BAB 2
LANDASAN TEORI

2.1 Penelitian Terdahulu

Pada bagian ini ditampilkan beberapa penelitian terdahulu tentang penggunaan MobileNetV3 yang dirangkum pada Tabel 2.1.

Tabel 2.1. Penelitian terdahulu tentang MobileNetV3

Penulis	**Judul**	**Topik**	**Hasil**
Ahmed *et al.*, 2025 [21]	Performance Comparison of Embedded AI Solutions for Classification and Detection in Lung Disease Diagnosis. Applied Sciences.	Perbandingan beberapa arsitektur CNN termasuk MobileNetV3 untuk skenario *edge deployment*, dengan pengukuran latensi inferensi langsung di perangkat *Raspberry Pi*.	MobileNetV3-Large menunjukkan performa yang kompetitif dengan *validation accuracy* 62,0% pada dataset dasar dan 63,4% pada dataset augmentasi, serta mencatat latensi inferensi 429,6 milidetik di *Raspberry Pi 4*.
Majeed *et al.*, 2024 [22]	Multi-class brain lesion classification using deep transfer learning with MobileNetV3. IEEE Access.	Klasifikasi lesi atau tumor otak multi kelas dari citra MRI menggunakan MobileNetV3 berbasis *transfer learning*.	MobileNetV3 mampu mencapai akurasi yang baik pada beberapa dataset MRI, yaitu 91,97% pada NINS 2022 dan 94,47% pada SBE-SMU, sehingga terlihat adanya variasi performa antar dataset yang digunakan.

Tabel 2.2. Penelitian terdahulu tentang MobileNetV3 (lanjutan)

Penulis	**Judul**	**Topik**	**Hasil**
Qureshi *et al.*, 2022 [23]	Intelligent ultra-light deep learning model for multi-class brain tumor detection. Applied Sciences.	Deteksi dan klasifikasi tumor otak multi kelas dari citra MRI dengan model yang dibuat sangat ringan dan cepat.	Pendekatan UL-BTD berbasis *Ultra-Light Deep Learning Architecture*, GLCM, dan SVM menghasilkan rata-rata *detection rate* 99,23% dengan waktu prediksi 11,69 milidetik per gambar.
Alisio, 2025 [17]	Klasifikasi Tumor Otak Berdasarkan Gambar MRI Menggunakan EfficientNetV2B1. Bachelor Thesis.	Klasifikasi tumor otak dari citra MRI menggunakan EfficientNetV2B1, dengan evaluasi metrik akurasi.	Model EfficientNetV2B1 mencapai akurasi 99,01%, *precision* 99,02%, *recall* 99,01%, dan *F1-score* 99,01%, sehingga dapat dijadikan pembanding pada studi lanjutan.

Ahmed *et al.* membandingkan beberapa arsitektur CNN pada skenario *edge* dan menguji model langsung di *Raspberry Pi 4*. Hasilnya menunjukkan MobileNetV3 Large bisa memberi kombinasi yang baik antara akurasi dan kecepatan, dengan latensi paling cepat 429,6 milidetik, dan semua model yang diuji tetap di bawah 2,4 detik untuk inferensi [21]. Majeed *et al.* memakai MobileNetV3 berbasis *transfer learning* untuk klasifikasi lesi otak dari citra MRI, dengan akurasi 91% pada NINS, 2022 dan 94% pada SBE SMU, sehingga terlihat bahwa performa MobileNetV3 dapat berubah tergantung jenis dataset dan cara evaluasinya [22]. Qureshi *et al.* menargetkan model dengan kebutuhan komputasi rendah dan waktu prediksi singkat untuk klasifikasi tumor otak, lalu melaporkan *detection rate* rata rata 99,23% serta waktu prediksi 11,69 milidetik per citra, yang menegaskan bahwa banyak studi model efisien juga mengejar kecepatan inferensi, bukan hanya akurasi [23]. Selain itu, Alisio melaporkan hasil klasifikasi tumor otak menggunakan EfficientNetV2B1 dengan metrik akurasi sebesar 99%, sehingga dapat menjadi pembanding yang kuat untuk melihat apakah model dengan kebutuhan komputasi yang lebih rendah seperti MobileNetV3 masih mampu mempertahankan kinerja yang baik [17].

