Meningkatkan Efisiensi Energi Perangkat Edge melalui Optimasi Pruning dan Kuantisasi Model.

Nambi Sembilu [a,1,*], Iqbal Ramadhani Mukhlis [a,2], Iswanda Fauzan Satibi [b,3]

[a] Program Studi Sistem Informasi, Fakultas Ilmu Komputer, UPN Veteran Jawa Timur, Surabaya, Indonesia
[b] Program Studi Bisnis Digital, Fakultas Ilmu Komputer, UPN Veteran Jawa Timur, Surabaya, Indonesia [1] Email Penulis
[1] nambi.si@upnjatim.ac.id *; [2] iqbal.ramadhani.fasilkom@upmjatim.ac.id; [3] satibi.if@upnjatim.ac.id

Submission: 13/04/2026, Revision: 19/04/2026, Accepted: 04/05/2026

Abstract

Edge computing devices are increasingly tasked with performing artificial intelligence inference under strict constraints on processing capacity and power consumption. This study evaluates magnitude-based weight pruning and dynamic quantization as practical model compression techniques for energy-efficient edge AI deployment. MobileNetV2, pretrained on ImageNet, was adapted to the CIFAR-10 classification task and compressed under three configurations: 40% L1 unstructured pruning followed by recovery fine-tuning (Prune40), dynamic INT8 post-training quantization (QuantINT8), and a sequential combination of both (Prune+Quant). All experiments were executed on a physical Intel N150 mini PC with a thermal design power of 6 watts, using PyTorch 2.1 in CPU-only inference mode. Results show that Prune40 reduced inference latency by 17.9% while simultaneously improving classification accuracy by 1.04 percentage points, attributed to the implicit regularisation effect of sparse weight removal and recovery fine-tuning. QuantINT8 yielded moderate latency savings (6.6%) with negligible accuracy loss. The combined pipeline achieved the lowest absolute latency at a marginal energy overhead. These findings establish magnitude pruning with recovery training as the most effective single-step compression strategy for low-power x86 edge platforms.

Keywords: *edge computing; model compression; pruning; quantization; green AI*

Abstrak

Perangkat edge computing dewasa ini menghadapi tuntutan yang semakin besar untuk menjalankan inferensi kecerdasan buatan di bawah keterbatasan kapasitas komputasi dan konsumsi daya. Penelitian ini mengevaluasi pemangkasan bobot berbasis magnitudo dan kuantisasi dinamis sebagai teknik kompresi model yang praktis untuk penerapan edge AI yang hemat energi. MobileNetV2 yang telah dilatih pada ImageNet diadaptasi ke tugas klasifikasi CIFAR-10, kemudian dikompresi dalam tiga konfigurasi: pemangkasan L1 tidak terstruktur 40% disertai recovery fine-tuning (Prune40), kuantisasi pascapelatihan INT8 dinamis (QuantINT8), dan pipeline gabungan keduanya secara berurutan (Prune+Quant). Seluruh eksperimen dijalankan pada mini PC Intel N150 (TDP 6 W) menggunakan PyTorch 2.1 berbasis CPU. Prune40 mereduksi latensi inferensi 17,9% sekaligus meningkatkan akurasi klasifikasi 1,04 poin persentase, yang dikaitkan dengan efek regularisasi implisit dari pemangkasan bobot sparse dan recovery fine-tuning. QuantINT8 menghasilkan penghematan latensi sedang (6,6%) dengan penurunan akurasi yang dapat diabaikan. Pipeline gabungan mencapai latensi absolut terendah dengan overhead energi yang kecil. Temuan ini menegaskan bahwa pemangkasan magnitudo dengan recovery training merupakan strategi kompresi langkah tunggal yang paling efektif untuk platform edge x86 berdaya rendah.

Kata kunci: *komputasi tepi; kompresi model; pemangkasan bobot; kuantisasi; green AI*

1. Pendahuluan

Perkembangan ekosistem Internet of Things (IoT) yang pesat dalam satu dekade terakhir telah menggeser paradigma komputasi kecerdasan buatan dari pusat data terpusat menuju perangkat tepi yang tersebar luas. Pendekatan ini, yang lazim disebut sebagai *edge computing*, menawarkan sejumlah keunggulan strategis: latensi respons yang lebih rendah, perlindungan privasi data yang lebih baik, serta pengurangan beban trafik jaringan secara signifikan [1]. Meski demikian, penerapan model deep learning pada perangkat tepi

menghadapi tantangan yang tidak ringan: keterbatasan kapasitas prosesor, memori yang terbatas, dan anggaran daya yang ketat, acapkali hanya berkisar beberapa watt.

Persoalan ini semakin krusial bila dipertimbangkan dari sudut pandang lingkungan. Badan Energi Internasional (IEA) memproyeksikan konsumsi listrik pusat data secara global dapat melampaui 945 terawatt-jam per tahun pada 2030, suatu angka yang hampir setara dengan kebutuhan energi tahunan Jepang [1,2]. Pada tataran model individual, Strubell et al. mendokumentasikan bahwa pelatihan satu sistem pemrosesan bahasa alami berskala besar dapat menghasilkan emisi karbon dioksida melebihi 550 metrik ton [3]. Realitas ini mendorong munculnya paradigma *Green AI* yang menempatkan efisiensi energi sebagai metrik utama, berdampingan dengan akurasi model [4].

Dalam konteks tersebut, dua teknik kompresi model yang paling banyak dikaji adalah pemangkasan bobot (*weight pruning*) dan kuantisasi (*quantization*). Pemangkasan bekerja dengan mengeliminasi parameter yang berkontribusi paling kecil terhadap prediksi model, menghasilkan jaringan yang lebih ramping tanpa mengorbankan kemampuan diskriminatifnya secara signifikan. Kuantisasi menurunkan presisi representasi numerik bobot dari floating-point 32-bit ke bilangan bulat 8-bit, memangkas kebutuhan memori dan beban komputasi aritmatika. Berbagai analisis empiris menunjukkan bahwa pipeline gabungan keduanya berpotensi menghemat energi inferensi tepi hingga 70% dibandingkan model tanpa kompresi [5].

