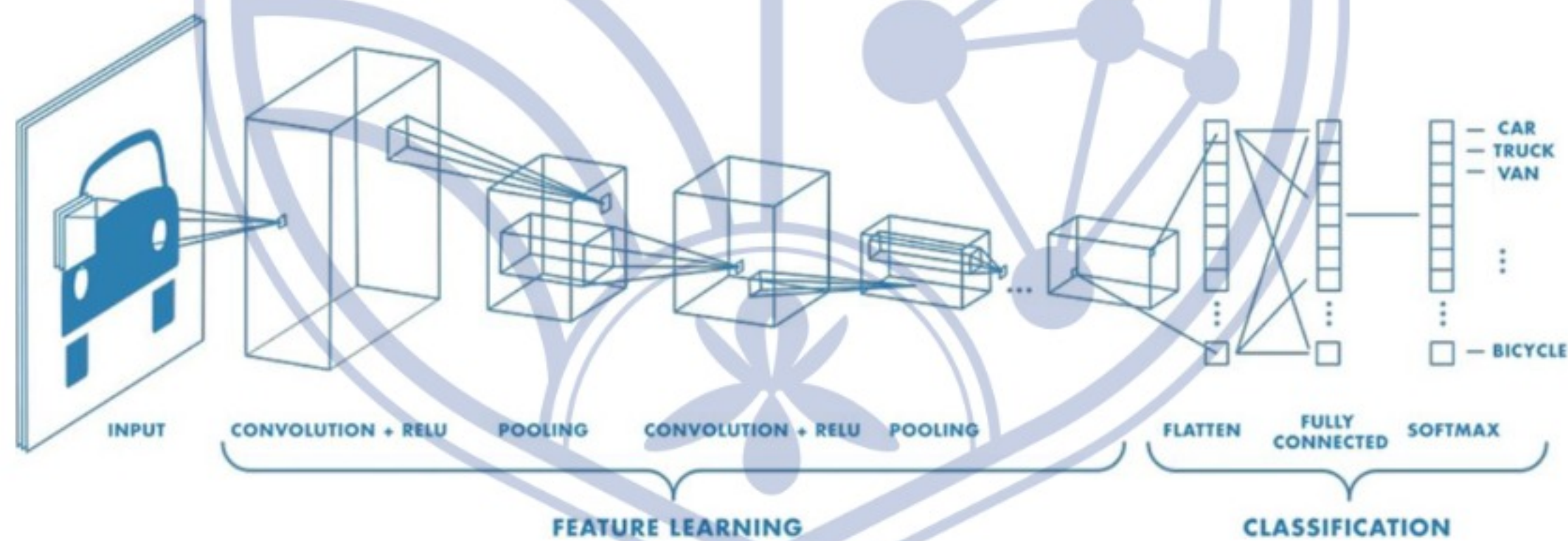
Gambar 2.3 *Machine Learning vs Deep Learning* [34]

2.6 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan salah satu jenis *deep learning* yang terinspirasi dari cara kerja korteks visual manusia, di mana komputer dilatih untuk dapat melihat dan memahami gambar seperti halnya manusia [35]. CNN merupakan

pengembangan dari metode *Artificial Neural Network* (ANN) yang dirancang khusus untuk memproses data berupa gambar, video, maupun suara [36]. CNN termasuk salah satu alat yang sangat efektif dalam bidang *computer vision* karena mampu mengenali, mengelompokkan, dan mengklasifikasikan gambar secara otomatis. CNN memiliki kemampuan unik dalam mempelajari data secara menyeluruh, mulai dari fitur sederhana seperti garis dan lengkungan, hingga fitur yang lebih kompleks seperti bentuk dan sudut suatu objek [37]. Kemampuan inilah yang membuat CNN banyak diterapkan di berbagai bidang, salah satunya untuk mendeteksi dan mengenali objek pada gambar seperti pengenalan wajah [38], dan klasifikasi jenis sampah [36].

Arsitektur CNN terdiri dari beberapa lapisan utama yang bekerja secara berurutan di mana keluaran dari satu lapisan menjadi masukan untuk lapisan berikutnya. Lapisan konvolusi bertugas mengekstraksi fitur-fitur visual dari gambar *input*. Hasil dari lapisan konvolusi kemudian diproses oleh fungsi aktivasi seperti ReLU yang berfungsi untuk menambahkan kemampuan model dalam mengenali fitur yang lebih kompleks. Selanjutnya, lapisan *pooling* bertugas menyederhanakan ukuran data tanpa menghilangkan informasi penting. Terakhir, lapisan *fully connected* berperan dalam proses pengambilan keputusan yang menghasilkan keluaran berupa klasifikasi berdasarkan data masukan yang diberikan [36], misalnya menentukan apakah suatu gambar kulit terdeteksi sebagai melanoma atau bukan.