Secara umum, penelitian-penelitian terdahulu menunjukkan bahwa MobileNetV3 telah diterapkan pada berbagai tugas klasifikasi citra, termasuk klasifikasi citra medis dan lesi otak. Hasil yang dilaporkan juga menunjukkan bahwa performa klasifikasi dapat

berbeda bergantung pada karakteristik dataset, jumlah kelas, dan rancangan eksperimen yang digunakan. Selain itu, penelitian sebelumnya menggunakan EfficientNetV2B1 telah menghasilkan performa klasifikasi yang tinggi pada klasifikasi tumor otak.

Berdasarkan penelitian-penelitian tersebut, penelitian ini mengimplementasikan MobileNetV3-Large untuk klasifikasi citra MRI tumor otak ke dalam empat kelas. Evaluasi dilakukan menggunakan metrik *accuracy*, *precision*, *recall*, *F1-score*, dan *confusion matrix*. Hasil klasifikasi terbaik kemudian dibandingkan dengan hasil EfficientNetV2B1 yang dilaporkan dalam penelitian sebelumnya.

2.2 Tumor Otak

Otak adalah pusat sistem saraf yang mengatur komunikasi antara kondisi tubuh dan perilaku. Misalnya, kebutuhan internal dan perubahan fisiologis dapat memengaruhi keputusan dan respons seseorang. Sebaliknya, aktivitas yang dilakukan juga memerlukan respons fisiologis yang rapi agar tubuh tetap stabil. Karena hubungan dua arah ini, otak tidak hanya memproses informasi, tetapi juga membantu menjaga keseimbangan fungsi tubuh supaya manusia bisa beradaptasi dengan kondisi internal maupun eksternal [1].

Secara global, gangguan pada sistem saraf berdampak besar dan makin sering dibahas sebagai prioritas kesehatan masyarakat. Analisis *Global Burden of Disease* (GBD) melaporkan bahwa pada tahun 2021 sekitar 3,40 miliar orang mengalami kondisi yang memengaruhi sistem saraf. Beban totalnya mencapai sekitar 443 juta DALYs, dan total DALYs meningkat sekitar 18,2% sejak tahun 1990 [4]. Karena itu, upaya peningkatan *brain health* didorong lewat strategi tingkat negara, termasuk rencana nasional dan kampanye kesadaran untuk memperkuat pencegahan, deteksi dini, dan penanganan gangguan yang berkaitan dengan kesehatan otak [2].

Tumor otak dapat dipahami sebagai pertumbuhan sel yang tidak normal di dalam rongga intrakranial. Pertumbuhan ini bisa berasal dari jaringan otak, atau dari struktur di sekitarnya seperti meninges dan area sekitar kelenjar hipofisis. Karena ruang di dalam tengkorak terbatas, tumor dapat menimbulkan gejala bukan hanya karena sifat selnya, tetapi juga karena menekan jaringan sekitar atau mengganggu fungsi area otak tertentu [24].

Penyebab tumor otak bisa beragam dan tidak selalu jelas pada setiap pasien. Salah satu faktor yang sering dibahas adalah paparan radiasi, termasuk dari *CT scan*. Studi berbasis populasi menunjukkan bahwa paparan *CT scan* berulang pada usia muda dapat berasosiasi dengan peningkatan risiko tumor intrakranial, terutama ketika jumlah paparannya makin banyak [25]. Walaupun kontribusinya terhadap total kasus di populasi tidak selalu besar, temuan ini tetap menekankan pentingnya penggunaan *CT scan* secara bijak dan sesuai indikasi, terutama pada anak [26]. Selain radiasi, faktor lain yang juga sering muncul adalah predisposisi genetik pada sebagian kecil kasus, misalnya sindrom

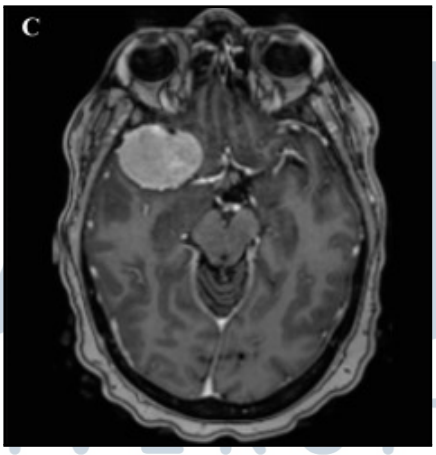
tertentu yang meningkatkan risiko tumor otak. Namun, pada banyak pasien, penyebab pastinya tetap tidak bisa ditunjuk ke satu faktor spesifik [27].

