BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Perkembangan *deep learning* telah menghasilkan model dengan performa tinggi pada berbagai tugas *image processing*. Namun, peningkatan akurasi tersebut umumnya diikuti oleh peningkatan kompleksitas model, ukuran parameter, serta kebutuhan komputasi yang besar. Hal ini menjadi kendala dalam penerapan model *deep learning* pada perangkat dengan keterbatasan sumber daya, seperti perangkat *mobile*, sistem *embedded*, dan lingkungan *edge computing* [1]. Tren *deployment Artificial Intelligence* (AI) pada *edge devices* semakin meningkat seiring berkembangnya aplikasi *Internet of Things (IoT)*, *autonomous vehicles*, dan *smart devices* yang membutuhkan pemrosesan *real-time* dengan latensi rendah [2]. *Edge AI* memungkinkan pemrosesan data langsung pada perangkat tanpa bergantung sepenuhnya pada *cloud computing*, sehingga meningkatkan responsivitas, privasi, dan efisiensi bandwidth. Namun, keterbatasan daya komputasi, memori, dan energi pada perangkat *edge* mengharuskan model *deep learning* dioptimasi agar dapat dijalankan secara efisien [3]. Oleh karena itu, diperlukan teknik optimasi yang mampu meningkatkan efisiensi model tanpa mengorbankan performa secara signifikan.

Apabila model CNN dalam format *floating-point* 32-bit (FP32) diterapkan langsung pada perangkat dengan sumber daya terbatas tanpa optimasi, sejumlah konsekuensi dapat timbul. Pertama, model berpotensi tidak dapat dimuat sepenuhnya ke dalam memori perangkat. Sebagai contoh, model ResNet-18 dalam format FP32 memiliki ukuran sekitar 42,73 MB, sedangkan perangkat *edge* kelas bawah umumnya hanya memiliki RAM di bawah 512 MB yang sebagian besar telah digunakan oleh sistem operasi dan proses lainnya. Kedua, latensi inferensi yang dihasilkan dapat terlalu tinggi untuk memenuhi kebutuhan aplikasi *real-time*, karena CPU pada perangkat tersebut tidak memiliki unit pemrosesan *floating-point* yang memadai. Ketiga, konsumsi daya yang berlebihan berdampak pada ketahanan baterai perangkat. Kondisi-kondisi ini secara langsung membatasi kemampuan pengembang untuk mengintegrasikan model AI ke dalam aplikasi yang berjalan di perangkat dengan keterbatasan *hardware*.

Salah satu pendekatan yang banyak digunakan adalah *model compression*,

khususnya teknik *quantization*. *Model compression* mencakup berbagai teknik seperti *pruning*, *knowledge distillation*, *low-rank factorization*, dan *quantization*, yang masing-masing memiliki *trade-off* berbeda antara tingkat kompresi dan akurasi [4]. *Quantization* mengurangi presisi representasi bobot dan aktivasi model, misalnya dari *floating-point 32-bit* (FP32) menjadi *integer 8-bit* (INT8), sehingga ukuran model dapat diperkecil dan waktu inferensi dapat dipercepat [5, 6]. FP32 merupakan format presisi standar yang digunakan dalam *training* dan *inference neural networks*, di mana setiap parameter direpresentasikan menggunakan 32 bit. *Quantization* memungkinkan pengurangan presisi menjadi FP16 yang secara teori akan memangkas setengah ukuran memori, atau INT8 yang hanya asumsinya hanya akan menggunakan seperempat ukuran memori FP32.

Terdapat dua pendekatan utama dalam *quantization*, yaitu *Post-Training Quantization* (PTQ) dan *Quantization-Aware Training* (QAT) [5, 7]. PTQ diterapkan pada model yang telah terlatih dan tidak memerlukan proses *training* ulang, sehingga lebih efisien secara komputasi. Namun, PTQ berpotensi mengalami penurunan akurasi yang lebih signifikan, terutama pada *quantization* dengan *bit-width* sangat rendah seperti 4-bit atau 2-bit [7]. Sebaliknya, QAT mengintegrasikan simulasi *quantization* selama proses *training*, sehingga model dapat beradaptasi terhadap efek *quantization* dan mempertahankan akurasi yang lebih tinggi [5]. Meskipun demikian, penurunan presisi tetap berpotensi menurunkan akurasi model. Dampak *quantization* tidak selalu seragam pada setiap arsitektur, sehingga diperlukan studi komparatif yang terkontrol untuk mengevaluasi pengaruhnya secara komprehensif [4, 8].