Gambar 2.4 Arsitektur CNN [39]

2.7 MobileNetV2

MobileNetV2 merupakan salah satu arsitektur *Convolutional Neural Network* (CNN) yang dirancang khusus untuk dapat berjalan secara efisien pada perangkat *mobile* dan IoT tanpa mengorbankan akurasi yang dihasilkan [13] [40]. Arsitektur ini merupakan pengembangan dari MobileNetV1 yang menghadirkan beberapa inovasi utama sebagai berikut [40]:

1. Penggunaan *Depthwise Separable Convolution*

Depthwise separable convolution adalah teknik yang memecah proses konvolusi standar menjadi dua tahap terpisah. Pada tahap pertama, *depthwise convolution* memproses setiap kanal secara independen, kemudian dilanjutkan dengan *pointwise convolution* menggunakan filter berukuran 1x1 untuk mengintegrasikan seluruh kanal menjadi satu keluaran. Teknik ini terbukti mampu menekan jumlah parameter dan beban komputasi secara signifikan dibandingkan konvolusi konvensional.

2. Penerapan *Linear Bottleneck*

Linear bottleneck berperan menjaga kelengkapan informasi penting pada setiap lapisan transformasi dalam jaringan, sehingga representasi fitur tidak hilang selama proses komputasi berlangsung.

3. *Inverted Residual Block*

Pada *inverted residual block*, lapisan yang dihubungkan dengan *skip connection* memiliki kanal lebih sedikit dibandingkan lapisan diantaranya sehingga memungkinkan model bekerja dengan jumlah parameter yang jauh lebih sedikit sehingga lebih hemat memori.

Secara keseluruhan, arsitektur MobileNetV2 disusun dengan konfigurasi yang berbeda pada tiap tahap, sebagaimana ditunjukkan pada Tabel 2.1 [41].

Tabel 2.1 Arsitektur Jaringan MobileNetV2

<i>Input</i>	<i>Operator</i>	<i>t</i>	<i>c</i>	<i>n</i>	<i>s</i>
$224^2 \times 3$	<i>Conv2d</i>	-	32	1	2
$112^2 \times 32$	<i>Bottleneck</i>	1	16	1	1
$112^2 \times 16$	<i>Bottleneck</i>	6	24	2	2
$56^2 \times 24$	<i>Bottleneck</i>	6	32	3	2
$28^2 \times 32$	<i>Bottleneck</i>	6	64	4	2
$14^2 \times 64$	<i>Bottleneck</i>	6	96	3	1
$14^2 \times 96$	<i>Bottleneck</i>	6	160	3	2
$7^2 \times 160$	<i>Bottleneck</i>	6	320	1	1
$7^2 \times 320$	<i>Conv2d 1x1</i>	-	1280	1	1
$7^2 \times 1280$	<i>Avgpool 7x7</i>	-	-	1	-
$1 \times 1 \times 1280$	<i>Conv2d 1x1</i>	-	k	-	

Operator *bottleneck* pada Tabel 2.1 merujuk pada *inverted residual block* yang menerapkan *linear bottleneck* di bagian akhir proyeksinya. Variabel t menyatakan faktor ekspansi yang diterapkan pada jumlah *channel* masukan tiap urutan (*sequence*) *bottleneck*, c adalah jumlah *channel* keluaran, n adalah jumlah pengulangan blok dalam satu urutan dan s adalah langkah pergeseran (*stride*) pada blok pertama dalam urutan tersebut (lapisan pengulangan berikutnya otomatis menggunakan *stride* 1). Adapun k pada baris terakhir adalah jumlah kelas keluaran pada tugas klasifikasi citra. Pada setiap *bottleneck*, koneksi residual (*skip connection*) hanya dapat diterapkan apabila *stride* bernilai 1 dan jumlah *channel* masukan sama dengan jumlah *channel* keluaran, sehingga dimensi *feature map* masukan dan keluaran identik dan dapat dijumlahkan secara elemen demi elemen. Apabila *stride* bernilai 2 (menyebabkan dimensi spasial *feature map* berkurang) atau jumlah *channel* berubah meskipun *stride* bernilai 1, maka koneksi residual tidak diterapkan pada blok tersebut.