Berdasarkan asalnya, tumor pada sistem saraf pusat dapat dibagi menjadi tumor primer dan tumor sekunder. Tumor primer berasal dari jaringan sistem saraf pusat, dan klasifikasinya umumnya mengikuti kerangka WHO [28]. Tumor sekunder adalah metastasis, yaitu kanker dari organ lain yang menyebar ke otak. Dalam praktik klinis, metastasis otak sering ditemukan pada pasien kanker, sehingga penting untuk dibedakan sejak awal dari tumor primer [29].

Tumor otak yang meliputi meningioma, pituitary, dan glioma dijelaskan pada bagian ini [30].

2.2.1 Meningioma

Meningioma adalah tumor yang tumbuh dari meninges, yaitu selaput pelindung otak dan sumsum tulang belakang [24]. Contoh tampilan meningioma pada hasil MRI dapat dilihat pada Gambar 2.1. Banyak kasus meningioma bersifat jinak dan tumbuh lambat, tetapi lokasinya tetap dapat menimbulkan masalah karena ruang di dalam tengkorak terbatas. Gejala yang sering muncul antara lain sakit kepala, kejang, gangguan penglihatan, atau kelemahan pada anggota gerak, dan biasanya bergantung pada ukuran tumor serta area yang terdampak [24].



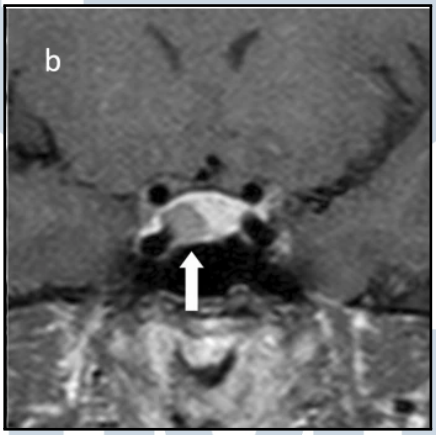
Gambar 2.1. Gambar MRI tumor otak jenis meningioma
Sumber: [24]

Dari sisi tampilan radiologis, meningioma umumnya muncul sebagai lesi *extra-axial* yang berhubungan dengan dura mater dan memiliki batas yang relatif tegas. Pada citra MRI dengan kontras, lesi ini sering menunjukkan *enhancement* yang kuat dan relatif homogen. Beberapa kasus juga dapat menunjukkan tanda *dural tail*, yaitu penebalan

atau peningkatan intensitas dura di sekitar lesi. Karakteristik lokasi, batas lesi, bentuk, dan distribusi intensitas tersebut dapat menjadi bagian dari pola visual yang dipelajari oleh model, meskipun tidak seluruh meningioma selalu menampilkan semua karakteristik tersebut [31].

2.2.2 Pituitary

Pituitary adalah tumor yang tumbuh di kelenjar hipofisis, yaitu kelenjar kecil di dasar otak yang berperan penting dalam mengatur fungsi endokrin tubuh melalui produksi hormon [32]. Contoh tampilan tumor pituitary pada hasil MRI dapat dilihat pada Gambar 2.2. Banyak tumor pituitary tidak bermetastasis, tetapi efek klinisnya dapat muncul akibat sekresi hormon berlebih atau efek massa pada struktur di sekitarnya. Keluhan yang sering muncul meliputi gangguan penglihatan, sakit kepala, dan gangguan hormonal, dengan manifestasi yang bergantung pada ukuran tumor dan jenis hormon yang terlibat [32].

