Department of Digital Business
Journal of Artificial Intelligence and Digital Business (RIGGS)
Homepage: https://journal.ilmudata.co.id/index.php/RIGGS
Vol. 4 No. 2 (2025) pp: 5163-5169
P-ISSN: 2963-9298, e-ISSN: 2963-914X

Analisis Peforma Algoritma Kompresi Huffman, LZW, BZIP2, Zstandard dan Brotli dalam Efisiensi Penyimpanan Data dan Transfer Data

Okta Irawati[1], Nurhasanah[2], Siti Fatul Zahrani[3], Ayu Talenta Lumbantoruan[4]
[1 2 3 4] Fakultas Ilmu Komputer, Program Studi Teknik Informatika, Universitas Pamulang, Tangerang Selatan, Indonesia
*dosen02610@unpam.ac.id, dosen02834@unpam.ac.id, sitifatulzahrani@gmail.com, ayutalentalumbantoruan@gmail.com

Abstrak

Penelitian ini bertujuan untuk mengevaluasi performa beberapa algoritma kompresi data dalam kaitannya dengan efisiensi penyimpanan dan kecepatan transfer data. Fokus utama terletak pada perbandingan antara efisiensi hasil kompresi, durasi proses kompresi-dekompresi, serta dampaknya terhadap waktu pengiriman data. Untuk mendukung analisis, digunakan teori dasar dari teknik kompresi seperti algoritma JPEG, Discrete Cosine Transform (DCT), dan Huffman Coding yang sering digunakan dalam pengolahan gambar digital.Data diperoleh melalui uji coba terhadap berbagai jenis file gambar digital (JPG, TIFF, GIF), kemudian dianalisis secara kualitatif dengan meninjau rasio kompresi, durasi proses, serta pengaruh karakteristik data terhadap efektivitas kompresi. Hasil analisis menunjukkan bahwa pemilihan algoritma yang sesuai (contohnya, adaptasi JPEG pada file non-JPG) dapat meningkatkan efisiensi ruang simpan hingga 40% dan mempercepat pengiriman data. Namun, efektivitas setiap algoritma sangat bergantung pada tingkat redundansi dan jenis format file yang digunakan.

Kata kunci: Algoritma Kompresi Data, Efisiensi Kompresi, Kecepatan Kompresi, Transfer Data, Format File

1. Latar Belakang

Teknik kompresi data merupakan salah satu elemen penting dalam dunia teknologi informasi yang berfungsi untuk mengecilkan ukuran data agar penyimpanan dan pengirimannya menjadi lebih efisien. Di tengah perkembangan teknologi yang semakin cepat dan volume data yang kian membesar, efisiensi dalam mengelola data menjadi sebuah kebutuhan yang tidak bisa diabaikan. Dengan menggunakan kompresi, data dapat disimpan dalam ukuran yang lebih kecil tanpa kehilangan esensi informasi yang penting. Selain itu, pengiriman data melalui jaringan juga menjadi lebih cepat karena ukuran file yang telah dikurangi. Hal ini tentu berdampak positif pada performa sistem secara keseluruhan.

Terdapat dua pendekatan umum dalam kompresi data, yaitu kompresi tanpa kehilangan (*lossless*) dan kompresi dengan kehilangan (*lossy*). Kompresi lossless mempertahankan seluruh informasi asli sehingga saat didekompresi, data akan kembali persis seperti semula, sangat cocok digunakan untuk dokumen penting seperti teks atau kode. Sebaliknya, kompresi lossy menghapus sebagian data yang dianggap kurang signifikan agar ukuran file dapat dikurangi secara drastis. Biasanya, metode lossy digunakan dalam file multimedia seperti gambar dan video yang toleran terhadap sedikit kehilangan kualitas. Pemilihan jenis kompresi harus disesuaikan dengan kebutuhan dan jenis data yang akan diproses.