Kendati demikian, sebagian besar kajian kompresi dalam literatur masih dilaksanakan dalam lingkungan simulasi atau pada akselerator AI khusus seperti GPU dan NPU. Evaluasi yang dilakukan secara langsung pada CPU x86 berdaya rendah -- khususnya jenis prosesor yang umum digunakan pada mini PC komersial, sistem otomasi industri, dan node IoT -- masih sangat terbatas [6,7]. Penelitian ini hadir untuk mengisi kesenjangan tersebut dengan menjalankan eksperimen secara langsung pada perangkat keras fisik Intel N150, sebuah prosesor dengan konsumsi daya hanya 6 watt.

Adapun kontribusi utama penelitian ini mencakup tiga pp. Pertama, evaluasi empiris pemangkasan L1 tidak terstruktur dan kuantisasi INT8 dinamis, baik secara individual maupun gabungan, pada platform CPU tepi Intel N150 yang sesungguhnya. Kedua, analisis kuantitatif terhadap trade-off akurasi, latensi, energi, dan ukuran model dari empat konfigurasi berbeda. Ketiga, panduan praktis bagi para pengembang yang menyasar penerapan *Green Edge AI* pada perangkat keras x86 berdaya rendah.

Sistematika penulisan artikel ini disusun sebagai berikut. Bagian 2 memaparkan tinjauan pustaka yang mendasari pendekatan kompresi model modern, mencakup perkembangan teknik pemangkasan, kuantisasi, dan kerangka Green AI. Bagian 3 menguraikan metode penelitian yang meliputi rancangan eksperimen, spesifikasi perangkat keras, dataset, konfigurasi kompresi, serta protokol evaluasi. Bagian 4 menyajikan hasil empiris beserta pembahasan kritis atas setiap konfigurasi dengan memperhatikan aspek akurasi, latensi, konsumsi energi, dan ukuran model. Bagian 5 menutup artikel dengan rangkuman temuan utama dan rekomendasi arah penelitian lanjutan. Alur yang demikian diharapkan memberikan pemahaman yang menyeluruh, baik terhadap hasil akhir penelitian maupun pertimbangan metodologis yang mendasari setiap keputusan teknis yang diambil.

Tantangan yang dihadapi perangkat tepi dalam menjalankan model kecerdasan buatan tidak hanya terbatas pada keterbatasan komputasi, namun juga meliputi dimensi lain yang sering kali luput dari perhatian. Pertama, keragaman perangkat keras tepi, mulai dari mikrokontroler berbasis ARM Cortex-M, mini PC berbasis Intel dan AMD, hingga perangkat berbasis RISC-V, menghadirkan fragmentasi ekosistem yang menyulitkan pengembangan solusi yang bersifat universal [7]. Kedua, keterbatasan termal pada banyak perangkat tepi menuntut desain model yang tidak hanya hemat energi dalam arti per inferensi, tetapi juga mampu mempertahankan kinerja yang stabil selama operasi berkelanjutan tanpa mengalami throttling akibat panas berlebih. Ketiga, keterbatasan memori tertanam, yang pada perangkat kelas rendah dapat berkisar antara 512 MB hingga 4 GB, menuntut representasi model yang efisien baik pada saat pemuatan maupun eksekusi runtime.

Selain aspek teknis, dimensi ekonomi juga memegang peranan penting dalam keputusan penerapan edge AI. Pada skala produksi masal perangkat IoT, perbedaan konsumsi daya beberapa miliwatt per perangkat dapat berakumulasi menjadi penghematan operasional yang signifikan [2]. Lebih jauh, kemampuan untuk memproses data secara lokal tanpa bergantung pada konektivitas cloud yang konsisten menjadi faktor pembeda yang kritis bagi aplikasi di daerah dengan infrastruktur jaringan yang terbatas, seperti sistem pemantauan

pertanian presisi atau perangkat medis untuk layanan kesehatan jarak jauh. Faktor-faktor inilah yang menjadikan penelitian tentang kompresi model bukan sekadar latihan akademik, melainkan kebutuhan praktis yang berdampak langsung pada adopsi teknologi secara luas.

Tren ekonomi global juga turut mendorong urgensi penelitian di bidang ini. Laporan industri terkini memperkirakan pasar perangkat keras edge AI akan tumbuh pada laju majemuk tahunan sekitar 25 persen dalam periode 2024 hingga 2030, didorong oleh permintaan kuat dari sektor manufaktur cerdas, kendaraan otonom, pengawasan keamanan berbasis kamera, serta perangkat medis portabel [2]. Namun pertumbuhan ekspansif ini hanya akan berkelanjutan apabila inferensi AI dapat dijalankan secara efisien pada perangkat keras yang relatif murah dan berdaya rendah. Tanpa terobosan dalam efisiensi model, biaya energi operasional dan kebutuhan pendinginan dapat dengan cepat menjadi penghambat utama penyebaran AI di lapangan. Penelitian seperti yang disajikan dalam artikel ini menjadi fondasi rekayasa yang menentukan apakah visi pertumbuhan tersebut dapat terwujud secara nyata atau berhenti pada angan-angan.