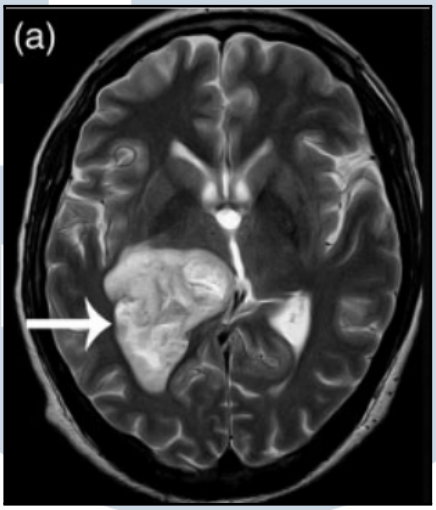


Gambar 2.2. Gambar MRI tumor otak jenis *pituitary*
Sumber: [33]

Pola visual utama tumor *pituitary* berkaitan dengan lokasinya yang berada pada daerah *sella turcica* di dasar otak. Lesi dapat menyebabkan perubahan ukuran kelenjar hipofisis, perluasan ke arah supraselar, atau pergeseran struktur di sekitarnya. Pada citra MRI, intensitas dan pola *enhancement* tumor juga dapat berbeda dari jaringan kelenjar hipofisis normal. Oleh karena itu, lokasi anatomis, bentuk lesi, ukuran, serta hubungan lesi dengan daerah selar dan supraselar dapat menjadi pola visual yang dipelajari model untuk membedakan kelas *pituitary* dari kelas lainnya [34].

2.2.3 Glioma

Glioma adalah tumor otak primer ganas pada orang dewasa yang dikenal sangat agresif [35]. Contoh tampilan glioma pada hasil MRI dapat dilihat pada Gambar 2.3. Tumor ini sering menimbulkan gejala seperti sakit kepala, kejang, gangguan kognitif, dan defisit neurologis fokal. Tingkat keparahan gejala biasanya dipengaruhi oleh lokasi tumor dan luas gangguan pada jaringan otak [35].



Gambar 2.3. Gambar MRI tumor otak jenis glioma
Sumber: [36]

Tampilan glioma pada citra MRI cenderung lebih bervariasi karena kelas ini mencakup tumor dengan tingkat dan karakteristik yang berbeda. Secara umum, glioma merupakan lesi *intra-axial* yang dapat memiliki bentuk tidak beraturan, batas infiltratif, serta distribusi intensitas yang heterogen. Terutama pada glioma derajat tinggi, citra dapat menunjukkan kombinasi jaringan tumor, edema, area nekrosis, dan bagian yang mengalami *enhancement*. Oleh karena itu, pola tekstur, ketidakteraturan kontur, heterogenitas intensitas, dan penyebaran lesi terhadap jaringan di sekitarnya dapat menjadi representasi visual yang dipelajari model. Namun, tidak seluruh glioma memiliki semua karakteristik tersebut karena tampilannya bergantung pada jenis dan tingkat tumor [37].

2.3 Convolutional Neural Network

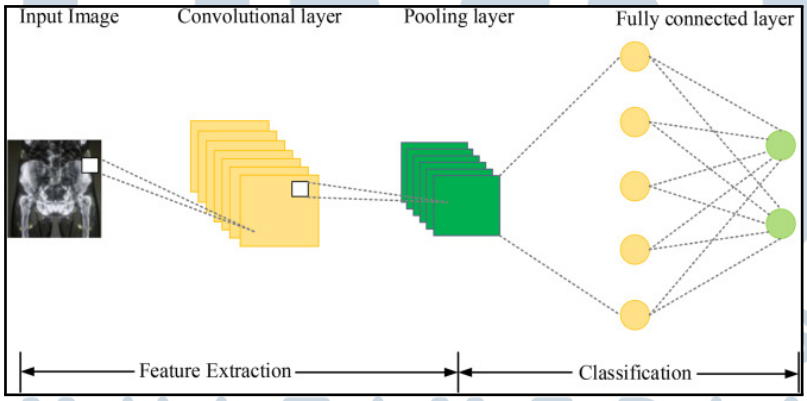
Convolutional Neural Network (CNN) merupakan jenis model *deep learning* yang telah dirancang secara khusus untuk memproses data gambar. Hal ini dikarenakan kemampuannya untuk dapat menagkap pola visual secara langsung dari piksel pada gambar tersebut tanpa perlu fitur buatan manual. CNN bekerja dengan memproses gambar lewat

lapisan-lapisan konvolusi sehingga informasi visual berubah menjadi representasi fitur yang lebih mudah dipakai untuk klasifikasi.