Selain itu, MobileNetV2 memiliki kompatibilitas yang tinggi dengan pendekatan *transfer learning*, di mana bobot model yang telah dilatih sebelumnya dapat dimanfaatkan kembali dan disesuaikan untuk tugas-tugas baru yang berbeda, sehingga secara signifikan mengurangi waktu dan sumber daya yang dibutuhkan dalam proses pelatihan [40]. Untuk memanfaatkan bobot model tersebut secara optimal, nilai piksel citra masukan perlu dinormalisasi menggunakan fungsi *preprocess_input* yang disediakan oleh *library* TensorFlow/Keras yang mengubah rentang nilai piksel dari $[0, 255]$ menjadi $[-1, 1]$ dengan rumus sebagai berikut :

$$x' = \frac{x}{127,5-1} \dots \dots \dots (1)$$

Dimana:

x = nilai piksel citra asli (memiliki rentang nilai dari 0 sampai 255)

x' = nilai piksel citra hasil normalisasi (memiliki rentang nilai -1 sampai 1)

2.8 Inisialisasi Bobot (*Weight Initialization*)

Keberhasilan pelatihan *deep learning* sangat bergantung pada penentuan bobot awal yang tepat guna menghindari kendala *vanishing* atau *exploding gradient* saat proses *backpropagation*. Bobot neuron diinisialisasi secara acak untuk mencegah munculnya keluaran di mana seluruh neuron memperbarui nilainya secara seragam sehingga gagal mempelajari pola data [42].

Secara umum, terdapat beberapa teknik inisialisasi bobot yang dapat diterapkan pada lapisan konvolusi maupun *fully connected*, yaitu [42]:

1. *Gaussian Random Initialization*

Teknik ini menginisialisasi matriks bobot menggunakan sampel dari distribusi Gauss (normal) dengan rata-rata (*mean*) nol dan deviasi standar yang kecil.

2. *Uniform Random Initialization*

Mirip dengan pendekatan Gauss, metode ini menggunakan sampel dari *uniform distribution* dengan rata-rata nol dan deviasi standar kecil. Kinerja distribusi Gauss dan *uniform* secara umum relatif identik. Namun, pada jaringan yang sangat dalam, kedua pendekatan acak ini rentan memicu hilangnya atau meledaknya nilai aktivasi pada propagasi maju maupun mundur.

3. *Xavier (Glorot) Initialization*

Metode ini menerapkan variansi bobot berdasarkan jumlah koneksi masuk dan keluar dari suatu neuron. Penyesuaian ini membantu menyeimbangkan sinyal *backpropagation* dan mempercepat konvergensi.

2.9 Batch Normalization

Batch normalization adalah teknik untuk menormalisasi nilai rata-rata dan variasi nilai aktivasi keluaran jaringan CNN agar terdistribusi secara Gauss standar. *Batch normalization* mempercepat proses pelatihan jaringan dalam dengan cara meredam fenomena *internal covariance shift*, yaitu kondisi dimana pergeseran distribusi aktivasi terus terjadi di tiap lapisan akibat pembaruan parameter bobot. Pergeseran distribusi yang tinggi membuat model sulit menemukan titik konvergensi karena target pembelajaran terus berubah. Melalui *batch normalization*, distribusi aktivasi tetap stabil selama pelatihan sehingga tidak hanya mempercepat konvergensi, tetapi juga menghindarkan jaringan dari masalah instabilitas seperti *vanishing/exploding gradient* [42].

Guna menjaga fleksibilitas representasi fitur, nilai aktivasi dinormalisasi terlebih dahulu lalu disesuaikan kembali nilainya menggunakan parameter yang dapat dipelajari (γ dan β) melalui persamaan berikut [42]:

$$y_j^i = \gamma_j \left(\frac{x_j^i - \mu_{x_j}}{\sqrt{\sigma_{x_j}^2 + \epsilon}} \right) + \beta_j \dots \dots \dots (2)$$

Dimana :

$$\mu_{x_j} = \text{rata-rata batch}$$

$\sigma_{x_j}^2$ = varians *batch*

ϵ = konstanta kecil

γ_j, β_j = parameter yang dipelajari

Menghitung rata-rata dari aktivasi fitur :

$$\mu_{x_j} = \frac{1}{m} \sum_{i=1}^m x_j^i \dots\dots\dots(3)$$

Dimana :

μ_{x_j} = nilai rata-rata aktivasi pada fitur/dimensi ke- j

m = jumlah sampel atau gambar dalam satu *mini-batch* (*batch size*)

i = indeks untuk sampel di dalam *batch* ($i = 1, 2, \dots, m$)

j = indeks untuk dimensi atau saluran fitur (*feature channel*)

Menghitung varians aktivasi fitur :

$$\sigma_{x_j}^2 = \frac{1}{m} \sum_{i=1}^m (x_j^i - \mu_{x_j})^2 \dots\dots\dots(4)$$

Dimana :

$\sigma_{x_j}^2$ = varians aktivasi pada dimensi ke- j

m = jumlah sampel atau gambar dalam satu *mini-batch* (*batch size*)

i = indeks untuk sampel di dalam *batch* ($i = 1, 2, \dots, m$)