2. Tinjauan Pustaka

Perkembangan teknik kompresi model dapat ditelusuri kembali ke awal era deep learning modern, ketika ukuran model yang membengkak mulai menjadi kendala utama bagi adopsi di lingkungan dengan sumber daya terbatas. Torralba dkk. [19] mengumpulkan 80 juta citra gambar sebagai sumber dataset berskala besar, yang kemudian menjadi fondasi benchmark CIFAR-10. Dari arsitektur AlexNet dengan sekitar 60 juta parameter pada ImageNet [20], pengembangan model terus berkembang menuju arsitektur yang semakin besar seperti VGG-16 dengan 138 juta parameter. Lonjakan ukuran model ini menimbulkan kesadaran akan pentingnya kompresi, yang awalnya didekati melalui teknik post-hoc seperti pemangkasan koneksi redundan dan kemudian berkembang menjadi pendekatan yang lebih terintegrasi dengan proses pelatihan. Saat ini, lanskap kompresi model telah berkembang menjadi disiplin yang matang dengan beragam teknik yang dapat dipilih berdasarkan karakteristik aplikasi target, dari mikrokontroler hingga server inferensi berdaya tinggi [11].

Fondasi teoretis kompresi model modern diletakkan oleh Han et al. [8] melalui pipeline *Deep Compression* yang mengintegrasikan pemangkasan berbasis ambang magnitudo, kuantisasi codebook, dan pengkodean Huffman. Pendekatan terpadu ini berhasil mencapai pengurangan ukuran model antara 35 hingga 49 kali lipat pada arsitektur AlexNet dan VGG-16 tanpa penurunan akurasi yang terukur. Temuan tersebut membuktikan bahwa mayoritas parameter dalam jaringan yang kelebihan parameterisasi bersifat redundan. Guo et al. [9] memperluas gagasan ini melalui teknik pemangkasan dinamis adaptif, sementara Frantar et al. [10] mendemonstrasikan bahwa model bahasa berukuran raksasa sekalipun dapat dipangkas secara agresif dalam satu langkah tanpa pelatihan ulang penuh [11].

Di ranah kuantisasi, Jacob et al. [12] menunjukkan bahwa *quantization-aware training* (QAT) menghasilkan retensi akurasi INT8 yang jauh lebih superior dibandingkan kuantisasi pascapelatihan konvensional. Benchmark standar industri seperti MLPerf Power [13] kini memungkinkan perbandingan efisiensi energi inferensi yang terstandar lintas platform dan arsitektur, mendorong komunitas riset untuk mengoptimalkan kompresi model secara lebih sistematis dan transparan [17].

MobileNetV2 [14] merupakan salah satu desain jaringan saraf konvolusional yang paling banyak diadopsi untuk perangkat tepi. Blok *inverted residual* dengan koneksi pintas dan konvolusi separable kedalamannya secara dramatis mengurangi operasi perkalian-akumulasi. Survei sistematis Antonini et al. [7] mengkonfirmasi bahwa arsitektur CNN kompak yang dikombinasikan dengan pemangkasan dan kuantisasi menghasilkan metrik inferensi terbaik di berbagai kelas perangkat keras tepi, dengan potensi penghematan energi berkisar 20 hingga 70 persen [15,16].

Dari perspektif keberlanjutan, Schwartz et al. [4] melalui kerangka *Green AI* dan Strubell et al. [3] yang mengkuantifikasi jejak karbon model NLP berskala besar, telah mendorong komunitas riset memperlakukan efisiensi energi sebagai metrik kolas pertama. Proyeksi IEA [1] dan Goldman Sachs Research [2] memperkirakan sekitar 60 persen peningkatan permintaan listrik pusat data hingga 2030 akan dipasok dari bahan bakar fosil. Njoku et al. [17] menunjukkan bahwa model yang terkuantisasi pada perangkat keras tepi nyata secara konsisten menghasilkan metrik energi per inferensi yang lebih baik dibandingkan padanan FP32-nya [18].

Di luar dua teknik utama yang menjadi fokus kajian ini, sejumlah pendekatan komplementer juga telah berkembang pesat dalam upaya menghadirkan deep learning pada perangkat berdaya rendah. Distilasi

pengetahuan (*knowledge distillation*) mentransfer kemampuan prediktif dari model berskala besar kepada model yang lebih ringkas melalui mekanisme pembelajaran guru-murid. Pendekatan faktorisasi peringkat-rendah (*low-rank factorization*) memecah matriks bobot yang besar menjadi perkalian beberapa matriks yang lebih kecil, sehingga menekan jumlah parameter sekaligus mempertahankan kapasitas representasi. Sementara itu, pencarian arsitektur otomatis (*neural architecture search*/NAS) mengeksplorasi ruang desain jaringan secara sistematis untuk menemukan arsitektur yang optimal pada batasan sumber daya tertentu [7].

Meskipun masing-masing teknik memiliki keunggulan tersendiri, literatur terkini mengindikasikan bahwa penerapan gabungan beberapa teknik kompresi justru memberikan hasil yang lebih konsisten dibanding teknik tunggal [6]. Namun demikian, kombinasi yang lebih banyak tidak selalu berarti lebih baik, karena setiap teknik memiliki overhead runtime tersendiri yang dapat berinteraksi secara tidak terduga pada arsitektur perangkat keras tertentu [7,17]. Oleh karena itu, penelitian ini secara disengaja membatasi fokus pada kombinasi pemangkasan dan kuantisasi yang paling umum diterapkan, sekaligus mengukur dampak aktualnya pada perangkat keras x86 berdaya rendah melalui eksperimen terkontrol.

Cheng dkk. [11] menyusun survei komprehensif tentang kompresi dan akselerasi model deep learning yang mencakup pemangkasan berbasis magnitudo dan terstruktur, kuantisasi, dekomposisi matrik berpangkat rendah, dan distilasi pengetahuan. Kajian ini secara eksplisit menganalisis trade-off antara laju kompresi dan penurunan akurasi pada berbagai arsitektur CNN yang umum digunakan, memberikan panduan praktis bagi peneliti dan praktisi dalam memilih teknik yang tepat sesuai kendala perangkat keras target. Lebih jauh, Xu dkk. [6] memperluas cakupan tersebut dengan mengevaluasi lebih dari 100 teknik optimasi model on-device, mengkonfirmasi bahwa kombinasi pemangkasan dan kuantisasi secara konsisten menghasilkan efisiensi inferensi terbaik lintas berbagai platform tepi.