2.3.1 Alur kerja CNN untuk klasifikasi citra

Pada tugas klasifikasi citra, CNN umumnya dapat dipandang sebagai dua bagian utama, yaitu *feature extraction* dan *classification head*, seperti yang dapat dilihat pada Gambar 2.4. Bagian *feature extraction* bertugas mengubah citra masukan menjadi kumpulan *feature map* melalui lapisan konvolusi, lalu ukurannya diringkas lewat proses *downsampling* seperti pooling atau konvolusi ber-*stride* agar komputasi lebih efisien. Setelah itu, *classification head* menerima fitur yang sudah diringkas untuk menghasilkan prediksi kelas, biasanya berupa vektor skor (*logits*) yang kemudian diubah menjadi probabilitas tiap kelas menggunakan fungsi *softmax*. Kelas dengan probabilitas paling tinggi dipilih sebagai hasil prediksi akhir.

Lapisan awal jaringan membentuk representasi fitur tingkat rendah (*low-level features*), seperti tepi, arah garis, perubahan intensitas, dan kontras lokal. Lapisan menengah menggabungkan fitur tersebut menjadi representasi yang lebih kompleks, seperti tekstur, kontur, bentuk, serta pola hubungan antardaerah pada citra. Selanjutnya, lapisan yang lebih dalam membentuk fitur tingkat tinggi (*high-level features*) yang merepresentasikan konfigurasi massa, distribusi intensitas, lokasi spasial, dan hubungan lesi dengan struktur anatomi di sekitarnya. Representasi tersebut disimpan dalam bentuk *feature map* dan diteruskan ke bagian *classification head* untuk menentukan kelas citra.



Gambar 2.4. Alur kerja CNN untuk klasifikasi citra

Sumber: [38]

Penjelasan yang lebih rinci mengenai alur CNN disajikan pada bagian selanjutnya.

A Konvolusi

Pada CNN, proses konvolusi menggunakan filter atau *kernel* berukuran kecil yang digeser (*sliding*) di atas citra masukan untuk menangkap pola lokal di setiap area. Di setiap posisi, kernel melakukan operasi perkalian dan penjumlahan terhadap potongan citra sehingga menghasilkan keluaran berupa *feature map*, yaitu peta aktivasi yang menunjukkan seberapa kuat pola tertentu muncul pada lokasi tertentu [39]. Salah satu keunggulan utama konvolusi adalah *weight sharing*, yaitu bobot pada kernel yang sama dipakai berulang saat kernel berpindah dari satu posisi ke posisi lain, sehingga jumlah parameter yang perlu dipelajari jauh lebih sedikit dibanding lapisan *fully connected* [40]. Selain lebih efisien, keluaran konvolusi juga tetap mempertahankan susunan spasial pada *feature map*, sehingga informasi lokasi fitur di dalam citra masih dapat dimanfaatkan pada tahap klasifikasi [39].

B Downsampling

Setelah beberapa lapisan konvolusi, ukuran *feature map* masih bisa cukup besar sehingga biaya komputasi pada lapisan berikutnya ikut meningkat. Karena itu, CNN umumnya melakukan *downsampling*, misalnya melalui *pooling* (seperti *max pooling* atau *average pooling*) atau konvolusi dengan *stride* lebih dari satu, untuk menurunkan resolusi spasial fitur [41]. Ketika resolusi spasial mengecil, jumlah operasi dan kebutuhan memori juga ikut turun, sehingga proses inferensi menjadi lebih cepat dan lebih hemat sumber daya [42]. Selain efisiensi, *pooling* juga membantu membuat fitur lebih stabil terhadap perubahan kecil pada citra, karena nilai dalam satu area diringkas menjadi representasi yang lebih padat, sehingga model cenderung tidak terlalu sensitif terhadap pergeseran kecil dan dapat membantu generalisasi [41].

