BAB II
KAJIAN PUSTAKA

2.1 Kompresi Gambar dengan CNN

Kompresi gambar adalah proses penting dalam media digital, yang bertujuan mengurangi jumlah data yang diperlukan untuk mewakili gambar diiringi dengan upaya mempertahankan kualitas visual yang dapat diterima (Ji & Karam, 2023). Proses ini menjadi semakin masif dalam era komunikasi visual modern, di mana volume data visual yang ditransmisikan dan disimpan meningkat signifikan untuk berbagai kebutuhan. Kompresi dicapai dengan menghilangkan redundansi dalam data gambar, yang dapat berupa lossy atau lossless, tergantung pada penghapusan permanen elemen data tertentu. Redundansi tersebut bisa bersifat spasial, statistik, maupun perseptual, dan pemanfaatannya memungkinkan representasi citra yang lebih ringkas tanpa mengorbankan informasi penting secara signifikan.

Tujuan utama kompresi gambar adalah untuk mengurangi biaya penyimpanan dan transmisi, pertimbangan yang semakin signifikan mengingat pertumbuhan secara eksponen dalam lalu lintas media visual digital (Rekha & S, 2022). Dalam konteks jaringan dengan bandwidth terbatas seperti LoRa, efisiensi kompresi menjadi salah satu faktor dalam memastikan keterkiriman data visual tanpa mengganggu stabilitas sistem. Seiring dengan perkembangan teknologi pembelajaran mesin, kompresi gambar berbasis pembelajaran telah muncul sebagai alternatif, memanfaatkan metodologi pembelajaran mendalam untuk mencapai rasio kompresi sambil mempertahankan atau meningkatkan kualitas visual. Pendekatan ini secara fundamental berbeda dari metode klasik karena melibatkan proses pelatihan jaringan saraf untuk mengenali pola representatif dalam gambar dan menyusun encoding-nya dalam bentuk *latent space* yang padat, serta dapat diterjemahkan kembali untuk merekonstruksi gambar.

Dari hal tersebut, metode ini juga membuka potensi untuk integrasi dengan sistem pemrosesan lanjutan. Metode ini tidak hanya disesuaikan dengan persepsi manusia tetapi juga dioptimalkan untuk tugas mesin, memungkinkan pemrosesan

langsung dalam domain terkompresi, yang sangat relevan untuk aplikasi seperti pengenalan berbasis gambar (Gupta dkk., 2024).

2.2 *Encoder-Decoder* Berbasis CNN

CNN dapat digunakan menjadi *encoder-decoder* untuk kompresi gambar yang memanfaatkan konvolusi jaringan saraf untuk mencapai kompresi dan rekonstruksi gambar yang memadai. Prinsip dasar dari metode ini melibatkan dua komponen utama, yaitu: *encoder*, yang memampatkan gambar input menjadi representasi kompak, dan *decoder*, yang merekonstruksi gambar dari representasi ini. Arsitektur ini (*Encoder-decoder* berbasis CNN) sangat menguntungkan untuk memproses volume besar gambar secara paralel, sehingga cocok untuk lingkungan dengan sumber daya komputasi yang besar, seperti pusat data (Prabhu dkk., 2021). Pendekatan ini juga dapat diadaptasi untuk sistem dengan sumber daya terbatas, terutama bila desain arsitektur dan ukuran *latent space*-nya dikendalikan secara efisien. Peran ruang laten, sering disebut sebagai hambatan, sangat penting dalam arsitektur ini. Ini berfungsi sebagai representasi terkompresi dari gambar input, menangkap informasi struktural penting sambil membuang data yang berlebihan. Representasi ringkas ini sangat penting untuk menjaga kualitas gambar yang direkonstruksi, karena memastikan bahwa hanya fitur yang paling signifikan yang dipertahankan. Efektivitas *bottleneck* dapat ditingkatkan dengan meminimalkan redundansi melalui teknik seperti menghukum redundansi fitur, yang mengarah pada representasi yang lebih kaya dan lebih beragam (Laakom dkk., 2022). Proses *encoding* bukan hanya sekadar pemampatan, tetapi juga penyaringan terhadap informasi yang paling relevan bagi proses *decoding*. Penggunaan CNN dalam konteks ini tidak hanya meningkatkan rasio kompresi tetapi juga mengupayakan rekonstruksi tetap memadai, menjadikannya solusi yang cukup berpeluang untuk tugas penglihatan di berbagai tingkatan (Abhiram & Khetavath, 2025). Hal ini menunjukkan bahwa CNN sebagai *encoder-decoder* dapat dikembangkan lebih lanjut untuk menjawab tantangan kompresi pada sistem transmisi data dengan keterbatasan kapasitas, selama rancangan arsitektur dan representasi *latent*-nya dioptimalkan secara tepat.