Penelitian terkait *quantization* pada CNN telah menunjukkan hasil yang beragam. Studi menunjukkan bahwa arsitektur yang lebih efisien seperti *Mobile Network* (MobileNet) cenderung lebih sensitif terhadap *quantization* dibandingkan arsitektur yang lebih besar seperti *Residual Network* (ResNet), serta memerlukan *Quantization-Aware Training* (QAT) untuk mempertahankan akurasi pada *integer 8-bit* (INT8) [8, 9]. Sebaliknya, beberapa penelitian melaporkan bahwa ResNet dapat mencapai hasil yang memuaskan bahkan dengan *Post-Training Quantization* (PTQ) [7]. Namun, meskipun berbagai penelitian telah mengkaji dampak *quantization* pada arsitektur CNN tertentu, kajian yang secara sistematis membandingkan dampak *quantization* pada arsitektur dengan tujuan desain berbeda, yaitu *accuracy-oriented* dan *efficiency-oriented*, menggunakan metode *quantization* yang sama dalam kondisi eksperimen yang terkontrol, masih relatif terbatas, khususnya pada *dataset* dan tugas yang identik.

Perlu dicatat pula bahwa dampak nyata *quantization* terhadap latensi inferensi sangat bergantung pada lingkungan *hardware* yang digunakan. Sebagai contoh, format FP16 umumnya diasumsikan lebih cepat dari FP32 karena ukuran data yang lebih kecil. Namun, asumsi ini hanya berlaku pada *hardware* yang memiliki dukungan komputasi *half-precision* secara *native*, seperti GPU modern. Pada lingkungan CPU yang umum ditemukan di perangkat *edge* dan *mobile*, operasi FP16 dapat dijalankan melalui emulasi yang justru menambah *overhead* komputasi [10]. Oleh karena itu, evaluasi dampak *quantization* perlu mempertimbangkan lingkungan *hardware* secara eksplisit agar hasilnya dapat diinterpretasikan secara tepat. Berdasarkan hal tersebut, penelitian ini bertujuan untuk mengevaluasi dampak *quantization* terhadap akurasi, ukuran model, dan waktu inferensi pada ResNet-18 dan MobileNetV2, serta mengkaji *trade-off* antara performa dan efisiensi yang muncul tanpa mengasumsikan keunggulan salah satu arsitektur.

Dalam konteks CNN, *Residual Network* (ResNet) merupakan salah satu arsitektur yang banyak digunakan sebagai *baseline* dalam berbagai penelitian karena kemampuannya mengatasi permasalahan degradasi performa pada jaringan yang semakin dalam melalui konsep *residual learning* dan *skip connection* [11]. Mekanisme ini memungkinkan informasi dan *gradient* mengalir lebih efektif selama proses pelatihan sehingga jaringan yang lebih dalam tetap dapat dioptimalkan dengan baik. Salah satu varian yang paling banyak digunakan adalah ResNet-18, yang terdiri dari 18 lapisan dengan sekitar 11,7 juta parameter. ResNet-18 menawarkan keseimbangan antara kompleksitas model dan akurasi sehingga sering digunakan sebagai acuan (*baseline*) dalam penelitian *computer vision*.

Di sisi lain, *Mobile Network* (MobileNet) dikembangkan dengan fokus utama pada efisiensi komputasi dan penggunaan memori untuk perangkat dengan sumber daya terbatas [12]. Berbeda dengan ResNet yang mengutamakan kemampuan representasi fitur, MobileNetV2 memanfaatkan *depthwise separable convolution*, *inverted residual*, dan *linear bottleneck* untuk mengurangi jumlah parameter serta operasi komputasi tanpa menurunkan akurasi secara signifikan. Dengan sekitar 3,4 juta parameter, MobileNetV2 menjadi salah satu arsitektur yang banyak digunakan pada perangkat *mobile* dan *edge* karena mampu memberikan kompromi yang baik antara efisiensi dan performa.

Perbedaan filosofi desain tersebut menjadikan ResNet-18 dan MobileNetV2 representatif sebagai objek penelitian. ResNet-18 mewakili arsitektur yang berorientasi pada akurasi (*accuracy-oriented*), sedangkan MobileNetV2 mewakili arsitektur yang berorientasi pada efisiensi (*efficiency-oriented*). Dengan

membandingkan kedua arsitektur menggunakan metode *quantization*, *dataset*, serta kondisi eksperimen yang sama, penelitian ini bertujuan mengevaluasi bagaimana perbedaan filosofi desain tersebut memengaruhi *trade-off* antara akurasi, ukuran model, dan latensi inferensi.