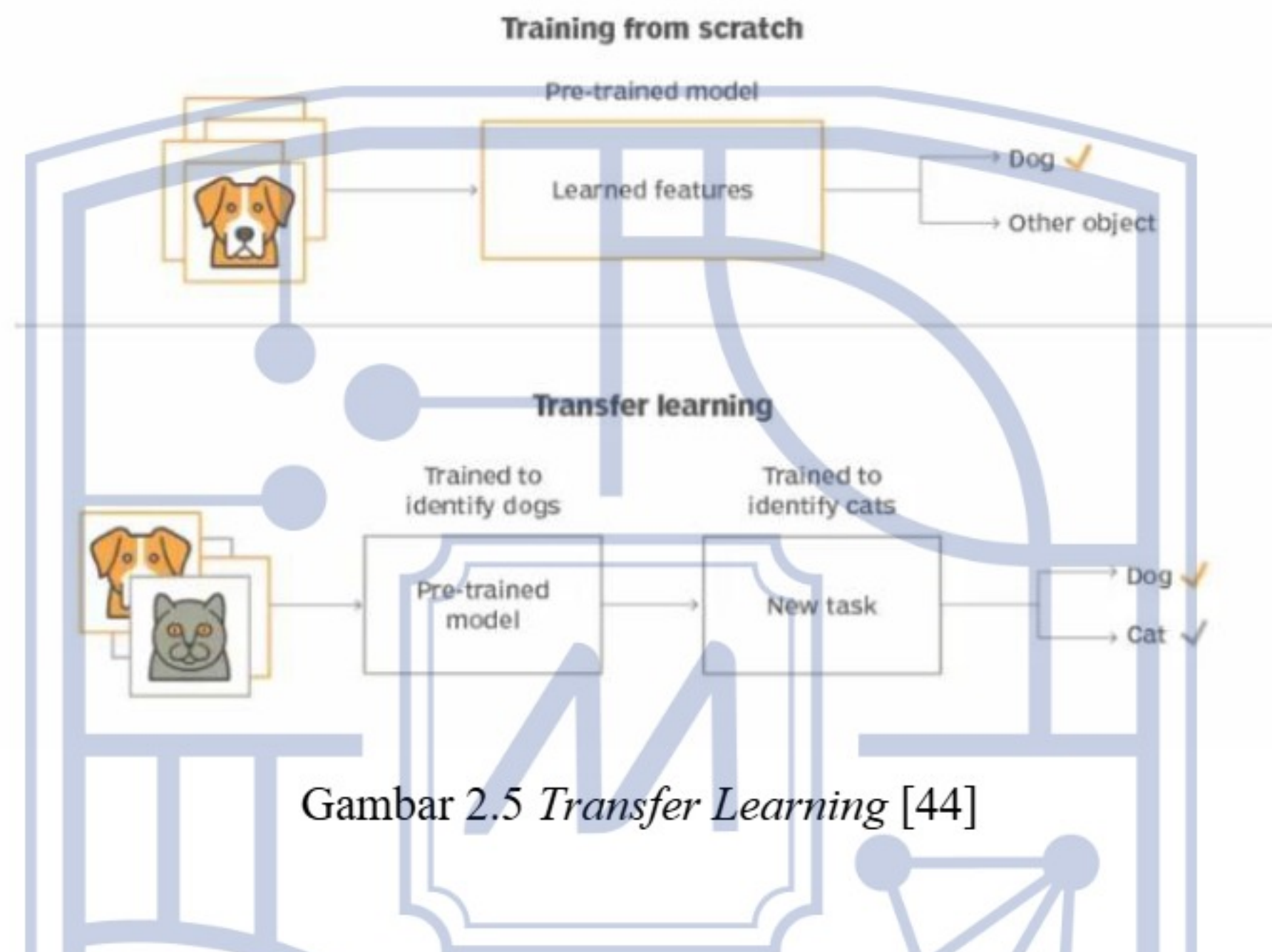
j = indeks untuk saluran fitur (*feature channel*)

2.10 Transfer Learning

Transfer learning adalah metode dalam *machine learning* yang terinspirasi dari cara belajar manusia, di mana pengetahuan yang telah dimiliki sebelumnya dapat dimanfaatkan untuk menyelesaikan masalah baru dengan lebih cepat dan efisien [43][1]. *Transfer learning* bekerja dengan memanfaatkan model CNN yang telah dilatih sebelumnya pada *dataset* berukuran besar seperti *ImageNet*, kemudian dilakukan penyesuaian agar model tersebut dapat digunakan untuk tugas baru yang berbeda [3] [28].

Salah satu keunggulan utama *transfer learning* adalah kemampuannya dalam mengurangi kebutuhan data yang besar untuk melatih sebuah model [28] [43]. Hal ini dimungkinkan karena model yang telah dilatih sebelumnya sudah memiliki pemahaman tentang fitur-fitur visual umum yang berguna untuk berbagai tugas *computer vision* [28]. Pengetahuan tersebut direpresentasikan dalam bentuk bobot jaringan yang kemudian dipindahkan ke model baru, sehingga model tidak perlu belajar dari nol dan dapat langsung memanfaatkan pengetahuan yang sudah ada sebelumnya [3].