C Classification Head

Pada arsitektur CNN untuk klasifikasi, keluaran dari *backbone* berupa *feature map* kemudian masuk ke bagian akhir (*classification head*) untuk diubah menjadi prediksi kelas. Umumnya, *feature map* ini diringkas terlebih dahulu, misalnya dengan *flatten* atau *global average pooling* (GAP), sehingga menjadi vektor fitur yang lebih ringkas. Pada penelitian ini, setelah tahap peringkasan fitur, digunakan *batch normalization* untuk menstabilkan distribusi nilai fitur yang masuk ke lapisan berikutnya, sehingga proses pelatihan cenderung lebih stabil dan konvergen lebih cepat. Vektor fitur tersebut kemudian dipetakan melalui lapisan *dense* untuk menghasilkan skor tiap kelas (*logits*). Setelah itu, *batch normalization* kembali digunakan untuk menjaga skala dan sebaran *logits* tetap stabil sebelum tahap klasifikasi akhir, sehingga model tidak terlalu sensitif terhadap perubahan skala aktivasi antar *batch*. Selanjutnya, fungsi *softmax* mengubah *logits* menjadi probabilitas untuk tiap

kelas, sehingga nilai probabilitas seluruh kelas berjumlah 1. Kelas dengan probabilitas paling tinggi kemudian dipilih sebagai hasil prediksi, misalnya glioma, meningioma, pituitary, atau *no tumor*.

2.3.2 Proses Pelatihan CNN

Pada proses pelatihan, CNN belajar dengan menyesuaikan bobot model agar prediksi semakin mendekati label yang benar [43]. Tingkat kesalahan prediksi diukur menggunakan *loss*, yaitu nilai yang menunjukkan seberapa jauh prediksi model dari jawaban sebenarnya [44]. Pembaruan dilakukan berulang dalam satuan *batch*, yaitu sejumlah data yang diproses sekaligus dalam satu langkah pembaruan, dan *epoch*, yaitu satu putaran ketika seluruh data latih telah diproses [45]. Agar pelatihan tetap efisien dan mengurangi risiko *overfitting*, proses pelatihan dapat dihentikan lebih awal ketika performa pada data validasi tidak lagi membaik [46]. Selain itu, laju pembelajaran dapat diturunkan secara bertahap ketika peningkatan performa mulai stagnan agar pembaruan bobot menjadi lebih stabil [47].

2.4 MobileNet

Dalam praktik *computer vision*, model *deep learning* sering tidak hanya dijalankan di server, tetapi juga perlu berjalan langsung di perangkat mobile atau *embedded*. Di lingkungan seperti ini, resource biasanya terbatas, seperti RAM kecil, komputasi terbatas, dan kebutuhan hasil yang cepat. Kalau model terlalu besar, proses inferensi bisa lambat, memori besar, dan sulit dipakai secara *real-time*. Dengan kebutuhan tersebut, MobileNet diperkenalkan sebagai keluarga arsitektur CNN yang ditujukan untuk menekan kompleksitas komputasi dan ukuran model, dan tetap menjaga performa klasifikasi agar tetap kompetitif pada berbagai tugas *vision* [48].

2.4.1 Konvolusi

MobileNet membuat proses konvolusi menjadi lebih ringan dengan memecah konvolusi standar ke dalam dua tahap, yaitu *depthwise convolution* dan *pointwise convolution*. Pada *depthwise convolution*, setiap kanal diproses secara terpisah menggunakan filternya masing-masing, sehingga model hanya fokus menangkap pola spasial di dalam satu kanal. Setelah itu, *pointwise convolution* digunakan untuk menggabungkan informasi dari berbagai kanal agar model tetap mampu membentuk representasi fitur yang lebih kompleks.

Pemisahan proses ini membuat jumlah operasi dan parameter jauh lebih kecil dibanding konvolusi standar, karena pemrosesan pola spasial dan penggabungan antar kanal

tidak dilakukan sekaligus dalam satu operasi besar. Secara praktis, pendekatan ini membuat MobileNet lebih efisien untuk dijalankan pada perangkat dengan sumber daya terbatas, dengan penurunan akurasi yang umumnya kecil pada berbagai tugas pengenalan citra [48].

2.4.2 Ukuran Model

Selain efisiensi konvolusi, MobileNet juga menyediakan dua pengaturan untuk mengontrol ukuran model dan beban komputasinya, yaitu *width multiplier* dan *resolution multiplier*. *Width multiplier* digunakan untuk menipiskan jaringan secara merata di setiap lapisan, sehingga jumlah kanal pada fitur internal berkurang. Dampaknya, jumlah parameter dan biaya komputasi ikut turun, sehingga model menjadi lebih kecil dan lebih cepat.