Di sisi lain, Aljaloud dkk. [18] menggarisbawahi bahwa pengurangan konsumsi energi model AI tidak hanya berdampak pada biaya operasional, tetapi juga berkontribusi pada target pengurangan emisi karbon secara keseluruhan. Mereka memperkirakan bahwa adopsi luas teknik kompresi model pada inferensi edge dapat mengurangi jejak karbon infrastruktur AI global hingga 15 sampai 30 persen dalam dekade mendatang. Temuan seperti ini memperkuat motivasi etis dan lingkungan dari penelitian kompresi model, menggeser posisinya dari domain teknis murni menuju instrumen pencapaian tujuan pembangunan berkelanjutan.

3. Metode Penelitian

3.1. Rancangan dan Platform Eksperimen

Penelitian ini menggunakan rancangan eksperimen komparatif dengan empat kondisi perlakuan yang dijalankan pada perangkat keras fisik. Pemilihan platform sengaja menghindari akselerator AI khusus agar hasil dapat digeneralisasikan ke kelas luas perangkat mini PC komersial dan node IoT industri. Perangkat yang digunakan adalah mini PC berbasis prosesor Intel N150 (Alder Lake-N) dengan konsumsi daya 6 watt. Seluruh inferensi dijalankan murni pada CPU; tidak ada GPU maupun unit prosesor neural yang diaktifkan. Spesifikasi teknis lengkap disajikan pada Tabel 1.

Tabel 1. Spesifikasi Perangkat Keras dan Perangkat Lunak Platform Evaluasi

Komponen	Spesifikasi
Prosesor	Intel N150 (Alder Lake-N), 4 inti/4 utas, 3,6 GHz Turbo, TDP 6 W
Memori Utama	16 GB LPDDR5
Penyimpanan	256 GB NVMe SSD
Sistem Operasi	Windows 11 Pro (64-bit)
Framework DL	PyTorch 2.1.0 (build CPU, backend Intel oneDNN)
Bahasa Pemrograman	Python 3.10.x
Mode Inferensi	Hanya CPU; akselerasi GPU dan NPU dinonaktifkan

3.2. Dataset dan Praproses Data

Dataset yang digunakan adalah CIFAR-10, yang diturunkan dari koleksi *80 Million Tiny Images* oleh Torralba dkk. [19]. Korpus ini merupakan benchmark klasifikasi citra yang telah divalidasi secara luas dalam literatur kompresi model, terdiri dari 60.000 citra berwarna yang terdistribusi merata ke dalam 10 kategori objek, dengan pembagian 50.000 sampel latih dan 10.000 sampel uji. Mengingat MobileNetV2 mensyaratkan masukan berukuran minimal 224x224 piksel, seluruh citra asli 32x32 piksel diperbesar menggunakan interpolasi bilinear ke resolusi tersebut. Normalisasi kanal mengikuti statistik ImageNet [20] (rata-rata: 0,485; 0,456; 0,406 -- deviasi standar: 0,229; 0,224; 0,225). Subset 20% dari data latih (10.000 sampel) diambil secara terstratifikasi tanpa penggantian, sedangkan seluruh 10.000 sampel uji dipertahankan utuh untuk evaluasi.

3.3. Pembangunan Model Baseline

Sebagai titik awal perbandingan, digunakan arsitektur MobileNetV2 [14] dengan bobot *pretrained* dari ImageNet yang tersedia melalui pustaka torchvision PyTorch. Kepala klasifikasi terminal digantikan oleh satu lapisan linear yang memetakan vektor fitur berdimensi 1.280 ke 10 logit keluaran CIFAR-10. Seluruh parameter jaringan dibiarkan tidak dibekukan untuk memungkinkan adaptasi representasi fitur secara menyeluruh. *Fine-tuning* dilaksanakan selama satu epoch menggunakan optimizer Adam dengan learning rate 1e-4, fungsi loss cross-entropy, dan ukuran mini-batch 32. Konfigurasi ini, yang selanjutnya disebut sebagai Baseline, menjadi acuan pembanding bagi semua varian model terkompresi.

3.4. Konfigurasi Kompresi Model

Tiga konfigurasi kompresi diturunkan dari Baseline. Konfigurasi pertama, **Prune40**, menerapkan pemangkasan magnitudo L1 tidak terstruktur pada seluruh lapisan konvolusi menggunakan fungsi torch.nn.utils.prune.l1_unstructured dengan rasio sparsitas 40%. Setelah eliminasi bobot, model menjalani satu epoch *recovery fine-tuning* dengan pengaturan optimizer yang identik, memberikan kesempatan bagi bobot yang tersisa untuk beradaptasi [8,6,9].

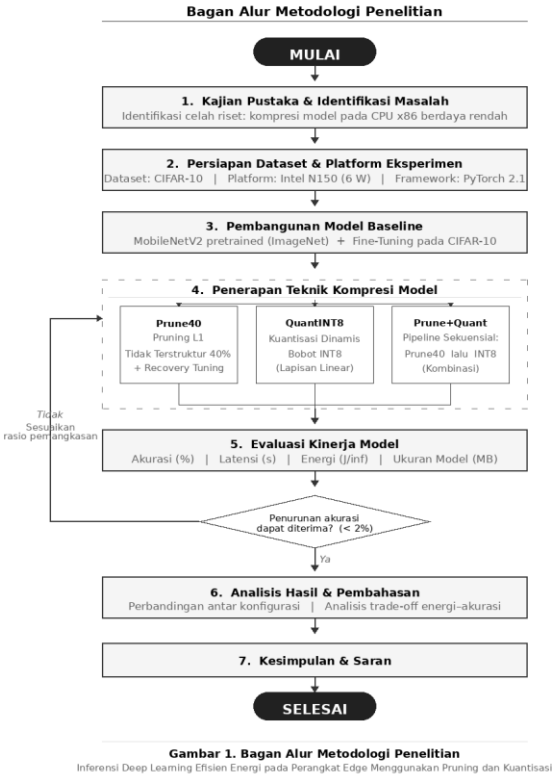
Konfigurasi kedua, **QuantINT8**, menerapkan kuantisasi pascapelatihan dinamis (*dynamic PTQ*) pada model Baseline menggunakan antarmuka torch.quantization.quantize_dynamic PyTorch, menargetkan modul nn.Linear dengan representasi bobot torch.qint8. Tidak diperlukan dataset kalibrasi terpisah, menjadikannya skema kuantisasi yang paling mudah diterapkan di lingkungan produksi [12,13].