Berbagai algoritma telah dikembangkan untuk mengimplementasikan teknik kompresi, masing-masing dengan kelebihan dan kekurangannya. Di antaranya terdapat algoritma Huffman Coding, LZW (Lempel-Ziv-Welch), Golomb Coding, serta Elias Gamma Coding yang semuanya termasuk dalam kompresi lossless. Huffman dan LZW populer digunakan di berbagai sistem karena kemampuannya dalam menghasilkan rasio kompresi yang baik. Sementara itu, algoritma seperti Golomb dan Elias Gamma sering dipakai untuk data dengan distribusi tertentu,

misalnya pada sistem pengkodean numerik. Dengan mengenal karakter setiap algoritma, pengguna bisa menentukan metode mana yang paling cocok untuk digunakan.

Pemilihan algoritma kompresi tidak bisa dilakukan secara sembarangan, karena harus mempertimbangkan berbagai aspek teknis dan kebutuhan sistem. Beberapa aplikasi mungkin lebih mengutamakan kecepatan proses kompresi, sementara yang lain lebih fokus pada hasil ukuran file yang sekecil mungkin. Di samping itu, faktor seperti kompleksitas perhitungan dan kesesuaian terhadap perangkat keras juga memengaruhi keputusan. Pada perangkat dengan kapasitas terbatas, algoritma yang ringan dan efisien lebih disukai agar tidak membebani sistem. Oleh karena itu, penting untuk melakukan pengujian dan evaluasi terhadap algoritma yang tersedia sebelum digunakan.

Penelitian ini diarahkan untuk menganalisis bagaimana performa dari beberapa algoritma kompresi dalam menyimpan dan mengirim data. Evaluasi dilakukan dengan mengamati efisiensi ukuran file setelah dikompresi serta kecepatan kompresi dan dekompresi. Dengan membandingkan performa tiap algoritma, penelitian ini berupaya memberikan informasi yang akurat mengenai kelebihan dan kekurangan masing-masing. Proses pengujian melibatkan perhitungan rasio kompresi, waktu eksekusi, serta kompleksitas implementasinya. Dari hasil analisis tersebut, diharapkan dapat diperoleh gambaran yang jelas tentang algoritma yang paling optimal dalam kondisi tertentu.

Seiring meningkatnya penggunaan teknologi berbasis cloud dan big data, pengelolaan data secara efisien menjadi hal yang sangat penting. Kompresi data memungkinkan sistem untuk bekerja lebih ringan, baik dalam penyimpanan lokal maupun dalam transfer data melalui jaringan. Baik perusahaan, instansi pemerintah, maupun individu kini sangat mengandalkan teknologi ini untuk menjaga sistem tetap optimal. Maka dari itu, pengembangan metode kompresi yang dapat beradaptasi dengan kebutuhan zaman menjadi langkah penting. Penelitian ini diharapkan mampu memberi kontribusi dalam upaya pengembangan teknologi yang mendukung efisiensi tersebut.

Lebih jauh lagi, efisiensi dari penggunaan algoritma kompresi juga berdampak pada aspek biaya dan lingkungan. Dengan mengurangi kebutuhan penyimpanan fisik dan beban pada jaringan, institusi bisa menekan pengeluaran untuk infrastruktur dan operasional. Selain itu, efisiensi energi juga dapat tercapai, karena data yang lebih ringan berarti konsumsi daya yang lebih rendah. Dari sudut pandang ekonomi dan lingkungan, penerapan kompresi yang tepat dapat membawa banyak manfaat jangka panjang. Oleh karena itu, penting untuk memilih algoritma yang tidak hanya unggul secara teknis tetapi juga efektif secara strategis.

Atas dasar pertimbangan tersebut, penelitian ini berusaha untuk mengkaji dan membandingkan empat algoritma utama, yakni Huffman, LZW, Golomb, dan Elias Gamma. Pemilihan algoritma ini didasarkan pada keberagaman karakteristik dan keunggulan masing-masing. Analisis terhadap performa kompresi dan dekompresi dilakukan agar dapat diketahui algoritma mana yang paling sesuai untuk diterapkan dalam berbagai kondisi nyata. Harapannya, hasil dari penelitian ini bisa menjadi acuan dalam pemilihan metode kompresi untuk akademisi maupun praktisi. Penelitian ini juga dapat membuka peluang baru dalam pengembangan teknik kompresi yang lebih efisien di masa depan.