Sementara itu, *resolution multiplier* mengecilkan ukuran input, yang kemudian memperkecil ukuran fitur internal di seluruh jaringan. Ketika resolusi input diperkecil, jumlah operasi komputasi biasanya turun cukup signifikan karena model memproses fitur dengan ukuran spasial yang lebih kecil. Dalam implementasi, pengaturan ini sering diwujudkan dengan memilih ukuran input yang lebih rendah dibanding ukuran standar, misalnya dengan memakai resolusi yang lebih kecil saat proses pelatihan atau inferensi[48].

2.4.3 MobileNetV3

MobileNetV2 melanjutkan ide MobileNet dengan memperkenalkan blok *inverted residual*, yaitu koneksi *shortcut* ditempatkan pada layer *bottleneck* yang tipis, bukan pada representasi yang lebar seperti pada ResNet [49]. Di dalam bloknya, MobileNetV2 tetap memakai *depthwise convolution* pada bagian ekspansi untuk menjaga komputasi tetap ringan, lalu memakai *linear bottleneck* dengan mengurangi atau menghilangkan *non-linearity* pada layer yang sempit supaya informasi tidak cepat hilang [49]. Paper MobileNetV2 juga menekankan konteks efisiensi dengan mengevaluasi *trade-off* antara akurasi dan jumlah operasi (*multiply-adds*), serta membandingkannya dengan latensi aktual dan jumlah parameter pada beberapa ukuran model [49]. Secara singkat, V2 adalah langkah perbaikan yang membuat blok MobileNet lebih efektif, dan menjadi jembatan sebelum masuk ke MobileNetV3 yang lebih dioptimalkan untuk perangkat mobile [49].

MobileNetV3 dikembangkan dengan target utama performa yang bagus di perangkat mobile, khususnya CPU pada ponsel. MobileNetV3 dikonfigurasikan melalui kombinasi *hardware aware neural architecture search* dan algoritma NetAdapt, lalu dirilis dalam dua varian yaitu MobileNetV3 varian *Large* dan MobileNetV3 varian *Small* [15]. Perbedaan utamanya ada pada kapasitas dan kebutuhan komputasi, di mana varian *Large* memiliki arsitektur yang lebih besar sehingga jumlah parameter dan operasi komputasinya lebih tinggi, sedangkan varian *Small* dibuat lebih ringkas untuk kondisi sumber daya yang

lebih terbatas. Secara umum, varian *Large* cenderung mengejar akurasi yang lebih tinggi, sementara varian *Small* cenderung mengejar ukuran model yang lebih kecil dan latensi yang lebih rendah pada konfigurasi standar yang dilaporkan [15].

Selain lewat pencarian arsitektur, MobileNetV3 juga menggunakan *building blocks* yang dibuat lebih efisien untuk lingkungan mobile. Secara ringkas, MobileNetV3 memanfaatkan blok yang menggabungkan operasi yang hemat komputasi seperti *depthwise convolution*, lalu menambahkan komponen efisien seperti *squeeze-and-excitation* (SE) dan modifikasi fungsi aktivasi berbasis swish yang lebih ramah untuk perangkat mobile.

Pada skenario *transfer learning*, model memanfaatkan bobot hasil pelatihan sebelumnya sebagai titik awal agar pelatihan pada dataset penelitian lebih cepat dan stabil [50]. Bagian utama model yang berperan sebagai pengekstrak fitur disebut *backbone*, sedangkan bagian akhir yang menghasilkan prediksi kelas disebut *classification head* [51]. Proses *freeze* berarti bobot pada *backbone* dikunci sehingga tidak ikut diperbarui saat pelatihan, sehingga pembelajaran difokuskan pada *classification head* [52]. Setelah *classification head* mulai stabil, tahap *unfreeze* atau *fine-tuning* dapat dilakukan dengan membuka kembali sebagian layer akhir pada *backbone* agar fitur yang dipelajari lebih sesuai dengan karakteristik data penelitian [53]. Pada tahap ini umumnya digunakan laju pembelajaran yang lebih kecil agar pembaruan bobot berlangsung lebih halus dan membantu mencegah *overfitting* pada data target yang terbatas [54].