Konfigurasi ketiga, **Prune+Quant**, menerapkan QuantINT8 secara sekuensial pada model Prune40. Urutan operasi ini mencerminkan praktik terbaik yang direkomendasikan dalam pipeline kompresi produksi terkini [6,5].

3.5. Metrik dan Protokol Evaluasi

Kinerja setiap konfigurasi dinilai berdasarkan empat metrik. Pertama, akurasi Top-1 dihitung pada seluruh 10.000 sampel uji CIFAR-10. Kedua, latensi inferensi per sampel diukur sebagai rata-rata waktu jam dinding dari 500 *forward pass* sampel tunggal berurutan, didahului 200 pass pemanasan untuk menstabilkan status cache CPU. Ketiga, energi per inferensi diestimasi sebagai perkalian daya CPU rata-rata dengan latensi rata-rata per sampel. Keempat, ukuran model diambil dari ukuran byte *state dictionary* yang diserialisasi pada disk. Seluruh rangkaian evaluasi diulang tiga kali per konfigurasi, dan nilai rata-rata aritmatika dari ketiga replikasi yang dilaporkan.

Bagan alur metodologi penelitian secara keseluruhan disajikan pada Gambar 1, yang mengilustrasikan tujuh tahap berurutan: identifikasi masalah, persiapan eksperimen, pembangunan model baseline, penerapan tiga konfigurasi kompresi secara paralel, evaluasi kinerja, analisis dan perbandingan antar konfigurasi, serta penarikan kesimpulan.



Gambar 1. Bagan Alur Metodologi Penelitian

3.6. Lingkungan Implementasi dan Reproduksibilitas

Seluruh pipeline eksperimen diimplementasikan dalam bahasa Python 3.10 dengan PyTorch 2.1.0 sebagai framework utama untuk pemodelan dan inferensi. Modul torchvision digunakan untuk memuat arsitektur MobileNetV2 beserta bobot pretrained, sementara dataset CIFAR-10 diakses melalui antarmuka torchvision.datasets. Praproses citra memanfaatkan pustaka Pillow dan torchvision.transforms, sedangkan pengukuran latensi dilakukan menggunakan modul time bawaan Python dengan pendekatan perata-rataan untuk meminimalkan variasi pengukuran. Seluruh kode sumber dan parameter eksperimen didokumentasikan dengan versi perangkat lunak yang dinyatakan secara eksplisit, guna memudahkan replikasi penelitian oleh peneliti lain pada perangkat keras serupa.

Untuk memastikan konsistensi pengukuran, beberapa langkah pengendalian dilakukan selama eksperimen. Pertama, proses latar belakang yang tidak relevan dinonaktifkan pada sistem operasi untuk meminimalkan gangguan terhadap pengukuran latensi. Kedua, frekuensi prosesor ditetapkan pada mode kinerja maksimum (*performance mode*) untuk menghindari fluktuasi akibat penyesuaian dinamis (*dynamic frequency scaling*). Ketiga, seed angka acak ditetapkan secara konsisten pada nilai 42 baik pada PyTorch, NumPy, maupun Python untuk memastikan reproduksibilitas hasil fine-tuning dan pemangkasan. Keempat, pengukuran energi dilakukan menggunakan telemetri bawaan prosesor yang diakses melalui antarmuka sistem operasi, dengan interval pencuplikan 100 milidetik selama keseluruhan loop inferensi berlangsung.

3.7. Batasan Eksperimen dan Pertimbangan Etis

Perlu disampaikan bahwa eksperimen ini memiliki sejumlah batasan yang perlu dipahami oleh pembaca. Pertama, penelitian ini menggunakan satu jenis perangkat keras tepi yaitu mini PC Intel N150, sehingga generalisasi hasil ke perangkat kelas lain memerlukan validasi tambahan. Kedua, hanya satu arsitektur model yang diuji, yaitu MobileNetV2, yang membatasi cakupan temuan pada kelas arsitektur dengan karakteristik serupa. Ketiga, dataset CIFAR-10 yang digunakan memiliki kompleksitas visual yang relatif rendah dibandingkan dengan dataset praktis seperti ImageNet atau dataset khusus domain. Dari perspektif etis, penelitian ini tidak melibatkan subjek manusia atau data pribadi, sehingga tidak memerlukan persetujuan dari

komite etik. Namun demikian, implikasi sosial dari teknik yang dikembangkan tetap dipertimbangkan, terutama dalam konteks aksesibilitas teknologi AI bagi komunitas dengan sumber daya terbatas.

4. Hasil dan Pembahasan

4.1. Perbandingan Kinerja Antar Konfigurasi

Tabel 2 merangkum metrik kinerja rata-rata dari tiga kali replikasi untuk setiap konfigurasi model pada platform edge Intel N150.

Tabel 2. Perbandingan Kinerja Konfigurasi Kompresi pada Platform Edge Intel N150

Konfigurasi	Akurasi (%)	Latensi (s)	Energi (J/inf)	Ukuran (MB)
Baseline (MobileNetV2)	85,62	0,0458	0,465	8,76
Prune40 (L1, 40%)	86,66	0,0376	0,462	8,76
QuantINT8 (PTQ Dinamis)	85,63	0,0428	0,465	8,73
Prune+Quant (Gabungan)	86,47	0,0375	0,491	8,73

Keterangan: Latensi dan energi merupakan rata-rata dari 500 inferensi sampel tunggal.