Arsitektur CNN dibangun dari serangkaian komponen inti yang secara bersama-sama membentuk proses transformasi data masukan menjadi representasi fitur yang semakin kompleks. Salah satu elemen utama dari CNN adalah dimensi *input* dan *output* dari setiap lapisan, yang umumnya dinyatakan dalam format tiga dimensi: tinggi (*height*), lebar (*width*), dan jumlah saluran (*channels*). Ukuran dimensi ini secara bertahap diperkecil melalui proses downsampling untuk menghasilkan representasi *latent* yang padat. Setiap lapisan konvolusi dalam CNN menerapkan operasi menggunakan kernel atau filter, yaitu matriks kecil yang digeser melintasi citra untuk mengekstraksi fitur lokal. Jumlah kernel menentukan jumlah saluran keluaran dan menjadi salah satu penentu kompleksitas model. Setelah konvolusi, biasanya diterapkan fungsi aktivasi nonlinier seperti ReLU (*Rectified Linear Unit*) untuk memperkenalkan kemampuan representasi nonlinier terhadap fitur yang diekstraksi.

Pengaturan *stride* dan *padding* juga memengaruhi hasil konvolusi. *Stride* adalah langkah pergeseran kernel saat melintasi input; semakin besar nilai *stride*, semakin kecil dimensi *output*. *Padding* digunakan untuk mengontrol dimensi output dengan menambahkan piksel di tepi input, biasanya untuk mempertahankan ukuran spasial. Selain konvolusi, beberapa arsitektur CNN menyertakan operasi *pooling*, seperti *max pooling*, yang bertujuan mereduksi dimensi spasial sambil mempertahankan fitur dominan. Di sisi *decoder*, digunakan *transposed convolution* atau metode upsampling lain untuk mengembalikan dimensi output ke ukuran semula. Jumlah lapisan konvolusi, urutan, dan kedalaman fitur memiliki pengaruh langsung terhadap ukuran dan struktur *latent space*. Semakin dalam dan kompleks struktur encoder, semakin tinggi kapasitas representasi *latent* yang dapat dihasilkan, namun dengan konsekuensi peningkatan ukuran data dan beban komputasi.

Selama proses pelatihan CNN, terdapat sejumlah parameter yang tidak dipelajari secara langsung oleh model, namun sangat berpengaruh terhadap hasil akhir. Parameter ini disebut *hyperparameters* dan harus ditentukan sebelum proses pelatihan dimulai. Salah satu yang utama adalah jumlah *epoch*, yaitu berapa kali seluruh dataset dilalui oleh model selama pelatihan. Semakin banyak *epoch*, semakin besar peluang model belajar pola dari data, namun juga meningkatkan

risiko *overfitting* jika tidak disertai dengan regulasi yang memadai. Parameter lainnya adalah *batch size*, yaitu jumlah sampel yang diproses dalam satu iterasi pembaruan bobot. Pemilihan *batch size* memengaruhi kestabilan dan kecepatan pelatihan; ukuran yang kecil cenderung menghasilkan gradien yang bervariasi, sedangkan ukuran besar membutuhkan sumber daya memori yang lebih tinggi.