2.4.4 Splitting Data

Pembagian data latih dan data uji merupakan salah satu bagian penting dalam evaluasi model klasifikasi. Dalam penelitian sebelumnya, beberapa rasio pembagian data telah digunakan pada model ringan maupun model klasifikasi citra secara umum. Rani dan Kumar menggunakan model MobileNet dengan variasi pembagian data dan melaporkan bahwa konfigurasi 90:10 memberikan performa terbaik pada tugas *human activity recognition* [55]. Bi *et al.* menggunakan MobileNet dengan pembagian 75% data latih dan 25% data uji pada identifikasi penyakit daun apel [56]. Selain itu, studi lain yang melibatkan MobileNetV2 juga menggunakan rasio 70:30 dalam evaluasi klasifikasi citra [57]. Untuk rasio 85:15, penelitian klasifikasi tumor otak berbasis MRI juga menunjukkan penggunaan pembagian data tersebut pada skenario klasifikasi empat kelas [58].

Berdasarkan penggunaan rasio-rasio tersebut pada penelitian sebelumnya, penelitian ini mengeksplorasi pembagian data 90:10, 85:15, 80:20, 75:25, dan 70:30 untuk melihat pengaruh komposisi data latih dan data uji terhadap performa MobileNetV3-*Large* pada klasifikasi tumor otak.

2.5 Confusion Metrics

Confusion matrix adalah matriks C yang merangkum hasil prediksi model. Setiap elemen $C_{i,j}$ menyatakan jumlah data dengan kelas asli i yang diprediksi sebagai kelas j, sehingga pola benar dan salah prediksi antar-kelas terlihat jelas dari nilai pada diagonal dan di luar diagonal.

Pada klasifikasi biner dengan kelas 0 dan 1, confusion matrix sering dinyatakan sebagai empat komponen, yaitu *true negative*, *false positive*, *false negative*, dan *true positive*. Confusion matrix dapat ditampilkan sebagai *count* atau dalam bentuk *normalized*, misalnya normalisasi per baris berdasarkan kelas asli, agar lebih mudah dibandingkan saat jumlah data tiap kelas tidak seimbang.

Pada klasifikasi biner, notasi dasar yang sering dipakai adalah TP, FP, TN, dan FN. TP adalah jumlah data kelas positif yang diprediksi positif, TN adalah jumlah data kelas negatif yang diprediksi negatif, FP adalah jumlah data kelas negatif yang salah diprediksi positif, dan FN adalah jumlah data kelas positif yang salah diprediksi negatif.

Metrik evaluasi inti yang langsung dihitung dari TP, TN, FP, dan FN adalah akurasi, presisi, *recall*, dan *F1-score*. Akurasi mengukur proporsi prediksi yang benar secara keseluruhan, presisi mengukur ketepatan prediksi positif, *recall* atau *sensitivity* atau TPR mengukur kemampuan model menemukan kelas positif, sedangkan *F1-score* merangkum keseimbangan antara presisi dan *recall*. *Accuracy* mengukur seberapa sering model memberikan prediksi yang benar pada seluruh data uji, baik untuk kelas positif maupun negatif, seperti ditunjukkan pada Rumus 2.1.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.1}$$

Precision mengukur seberapa banyak prediksi positif yang benar, dari seluruh data yang diprediksi positif oleh model, seperti ditunjukkan pada Rumus 2.2.

$$Precision = \frac{TP}{TP+FP} \tag{2.2}$$

Recall mengukur seberapa banyak data positif yang berhasil ditemukan oleh model, dari seluruh data yang sebenarnya positif, seperti ditunjukkan pada Rumus 2.3.

$$Recall = \frac{TP}{TP+FN} \tag{2.3}$$

F1-score merangkum keseimbangan antara *precision* dan *recall* dalam satu nilai,

seperti ditunjukkan pada Rumus 2.4. Nilai *F1-score* akan tinggi jika *precision* dan *recall* sama-sama tinggi, dan akan menurun jika salah satunya rendah.

$$F1\text{-}score = \frac{2TP}{2TP + FP + FN} \tag{2.4}$$