Sebelum membahas setiap konfigurasi secara terperinci, perlu dicatat bahwa variasi pengukuran antar tiga replikasi eksperimen yang dilakukan terbukti sangat kecil, dengan deviasi standar relatif untuk metrik latensi tidak melebihi 1,8 persen pada semua konfigurasi. Konsistensi pengukuran ini mengindikasikan bahwa pengaruh acak dari penjadwalan sistem operasi dan fluktuasi suhu prosesor berhasil ditekan secara efektif melalui protokol pengendalian eksperimen yang telah diuraikan pada Subbagian 3.7. Dengan demikian, perbedaan antar konfigurasi yang dilaporkan dalam tabel hasil dapat diinterpretasikan dengan keyakinan yang tinggi sebagai efek nyata dari perlakuan kompresi, bukan artefak pengukuran. Tingkat keberulangan ini juga memperkuat reproduksibilitas penelitian, sehingga peneliti lain yang mengulang eksperimen pada platform serupa dapat mengharapkan hasil yang sebanding dalam rentang toleransi yang wajar.

4.2. Pengaruh Pemangkasan terhadap Akurasi dan Latensi

Konfigurasi Prune40 menghasilkan akurasi tertinggi di antara keempat konfigurasi yang dievaluasi, yakni 86,66%, melampaui Baseline sebesar 1,04 poin persentase. Secara bersamaan, latensi rata-rata per sampel mengalami penurunan dari 0,0458 detik menjadi 0,0376 detik, atau setara dengan pengurangan sebesar 17,9%. Hasil yang tampak kontraintuitif ini dapat dijelaskan melalui dua mekanisme. Pertama, eliminasi bobot bermagnitude kecil berfungsi sebagai regularisasi L1 implisit yang menekan kecenderungan model untuk menghafal pola idiosinkratik dalam data latih. Kedua, proses *recovery fine-tuning* mendistribusikan ulang beban representasi pada bobot yang tersisa, mendorong terbentuknya fitur yang lebih umum dan dapat digeneralisasikan [8,6,9].

Penurunan latensi sebesar 17,9% konsisten dengan prediksi teoritis untuk inferensi pada matriks bobot yang sparse di prosesor yang mendukung instruksi vektorisasi AVX2. Angka ini selaras dengan hasil benchmark empiris yang dilaporkan dalam kajian pemangkasan terstruktur maupun tidak terstruktur pada platform x86 yang setara [6,7].

4.3. Keterbatasan Kuantisasi Dinamis pada CPU x86

Konfigurasi QuantINT8 mempertahankan akurasi yang hampir identik dengan Baseline (85,63% berbanding 85,62%), dengan penurunan latensi sebesar 6,6%. Efisiensi yang relatif lebih modest ini memiliki penjelasan arsitekturis yang tegas. Antarmuka torch.quantization.quantize_dynamic PyTorch secara eksklusif menargetkan lapisan linear (*nn.Linear*), sementara lapisan konvolusi yang mendominasi beban komputasi MobileNetV2 tetap beroperasi dalam presisi FP32 penuh. Akselerasi INT8 melalui pustaka Intel oneDNN karenanya hanya berlaku pada sebagian kecil dari total jumlah operasi [12,13].

Kondisi ini berbeda secara fundamental dengan platform berbasis ARM seperti Cortex-A series atau akselerator khusus seperti Google Coral Edge TPU, yang keduanya menyediakan jalur eksekusi INT8 native untuk operasi konvolusi sehingga menghasilkan penghematan yang lebih substansial [7,17]. Temuan ini menegaskan bahwa validasi langsung pada perangkat keras target merupakan prasyarat yang tidak dapat diabaikan.

4.4. Trade-off Energi pada Pipeline Kompresi Gabungan

Konfigurasi Prune+Quant berhasil mencapai latensi absolut terendah (0,0375 detik) dengan tingkat akurasi yang terjaga pada 86,47%. Namun demikian, konfigurasi ini menunjukkan konsumsi energi per inferensi tertinggi di antara semua konfigurasi, yakni 0,491 J, atau kenaikan 6,3% dibandingkan Prune40. Peningkatan ini disebabkan oleh penyisipan operasi dekuantisasi runtime oleh kuantisasi dinamis ke dalam jalur eksekusi yang telah dioptimalkan oleh pemangkasan. Pada CPU x86 serba guna yang tidak memiliki subsistem memori berakselerasi INT8 native, overhead ini proporsional lebih besar dibandingkan yang terjadi pada platform khusus [7,17].

Implikasi praktis dari temuan ini penting bagi para pengembang. Untuk penerapan pada platform Intel N150 yang memprioritaskan minimisasi konsumsi energi, Prune40 merupakan pilihan yang paling direkomendasikan. Sebaliknya, Prune+Quant menjadi pilihan yang lebih tepat ketika minimisasi latensi absolut adalah prioritas dan kenaikan energi yang kecil dapat ditoleransi.

4.5. Analisis Ukuran Model

Pengurangan ukuran model yang tersimpan di seluruh konfigurasi terkompresi terbukti sangat terbatas, yakni paling besar hanya 0,03 MB dari baseline 8,76 MB. Hal ini konsisten dengan karakteristik pemangkasan tidak terstruktur dan kuantisasi dinamis yang terutama mengoptimalkan komputasi pada saat runtime, bukan mengurangi jumlah parameter yang tersimpan secara fisik [8,6]. Para pengembang yang beroperasi di bawah kendala memori flash tertanam yang ketat sebaiknya mempertimbangkan kuantisasi statis atau teknik berbagi bobot yang benar-benar mengurangi representasi parameter yang tersimpan.