Transfer learning juga terbukti jauh lebih efisien dibandingkan melatih model dari awal [28]. Melatih model *deep learning* dari awal dengan *dataset* yang terbatas seringkali menghasilkan akurasi yang rendah karena model tidak mendapatkan informasi yang cukup mengenai variasi data [3]. Dengan *transfer learning*, waktu dan sumber daya komputasi yang dibutuhkan dapat berkurang secara signifikan, sekaligus dapat meningkatkan performa model pada tugas yang dituju [28].



Gambar 2.5 *Transfer Learning* [44]

2.11 *Fine-Tuning*

Fine-tuning merupakan salah satu metode *transfer learning* yang dilakukan dengan mengambil seluruh atau sebagian model yang telah dilatih sebelumnya (*pretrained*) kemudian melatihnya kembali menggunakan data spesifik dari tugas baru di mana bobot model tersebut dapat diperbarui selama proses pelatihan pada tugas baru. Pendekatan ini bermanfaat khususnya ketika fitur spesifik yang dibutuhkan pada tugas baru belum sepenuhnya dipelajari oleh model *pretrained*, misalnya pada analisis citra medis di mana model CNN yang sebelumnya dilatih untuk tugas klasifikasi citra umum dilatih ulang menggunakan data citra medis berlabel untuk tugas yang lebih spesifik seperti klasifikasi jenis tumor [45].

Terdapat beberapa strategi yang dapat diterapkan dalam proses *fine-tuning*, yaitu [45]:

1. *Partial fine-tuning*

Partial fine-tuning hanya membuka (*unfreeze*) sebagian lapisan model yang umumnya lapisan-lapisan akhir yang dekat dengan *output* sementara lapisan-lapisan awal tetap dibekukan karena lapisan awal cenderung mempelajari fitur tingkat rendah yang bersifat

umum seperti garis, tepi, dan gradien, sedangkan lapisan akhir mempelajari fitur tingkat tinggi yang lebih spesifik terhadap tugas baru.

2. *Fully fine-tuning*

Fully fine-tuning membuka seluruh lapisan model sehingga baik fitur tingkat rendah maupun tinggi ikut disesuaikan terhadap data tugas baru. Pendekatan ini umumnya digunakan ketika data pada tugas target berbeda signifikan dari data yang digunakan pada proses *pre-training*.

3. *Progressive fine-tuning*

Progressive fine-tuning adalah proses pembukaan lapisan yang dilakukan secara bertahap dalam beberapa tahap pelatihan.

4. *Adaptive fine-tuning*

Adaptive fine-tuning menerapkan *learning rate* yang berbeda untuk tiap kelompok lapisan di mana lapisan yang lebih dekat ke *input* menggunakan *learning rate* lebih kecil untuk mempertahankan fitur umum yang telah dipelajari, sedangkan lapisan yang dekat ke *output* menggunakan *learning rate* lebih besar agar dapat beradaptasi lebih cepat terhadap tugas baru.

2.12 SE Attention

Mekanisme atensi (*attention mechanism*) merupakan salah satu inovasi penting dalam arsitektur *deep learning*, khususnya pada bidang pemrosesan bahasa alami (*natural language processing*) dan segmentasi citra. Konsep dasar metode ini adalah mengoptimalkan penggunaan daya komputasi secara efisien. Mekanisme ini bekerja dengan memberikan bobot nilai yang lebih besar pada area fitur yang penting, sementara area latar belakang (*background*) yang kurang relevan akan diberikan bobot yang rendah. Melalui pendekatan tersebut, keterbatasan sumber daya komputasi dapat dialokasikan secara tepat untuk meningkatkan akurasi serta kecepatan pemrosesan model. Mekanisme atensi bekerja melalui 3 tahapan utama, yaitu [46]:

1. Kompresi spasial (*squeeze operation*)

Tahap pertama ini bertujuan untuk merangkum informasi gambar. Melalui metode *global average pooling*, data gambar dua dimensi ($H \times W$) yang luas diperkecil menjadi satu nilai angka saja yang mewakili karakteristik keseluruhan dari saluran tersebut. *Output* dari proses ini berbentuk matriks satu dimensi $z = [z_1, z_2, \dots, z_c]$, yang dihitung menggunakan persamaan :

$$z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W u_c(i, j) \dots \dots \dots (5)$$

Dimana:

H = tinggi (*height*) gambar pada saluran fitur (*feature map*)

W = lebar (*width*) gambar pada saluran fitur (*feature map*)