Parameter penting lainnya adalah *learning rate*, yang mengatur besarnya langkah pembaruan bobot pada setiap iterasi. *Learning rate* yang terlalu besar dapat menyebabkan model gagal konvergen, sedangkan nilai yang terlalu kecil memperlambat proses pelatihan. Untuk mengoptimalkan proses pelatihan, digunakan juga *optimizer*, yaitu algoritma yang menentukan bagaimana bobot diperbarui berdasarkan gradien. Optimizer seperti Adam dan Stochastic Gradient Descent (SGD) merupakan pilihan umum karena efisiensi dan kestabilannya dalam berbagai kondisi pelatihan. Dalam beberapa kasus, digunakan juga teknik regulasi seperti *dropout* untuk mencegah *overfitting* dengan cara mengabaikan sebagian unit selama pelatihan, atau *weight decay* untuk mengendalikan kompleksitas model dengan penalti pada bobot besar. Gabungan pengaturan *hyperparameter* ini dapat memengaruhi secara langsung proses konvergensi dan generalisasi model. Oleh karena itu, pemilihan nilai-nilai tersebut dalam konteks penelitian ini dilakukan secara hati-hati agar sesuai dengan kapasitas komputasi dan karakteristik dataset yang digunakan.

2.3 Loss Function

Fungsi *loss* merupakan komponen kunci dalam proses pelatihan model pembelajaran mendalam. Dalam tugas rekonstruksi citra, fungsi *loss* berfungsi sebagai indikator kesalahan antara citra asli dan citra hasil rekonstruksi, serta mengarahkan proses pembaruan bobot model selama pelatihan. Pemilihan fungsi *loss* memengaruhi arah optimisasi model, kualitas visual yang dihasilkan, dan kestabilan pelatihan. Berikut ini adalah beberapa fungsi *loss* yang digunakan dalam penelitian ini.

2.3.1 MSE

MSE adalah fungsi *loss* yang paling umum digunakan dalam tugas regresi dan rekonstruksi citra. MSE mengukur rata-rata kuadrat selisih nilai piksel antara gambar asli dan gambar hasil rekonstruksi (Sharma dkk., 2016). MSE memberikan penalti lebih besar terhadap kesalahan besar, yang membantu model mengurangi kesalahan signifikan pada piksel tertentu. Namun, metrik ini sering menghasilkan gambar yang terlihat kabur secara visual, karena tidak mempertimbangkan struktur spasial atau persepsi manusia terhadap kualitas gambar. Meskipun demikian, kesederhanaannya dan kemampuannya memberikan gradien yang stabil menjadikan MSE sebagai titik awal yang umum dalam pelatihan model kompresi (Sara dkk., 2019).

2.3.2 MAE

MAE menghitung rata-rata dari nilai absolut selisih antar piksel. MAE lebih robust terhadap outlier dibanding MSE, karena tidak memberikan penalti kuadrat terhadap kesalahan besar (Robeson & Willmott, 2023). Namun, gradien dari MAE bersifat konstan, sehingga dapat membuat proses optimisasi lebih lambat dalam beberapa kasus. MAE cenderung menghasilkan gambar dengan kesalahan yang merata, namun kadang kurang tajam secara visual. Fungsi ini tetap menjadi pilihan menarik ketika stabilitas pelatihan lebih diutamakan daripada ketepatan ekstrem terhadap piksel tertentu.

2.3.3 SSIM

SSIM adalah metrik berbasis persepsi yang menilai kualitas gambar dengan mempertimbangkan kesamaan struktur, luminansi, dan kontras antara dua gambar. SSIM lebih sejalan dengan persepsi visual manusia (Hwang dkk., 2020), sehingga digunakan baik sebagai fungsi *loss*. Dalam konteks pelatihan model, SSIM dapat digunakan sebagai fungsi *loss* dengan memaksimalkan skor kesamaan struktural. Karena sifatnya yang tidak sepenuhnya diferensiabel, penerapan SSIM sering kali dikombinasikan atau disesuaikan agar tetap dapat digunakan dalam proses

backpropagation. Penggunaan SSIM sebagai fungsi *loss* membantu mempertahankan struktur spasial dan menghasilkan gambar yang secara visual lebih alami