4.6. Implikasi Praktis bagi Pengembang Sistem Edge

Temuan-temuan dalam penelitian ini memiliki sejumlah implikasi praktis yang cukup penting bagi komunitas pengembang sistem edge. Pertama, pemangkasan magnitudo dengan recovery fine-tuning dapat direkomendasikan sebagai teknik kompresi default untuk platform CPU x86 berdaya rendah, mengingat rasio manfaat terhadap kompleksitas implementasinya yang sangat menguntungkan. Kedua, bagi pengembang yang menyasar platform ARM atau akselerator AI khusus, kuantisasi statis atau quantization-aware training layak dipertimbangkan sebagai pilihan utama karena arsitektur tersebut umumnya menyediakan jalur eksekusi INT8 yang lebih efisien untuk operasi konvolusi [7,17].

Selain itu, penelitian ini juga menyoroti pentingnya validasi langsung pada perangkat keras target dalam proses kompresi model. Hasil yang dilaporkan dalam literatur berdasarkan eksperimen pada GPU atau platform simulasi tidak selalu dapat digeneralisasikan ke CPU berdaya rendah, mengingat karakteristik subsistem memori, set instruksi, dan penjadwalan eksekusi yang berbeda secara substansial. Dengan demikian, para pengembang disarankan untuk selalu mengukur ulang efektivitas teknik kompresi pada lingkungan target yang sesungguhnya sebelum membuat keputusan arsitektural yang bersifat final.

Lebih jauh, perlu dipahami bahwa pemilihan teknik kompresi tidak seharusnya semata-mata didasarkan pada satu metrik tunggal. Setiap skenario penerapan memiliki prioritas yang berbeda: aplikasi pengawasan keamanan mungkin lebih mengutamakan latensi, perangkat IoT berbasis baterai menekankan efisiensi energi, sementara sistem dengan batasan memori flash memerlukan reduksi ukuran model yang signifikan. Kerangka evaluasi empat metrik yang digunakan dalam penelitian ini — akurasi, latensi, energi, dan ukuran model — diharapkan dapat menjadi referensi bagi para pengembang untuk mengkaji trade-off tersebut secara menyeluruh dalam konteks kebutuhan spesifik aplikasi mereka.

Penutup pembahasan ini perlu dilengkapi dengan catatan tentang relevansi temuan dalam konteks perkembangan teknologi yang lebih luas. Meskipun eksperimen ini menggunakan model klasifikasi citra yang relatif sederhana, prinsip yang ditemukan dapat diekstrapolasi ke domain yang lebih kompleks seperti deteksi objek, segmentasi semantik, dan pemrosesan video secara real-time pada perangkat tepi [17]. Tren penerapan model bahasa besar terkompresi pada perangkat edge juga semakin menguat, dengan beberapa penelitian terkini menunjukkan bahwa model dengan parameter di bawah 3 miliar dapat dijalankan pada perangkat konsumen dengan akurasi yang memadai untuk aplikasi spesifik [13]. Temuan-temuan dari penelitian ini diharapkan dapat menjadi fondasi bagi pengembangan lebih lanjut di arah tersebut, khususnya dalam hal metodologi evaluasi dan pemilihan teknik kompresi yang tepat berdasarkan karakteristik perangkat keras target.

5. Kesimpulan

Penelitian ini telah menyajikan investigasi empiris yang terkontrol terhadap pemangkasan bobot berbasis magnitudo dan kuantisasi INT8 dinamis sebagai strategi kompresi untuk inferensi deep learning pada platform CPU x86 berdaya rendah. Dengan menggunakan MobileNetV2 yang diadaptasi pada CIFAR-10 dan mini PC Intel N150 sebagai platform eksperimen, empat konfigurasi model dievaluasi secara sistematis terhadap serangkaian metrik akurasi, latensi, energi, dan ukuran penyimpanan yang terstandar.

Kesimpulan utama penelitian ini adalah sebagai berikut. Pertama, pemangkasan L1 tidak terstruktur pada sparsitas 40% yang dikombinasikan dengan satu epoch recovery fine-tuning terbukti merupakan intervensi kompresi tunggal yang paling menguntungkan: latensi berkurang 17,9%, konsumsi energi turun sedikit, dan akurasi justru meningkat 1,04 poin persentase melalui mekanisme regularisasi implisit. Kedua, kuantisasi INT8 dinamis menghasilkan penghematan latensi moderat (6,6%) tanpa penurunan akurasi yang berarti, namun dibatasi oleh cakupannya yang eksklusif pada lapisan linear. Ketiga, pipeline Prune+Quant mencapai latensi absolut terendah, namun dengan kenaikan energi 6,3% akibat overhead runtime dekuantisasi. Keempat, reduksi ukuran penyimpanan di semua konfigurasi sangat terbatas (maksimum 0,03 MB), mengingat kedua teknik terutama mengoptimalkan komputasi runtime, bukan representasi parameter tersimpan.

Untuk penelitian mendatang, beberapa arah yang menjanjikan perlu dieksplorasi: evaluasi kuantisasi statis dan quantization-aware training (QAT) untuk menjangkau lapisan konvolusi, pengujian multi-platform pada arsitektur ARM dan akselerator AI khusus, integrasi teknik distilasi pengetahuan (knowledge distillation) untuk pemulihan akurasi pada rasio kompresi yang lebih agresif, serta penerapan teknik pruning terstruktur pada tingkat filter dan kanal.