$U_c(i, j)$ = nilai titik koordinat tertentu pada saluran fitur

2. Operasi *excitation*

Operasi *excitation* menghasilkan bobot yang berbeda untuk dialokasikan pada setiap saluran. Proses ini menggunakan dua lapisan terhubung penuh (*fully connected layers*) serta menggunakan kombinasi fungsi aktivasi ReLU dan sigmoid. Proses ini dirumuskan dengan persamaan berikut :

$$s = F_{ex}(z, W) = \sigma(W_2 \times \delta(W_1 \times z)) \dots \dots \dots (6)$$

Dimana:

z = vektor *input* hasil operasi *squeeze*

W_1 dan W_2 = matriks bobot lapisan terhubung penuh (*fully connected*)

δ = fungsi aktivasi ReLU untuk menyaring data

σ = fungsi aktivasi sigmoid (mengubah nilai menjadi skala 0 sampai 1)

3. Operasi *scale*

Tahapan akhir adalah mengalikan bobot s_c (hasil *output* dari operasi *excitation* sebelumnya) dengan nilai fitur asli u_c . Operasi ini berfungsi sebagai saringan agar model hanya fokus pada informasi penting dari setiap saluran fitur. Rumus operasi *scale* ini ditunjukkan pada persamaan berikut :

$$F_{scale}(s_c, u_c) = \tilde{X}_c = s_c \times u_c \dots \dots \dots (7)$$

Dimana:

s_c = Nilai bobot hasil operasi eksitasi pada saluran ke- c (bernilai 0 hingga 1)

U_c = Nilai fitur asli pada saluran ke- c

$\tilde{X}_c$ = Nilai fitur akhir saluran ke- c setelah disesuaikan skalanya.

2.13 Metode *Global Pooling*

Global pooling merupakan teknik yang digunakan untuk meringkas saluran fitur (*feature map*) hasil lapisan konvolusi menjadi sebuah nilai tunggal per saluran, tanpa memandang resolusi *feature map* tersebut. Dua metode *global pooling* yang umum digunakan adalah *Global Average Pooling* (GAP) dan *Global Max Pooling* (GMP). GAP

menghasilkan nilai keluaran dengan menghitung rata-rata seluruh nilai aktivasi pada *feature map* yang dapat dirumuskan sebagai berikut [47] :

$$f_{avg}(X) = \frac{1}{N \times M} \sum_{i=1}^N \sum_{j=1}^M X_{ij} \dots\dots\dots (8)$$

Dimana:

f_{avg} = nilai keluaran hasil GAP

N = lebar *feature map*

M = tinggi *feature map*

X_{ij} = nilai aktivasi pada posisi baris ke- i dan kolom ke- j

Sementara itu, GMP mengambil nilai aktivasi tertinggi pada *feature map* yang dapat dirumuskan sebagai berikut [47]:

$$f_{max}(X) = \max_{ij}(X_{ij}) \dots\dots\dots (9)$$

Dimana:

$f_{max}(X)$ = nilai keluaran hasil GMP

$\max_{ij}$ = operasi pengambilan nilai tertinggi dari seluruh X_{ij}

2.14 Focal Loss

Focal loss merupakan modifikasi dari *binary cross-entropy* yang dirancang untuk mengatasi masalah ketidakseimbangan kelas (*class imbalance*). Metode ini bekerja dengan cara menurunkan bobot kontribusi pada sampel yang mudah diklasifikasikan, sehingga model dapat lebih fokus dalam mempelajari sampel yang lebih sulit. Secara matematis, fungsi *loss binary cross-entropy* dapat ditulis ulang berdasarkan probabilitas prediksi kelas acuan (p_t) sebagai berikut [48]:

$$L_{BCE(p,y)} = CE(p_t) = -\log(p_t) \dots\dots\dots (10)$$

Dimana:

L_{BCE} = nilai *loss binary cross-entropy*

p = probabilitas hasil prediksi model (bernilai 0 sampai 1)

y = label asli *ground truth* (1 jika positif, 0 jika negative)

p_t = tingkat akurasi prediksi model terhadap label asli

Fungsi *focal loss* kemudian dibentuk dengan menambahkan faktor pemodulasi ke dalam fungsi *loss binary cross-entropy* tersebut, yang dirumuskan sebagai berikut [48]:

$$L_F(p_t) = -\alpha(1 - p_t)^\gamma \log(p_t) \dots\dots\dots (11)$$

Dimana:

L_F = nilai *focal loss*

α (alpha) = parameter pengatur keseimbangan bobot antar kelas

γ (gamma) = parameter pengendali tingkat penurunan bobot untuk sampel mudah

2.15 TensorFlow

TensorFlow merupakan *library open-source* yang dikembangkan oleh Google dan dirancang untuk membangun serta melatih berbagai jenis model pembelajaran mesin. *Library* ini menyediakan infrastruktur yang andal untuk mengembangkan berbagai model AI, mulai dari jaringan saraf untuk pengenalan gambar, teks, hingga berbagai tugas kecerdasan buatan lainnya. TensorFlow dikenal sangat fleksibel karena mendukung beragam model *deep learning* dan *machine learning*, serta dapat dijalankan di berbagai platform mulai dari komputer *desktop*, perangkat *mobile*, hingga perangkat *edge*. Dengan kemampuan tersebut, TensorFlow dapat dimanfaatkan untuk membangun model AI dalam berbagai skala, mulai dari yang sederhana hingga sangat kompleks [25].