1.2 Rumusan Masalah

Berdasarkan latar belakang dan konteks penelitian yang telah dipaparkan sebelumnya, berikut adalah rumusan masalah untuk penelitian ini:

1. Bagaimana pengaruh *quantization* terhadap akurasi model ResNet-18 dan MobileNetV2?
2. Seberapa besar perubahan ukuran model dan waktu inferensi setelah penerapan *quantization*?
3. Bagaimana perbandingan *trade-off* antara performa dan efisiensi pada ResNet-18 dan MobileNetV2?

1.3 Batasan Permasalahan

Berikut adalah batasan permasalahan yang diterapkan dalam penelitian untuk memperjelas ruang lingkup pembahasan serta memastikan penelitian tetap terarah dan terkontrol:

1. Model yang digunakan dibatasi pada dua arsitektur CNN, yaitu ResNet-18 dan MobileNetV2.
2. Teknik *quantization* yang digunakan dibatasi pada FP16 dan INT8 dengan metode *Post-Training Quantization*.
3. *Dataset* yang digunakan adalah CIFAR-10 dan CIFAR-100 sebagai pembanding.
4. Evaluasi kinerja difokuskan pada tiga metrik utama: akurasi, ukuran model, dan waktu inferensi.
5. Pengukuran latensi inferensi dilakukan di lingkungan CPU dan tidak mencakup pengukuran di GPU atau perangkat *mobile* nyata.
6. Penelitian tidak membahas teknik optimasi lain seperti *pruning*, *knowledge distillation*, atau kombinasi metode kompresi.

1.4 Tujuan Penelitian

Berikut adalah tujuan penelitian yang ingin dicapai sebagai upaya untuk menjawab rumusan masalah yang telah ditetapkan:

1. Menganalisis dampak *quantization* dari FP32 ke FP16 dan INT8 terhadap akurasi pada model ResNet-18 dan MobileNetV2.
2. Mengukur perubahan ukuran model dan waktu inferensi setelah penerapan *quantization*.
3. Membandingkan *trade-off* antara performa dan efisiensi pada ResNet-18 dan MobileNetV2.

1.5 Manfaat Penelitian

Berikut adalah manfaat yang diharapkan dari hasil penelitian, baik dari sisi akademis maupun praktis:

1. Memberikan analisis dan komparasi mengenai dampak *quantization* terhadap akurasi, perubahan ukuran model, dan waktu inferensi pada dua arsitektur CNN dengan karakteristik berbeda, sehingga memperkaya kajian tentang *model compression* dalam *deep learning*.
2. Memberikan dasar pertimbangan awal dalam memilih format presisi numerik dengan mempertimbangkan *trade-off* antara ukuran model, kecepatan inferensi, dan akurasi pada skenario inferensi berbasis CPU.
3. Berkontribusi sebagai studi komparatif terkontrol yang mendokumentasikan perilaku *quantization* pada arsitektur CNN ringan di lingkungan CPU, melengkapi literatur yang ada yang umumnya berfokus pada model berskala lebih besar atau lingkungan GPU.

1.6 Sistematika Penulisan

Sistematika penulisan laporan adalah sebagai berikut:

- Bab 1 PENDAHULUAN
  Bab ini memuat latar belakang penelitian, rumusan masalah, tujuan penelitian, batasan penelitian, serta sistematika penulisan.

- Bab 2 LANDASAN TEORI

  Bab ini membahas landasan teori dan tinjauan pustaka yang relevan dengan penelitian, meliputi konsep *deep learning*, *Convolutional Neural Network*, ResNet, MobileNet, serta *quantization*.

- Bab 3 METODOLOGI PENELITIAN

  Bab ini menjelaskan metodologi penelitian yang digunakan, termasuk desain eksperimen, arsitektur model, *dataset*, skema *quantization*, serta metrik evaluasi yang digunakan.

- Bab 4 HASIL DAN DISKUSI

  Bab ini menyajikan hasil eksperimen dan pembahasan terkait pengaruh *quantization* terhadap akurasi, ukuran model, dan waktu inferensi pada arsitektur ResNet-18 dan MobileNetV2.

- Bab 5 KESIMPULAN DAN SARAN

  Bab ini berisi kesimpulan dari hasil penelitian yang telah dilakukan serta saran untuk pengembangan penelitian selanjutnya.