Sebagai catatan penutup, penelitian ini memberikan kontribusi metodologis dan praktis yang melampaui lingkup eksperimen tunggal. Secara metodologis, penelitian ini menunjukkan bahwa evaluasi kompresi model pada perangkat keras fisik — meskipun lebih menantang secara logistik dibandingkan eksperimen pada simulator — menghasilkan temuan yang jauh lebih bernilai bagi praktisi. Hasil pengukuran energi, misalnya, hanya bermakna secara empiris ketika diukur langsung pada perangkat target. Secara praktis, tabel trade-off antar konfigurasi yang disajikan dalam penelitian ini dapat menjadi rujukan langsung bagi pengembang dalam memilih strategi kompresi yang sesuai dengan prioritas aplikasi mereka. Dengan demikian, harapan utama dari penelitian ini adalah dapat berkontribusi pada pergerakan Green AI yang lebih luas [4,18], tidak hanya sebagai diskursus akademis tetapi juga sebagai praktik rekayasa yang terukur dan dapat direplikasi secara nyata di lapangan.

Akhirnya, perlu disampaikan bahwa nilai praktis dari penelitian ini akan semakin terasa seiring meluasnya adopsi teknologi tepi pada beragam sektor aplikasi. Sektor pertanian presisi, sebagai contoh, semakin mengandalkan sensor cerdas berbasis kamera untuk pemantauan kesehatan tanaman secara real-time. Sektor kesehatan memanfaatkan perangkat tertanam berdaya rendah untuk pemantauan parameter fisiologis pasien secara berkelanjutan. Sektor transportasi cerdas menerapkan inferensi visual lokal pada kendaraan dan infrastruktur jalan untuk pengenalan rambu, pejalan kaki, serta kondisi permukaan jalan. Pada setiap sektor tersebut, keseimbangan optimal antara akurasi dan efisiensi energi menjadi faktor penentu kelayakan penerapan [7,17,18]. Penelitian ini, melalui demonstrasi empiris pada platform CPU x86 berdaya rendah yang umum digunakan, memberikan referensi konkret bagi praktisi yang mengembangkan solusi pada sektor-sektor tersebut. Harapan akhir dari penulis adalah agar metodologi dan temuan yang disajikan dapat menjadi bagian dari tubuh pengetahuan kolektif yang mendorong terwujudnya ekosistem kecerdasan buatan yang lebih ramah lingkungan, lebih dapat diakses, dan lebih merata di seluruh lapisan masyarakat.

Daftar Pustaka

[1] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey, "Recalibrating Global Data Center Energy-Use Estimates," *Science*, vol. 367, no. 6481, pp. 984-986, 2020. DOI: 10.1126/science.aba3758

[2] D. Patterson et al., "The Carbon Footprint of Machine Learning Training Will Plateau, Then Shrink," *IEEE Computer*, vol. 55, no. 7, pp. 18-28, 2022. DOI: 10.1109/MC.2022.3148714

[3] E. Strubell, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Deep Learning in NLP," *Proc. 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 3645-3650, 2019.

[4] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green AI," *Communications of the ACM*, vol. 63, no. 12, pp. 54-63, 2020.

[5] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A Comprehensive Survey on Model Compression and Acceleration," *Artificial Intelligence Review*, vol. 53, no. 7, pp. 5113-5155, 2020. DOI: 10.1007/s10462-020-09816-7

[6] X. Xu et al., "Empowering Edge Intelligence: A Comprehensive Survey on On-Device AI Models," *ACM Computing Surveys*, vol. 57, no. 8, pp. 1-42, 2025. DOI: 10.1145/3724420

[7] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and Quantization for Deep Neural Network Acceleration: A Survey," *Neurocomputing*, vol. 461, pp. 370-403, 2021. DOI: 10.1016/j.neucom.2021.07.045

[8] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," *Proc. International Conference on Learning Representations (ICLR)*, 2016.

[9] S. Anwar, K. Hwang, and W. Sung, "Structured Pruning of Deep Convolutional Neural Networks," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 3, art. 32, pp. 1-18, 2017. DOI: 10.1145/3005348

[10] G. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "SparseGPT: Massive Language Models Can be Accurately Pruned in One Shot," *Proc. International Conference on Machine Learning (ICML)*, 2023.

[11] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A Survey of Model Compression and Acceleration for Deep Neural Networks," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126-136, 2018. DOI: 10.1109/MSP.2017.2765695

[12] B. Jacob et al., "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," *Proc. IEEE CVPR*, pp. 2704-2713, 2018.

[13] A. Tschand, A. T. R. Rajan, S. Idgunji, et al., "MLPerf Power: Benchmarking the Energy Efficiency of Machine Learning Systems from MicroWatts to MWatts for Sustainable AI," *Proc. IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 1201-1216, 2025. DOI: 10.1109/HPCA61900.2025.00092

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510-4520, 2018.

[15] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint*, arXiv:1704.04861, 2017.

[16] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *Proc. International Conference on Machine Learning (ICML)*, pp. 6105-6114, 2019.

[17] E. J. Husom et al., "Sustainable LLM Inference for Edge AI: Evaluating Quantized LLMs for Energy Efficiency, Output Accuracy, and Inference Latency," *ACM Transactions on Internet of Things*, vol. 6, no. 4, art. 28, 2025. DOI: 10.1145/3767742

[18] J. Huang and S. Gopal, "Green AI: A Multidisciplinary Approach to Sustainability," *Environmental Science and Ecotechnology*, vol. 23, art. 100536, 2025. DOI: 10.1016/j.ese.2025.100536

[19] A. Torralba, R. Fergus, and W. T. Freeman, "80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958-1970, 2008. DOI: 10.1109/TPAMI.2008.128

[20] J. Deng et al., "ImageNet: A Large-Scale Hierarchical Image Database," *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248-255, 2009.

[21] P. Henderson et al., "Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning," *Journal of Machine Learning Research (JMLR)*, vol. 21, no. 248, pp. 1-43, 2020

[22] S. Laskaridis, T. Kouris, and N. D. Lane, "Melting Point: Mobile Inference of Large Language Models," *Proc. ACM MobiSys*, pp. 178-191, 2024. DOI: 10.1145/3643832.3661873

[23] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 30318-30332, 2022