Dari sisi cara kerjanya, TensorFlow menggunakan konsep *computational graph*, yaitu operasi matematika dalam bentuk simpul (*node*) yang saling terhubung melalui tepian (*edge*). Model AI dibangun dalam bentuk graf komputasi tersebut, dan TensorFlow mengoptimalkan proses eksekusinya sehingga memungkinkan komputasi yang lebih efisien. Selain itu, TensorFlow juga mendukung optimasi kinerja untuk berbagai kebutuhan pengembangan menggunakan teknologi seperti TensorRT [25].

2.16 Confusion Matrix

Dalam evaluasi performa model klasifikasi citra kulit, penggunaan akurasi saja tidak dapat memberikan gambaran yang cukup jelas mengenai kemampuan model secara keseluruhan. Akurasi adalah metrik yang mengukur seberapa besar proporsi prediksi yang benar dari keseluruhan prediksi yang dilakukan oleh model [32]. Meskipun akurasi memberikan informasi dasar tentang model secara menyeluruh. Dalam konteks medis, kesalahan model dalam melewatkan kasus melanoma yang sebenarnya positif atau yang disebut *false negative* (FN) dapat berdampak fatal bagi pasien karena melanoma yang tidak terdeteksi akan terus berkembang. Oleh karena itu, selain akurasi, metrik *recall*, *precision*, dan *F1-score* juga digunakan untuk memberikan gambaran evaluasi yang lebih lengkap terhadap performa model dalam mendeteksi melanoma.

Confusion matrix adalah sebuah tabel yang digunakan untuk memvisualisasikan efektivitas model klasifikasi dengan merangkum jumlah prediksi yang dihasilkan ke dalam empat kategori yaitu [32] (misalnya kelas melanoma) :

1. *True positive* (TP) : kondisi di mana model memprediksi citra kulit sebagai melanoma dan prediksi tersebut benar.
2. *True negative* (TN) : kondisi di mana model tidak memprediksi citra kulit sebagai melanoma dan prediksi tersebut benar.
3. *False positive* (FP) : kondisi di mana model memprediksi citra kulit sebagai melanoma namun prediksi tersebut salah.
4. *False negative* (FN) : kondisi di mana model tidak memprediksi citra kulit sebagai melanoma namun prediksi tersebut salah [33].

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Gambar 2.6 *Confusion Matrix*

Selain itu, *confusion matrix* dapat digunakan untuk menghitung beberapa metrik evaluasi untuk memberikan gambaran lebih lengkap tentang performa model, antara lain [33] :

1. Akurasi : mengukur persentase keseluruhan prediksi yang benar dari semua prediksi yang dilakukan model. Rumus untuk menghitung akurasi :

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \dots\dots\dots (12)$$

2. *Sensitivity (Recall)* : mengukur kemampuan model dalam mendeteksi kasus positif yang sebenarnya, dalam hal ini seberapa baik model mendeteksi melanoma yang memang benar melanoma. Rumus untuk menghitung *sensitivity (recall)* :

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

3. *Precision* : mengukur seberapa akurat model ketika memprediksi suatu citra melanoma. Rumus untuk menghitung *precision* :

$$\text{Precision} = \frac{TP}{TP+FP} \times 100\% \dots\dots\dots (14)$$

4. *F1-score* : metrik yang menggabungkan nilai *precision* dan *sensitivity (recall)* menjadi satu nilai tunggal. Metrik ini berguna untuk memberikan gambaran performa model secara lebih menyeluruh.

$$F1 - score = 2 \times \frac{(precision \times recall)}{(precision + recall)} \times 100\% \dots \dots \dots (15)$$

2.17 Penelitian Terdahulu

Suatu penelitian mengembangkan sistem deteksi dini kanker kulit menggunakan tiga model *deep learning*, yaitu CNN, DNN, dan EfficientNet yang kemudian diintegrasikan ke dalam aplikasi berbasis *web* menggunakan Flask. *Dataset* yang digunakan adalah HAM10000 yang mencakup tujuh kelas lesi kulit. Hasil evaluasi menunjukkan bahwa EfficientNet memberikan performa terbaik dengan akurasi 78,44%, diikuti CNN dengan akurasi 69,76%, sementara DNN menunjukkan performa terendah dengan akurasi 40,52%. Meskipun EfficientNet unggul secara keseluruhan, model ini masih membutuhkan peningkatan akurasi pada kelas minoritas, serta belum diuji pada data klinis baru di lingkungan nyata [49].