2.3.4 Gradient + MSE

Kombinasi antara Gradient Loss dan MSE bertujuan untuk mempertahankan baik kesamaan nilai piksel maupun ketajaman kontur pada gambar. Gradient Loss bekerja dengan menghitung perbedaan gradien (derivatif spasial) antara gambar asli dan hasil rekonstruksi (Shi dkk., 2024). Pendekatan ini berguna dalam mengurangi dampak dari kelemahan MSE yang cenderung mengaburkan detail. Dengan menambahkan komponen *gradient*, model didorong untuk lebih peka terhadap perbedaan lokal yang tajam. Kombinasi dua fungsi ini biasanya dirancang dalam rasio tertentu, misalnya 7:3 antara MSE dan Gradient Loss, guna menyeimbangkan sensitivitas numerik dan persepsi visual.

2.3.5 Charbonnier

Charbonnier Loss menggabungkan keunggulan MAE dalam ketahanan terhadap outlier dengan gradien yang lebih stabil, menjadikannya pilihan dalam tugas pemulihan citra dan rekonstruksi. Fungsi ini cenderung menghasilkan gambar yang lebih bersih dan halus tanpa kehilangan terlalu banyak detail, dan telah digunakan secara luas dalam berbagai model super-resolution (Xu dkk., 2022).

2.4 Kompresi Tambahan *Latent Space*

2.4.1 Blosc

Blosc adalah pustaka kompresi data lossless yang dibuat khusus untuk meningkatkan kecepatan akses memori. Blosc menggabungkan teknik *blocking*, *suffling*, dan kompresi sehingga dapat dioptimalkan untuk kecepatan (Alted, 2010). Teknik *Blocking* ini dapat mengurangi lonjakan aktivitas pada memory bus sehingga pemindahan data dapat dipercepat. Blosc menggunakan algoritma BloscLZ yang merupakan turunan algoritma kompresi cepat untuk data biner (Bui dkk., 2014).

2.4.2 Zlib

Zlib, adalah *library* kompresi data bebas dan tidak dipatenkan yang digunakan untuk tujuan yang lebih umum. Zlib dapat digunakan secara luas pada hampir semua perangkat keras dengan berbagai sistem operasi. Zlib menggunakan metode *deflate* yang berasal dari PKZIP 2.x yang sejalan dengan metode yang digunakan pada *gzib* dan *Zip* (Gailly & Adler, 2004). Format data zlib bersifat fleksibel dan bisa lintas platform, dengan pengaturan jejak memori yang independen dari data input.

2.5 Kanal LoRa

Teknologi LoRa, memiliki karakter konsumsi daya yang rendah dan kemampuan jarak jauh, dibatasi oleh kecepatan data maksimum 37,5 Kbps (Sun dkk., 2017), yang secara signifikan berdampak pada ukuran data dan waktu pengiriman. Kecepatan data di LoRa dipengaruhi oleh parameter seperti Spreading Factor (SF), Bandwidth (BW), dan Code Rate (CR), dengan SF yang lebih tinggi menyebabkan kecepatan data yang lebih rendah tetapi jangkauan meningkat, dan sebaliknya (More & Patel, 2023). Kecepatan data LoRa yang rendah yang melekat menghasilkan durasi paket yang lebih lama, yang dapat meningkatkan kemungkinan tabrakan paket, terutama di lingkungan jaringan padat, sehingga mempengaruhi throughput jaringan secara keseluruhan. Batasan mendasar dari kecepatan data LoRa mengharuskan pengelolaan ukuran data dan waktu pengiriman yang cermat, terutama dalam skenario yang membutuhkan keandalan tinggi dan latensi rendah. Pertukaran antara kecepatan data dan cakupan harus seimbang untuk mengoptimalkan kinerja jaringan, karena kecepatan data yang lebih tinggi mengurangi jangkauan dan sebaliknya. Kecepatan data LoRa yang rendah menimbulkan tantangan untuk ukuran data yang besar dan pengiriman cepat, penelitian berkelanjutan dan kemajuan teknologi terus meningkatkan kemampuan dan potensi aplikasinya di jaringan IoT.