Penelitian lain melakukan klasifikasi kanker kulit jinak dan ganas menggunakan GoogleNet yang dimodifikasi dengan penambahan *dull razor filtering* untuk menghilangkan *noise* berupa piksel rambut halus pada citra kulit. *Dataset* yang digunakan sebanyak 600 citra kanker kulit. Hasil pengujian menunjukkan akurasi terbaik sebesar 97,79% dimana performa ini lebih unggul dibandingkan penelitian sebelumnya yang menggunakan GoogleNet tanpa *dull razor filtering* dengan akurasi 93,09%. Meskipun demikian, penelitian ini memiliki keterbatasan berupa jumlah *dataset* yang sangat terbatas serta belum diimplementasikan dalam bentuk aplikasi untuk deteksi secara *real-time* [50].

Penelitian lain melakukan klasifikasi melanoma menggunakan tiga pendekatan berbasis *transfer learning*. Pendekatan pertama menggunakan VGG19, ResNet18, dan MobileNetV2 sebagai *classifier*. Pendekatan kedua memanfaatkan ketiga model tersebut sebagai *feature extractor* yang dikombinasikan dengan empat *classifier machine learning* yaitu *support vector machines (SVM)*, *decision trees*, *naïve bayes*, dan *k-nearest neighbor (k-NN)*. Pendekatan ketiga menggabungkan seluruh model sebagai *feature extractor* dengan *classifier* yang sama. *Dataset* yang digunakan adalah ISIC 2018. Hasil terbaik diperoleh dari kombinasi *feature extraction* dan algoritma klasifikasi, yang menunjukkan performa lebih tinggi dibandingkan performa model secara tunggal. Namun, penelitian ini hanya berfokus

pada klasifikasi biner melanoma dan bukan melanoma tanpa mempertimbangkan efisiensi komputasi model yang digunakan [51].

Tabel 2.2 Tabel Perbandingan Penelitian Terdahulu

No	Metode / Model	Dataset	Hasil	Kelebihan	Keterbatasan
1	CNN, DNN, dan EfficientNet [49]	HAM 10000	EfficientNet mencapai akurasi 78,44%, CNN 69,76%, dan DNN 40,52%	EfficientNet menunjukkan performa terbaik dibandingkan model lain.	Akurasi pada kelas minoritas masih rendah serta belum diuji pada data klinis di lingkungan nyata.
2	GoogleNet dengan penambahan metode <i>dull razor filtering</i> [50]	600 citra kanker kulit	Akurasi sebesar 97,79% meningkat dari 93,09% tanpa <i>filtering</i> .	Mampu meningkatkan akurasi melalui pengurangan <i>noise</i> .	Jumlah <i>dataset</i> terbatas dan belum diimplementasikan dalam sistem deteksi <i>real-time</i> .
3	<i>Transfer learning</i> menggunakan VGG19, ResNet18, dan MobileNetV2 yang dikombinasikan dengan algoritma klasifikasi (SVM, <i>decision tree</i> , <i>naïve bayes</i> , <i>k-NN</i>) [51]	ISIC 2018	Kombinasi <i>feature extraction</i> dan <i>classifier</i> memberikan performa terbaik dibandingkan model tunggal.	Memberikan performa lebih tinggi dibandingkan penggunaan satu model saja.	Hanya fokus pada klasifikasi biner serta tidak mempertimbangkan efisiensi komputasi.

Berdasarkan kajian terhadap penelitian-penelitian terdahulu, dapat disimpulkan bahwa penerapan metode *deep learning* dalam deteksi kanker kulit melanoma telah menunjukkan hasil yang cukup baik, terutama dengan penggunaan arsitektur yang semakin kompleks. Namun demikian, masih terdapat permasalahan utama yang muncul, yaitu belum optimalnya akurasi model serta tingginya kebutuhan komputasi yang berdampak pada efisiensi model. Oleh karena itu, penelitian ini mengusulkan pengembangan model deteksi kanker kulit melanoma menggunakan arsitektur MobileNetV2 berbasis *transfer learning* yang dikenal ringan dan efisien, sehingga cocok untuk meningkatkan kinerja tanpa membebani komputasi. Untuk meningkatkan akurasi, model akan dikombinasikan dengan mekanisme *SE-attention* dan teknik *feature fusion* berupa penggabungan hasil *global average pooling* dan *global max pooling* guna memperkuat representasi fitur yang dihasilkan. Selain itu, penerapan *focal loss* dan proses *fine-tuning* dilakukan untuk mengoptimalkan performa model, sehingga diharapkan dapat menghasilkan model deteksi melanoma yang tidak hanya lebih akurat tetapi juga lebih efisien.