2. Metode Penelitian

Penelitian ini dilakukan dengan metode penelitian pustaka dengan pendekatan deskriptif kualitatif. Ini berarti bahwa analisis dilakukan melalui tinjauan pustaka dari berbagai sumber relevan mengenai algoritma kompresi data, tanpa percobaan langsung atau eksperimen laboratorium. Tujuan utama dari pendekatan ini adalah untuk membandingkan kinerja sejumlah algoritma kompresi yang telah dianalisis oleh peneliti sebelumnya, seperti *Huffman Coding, Lempel-Ziv-Welch (LZW), Burrows-Wheeler Transform (BWT)*, dan *Deflate*.

Data diperoleh dari jurnal ilmiah nasional dan internasional, prosiding konferensi, dan repositori ilmiah online seperti *Google Scholar*, *ResearchGate*, dan *DergiPark*. Pemilihan referensi dilakukan secara selektif berdasarkan kesesuaian topik, reputasi penerbit, dan tahun publikasi (sebaiknya diterbitkan dalam lima tahun terakhir untuk menjaga aktualitas informasi).

Proses pencarian dilakukan menggunakan kata kunci seperti "perbandingan algoritma kompresi data", "*Huffman* vs *LZW*", dan "kinerja kompresi biner". Setiap referensi yang diperoleh dianalisis berdasarkan parameter kunci

DOI: https://doi.org/10.31004/riggs.v4i2.1396

seperti tingkat kompresi, waktu pemrosesan, serta jenis data yang digunakan. Hasil dari setiap referensi dibandingkan untuk mengidentifikasi pola atau tren kinerja antara algoritma

3. Hasil dan Pembahasan

3.1 *Pengantar kompresi data*

Kompresi data adalah proses mengubah data asli menjadi bentuk yang lebih ringkas dengan tujuan utama mengurangi ukuran file tanpa menghilangkan informasi penting di dalamnya. Teknik ini sangat krusial dalam era digital saat ini, di mana pertumbuhan data yang pesat menuntut efisiensi tinggi dalam penyimpanan dan transfer data, baik melalui media fisik maupun jaringan internet. Dengan kompresi, kebutuhan ruang penyimpanan dapat ditekan dan waktu pengiriman data menjadi lebih singkat, sehingga mendukung performa sistem informasi modern.

3.1.1 Jenis-jenis Kompresi Data

Secara umum, kompresi data terbagi menjadi dua kategori utama, yaitu lossless dan lossy:

a. Kompresi Lossless: Metode ini memungkinkan data yang telah dikompresi untuk dikembalikan ke bentuk aslinya secara utuh tanpa kehilangan informasi sedikit pun. Kompresi lossless sangat penting untuk aplikasi yang membutuhkan keakuratan data, seperti dokumen teks, file program, data medis, serta arsip digital. Contoh algoritma populer dalam kategori ini adalah Huffman Coding, Lempel-Ziv-Welch (LZW), Lempel–Ziv–Markov chain algorithm (LZMA), Prediction by Partial Matching (PPM), dan Burrows-Wheeler Transform (BWT). Penggunaan kompresi lossless memastikan integritas data tetap terjaga, sehingga sangat cocok untuk file yang membutuhkan keutuhan informasi.
b. Kompresi Lossy: Metode ini mengurangi ukuran file dengan cara menghilangkan sebagian data yang dianggap kurang penting atau tidak terlalu memengaruhi kualitas informasi secara signifikan. Kompresi lossy umum digunakan pada file multimedia seperti gambar (JPEG), audio (MP3), dan video (MPEG), di mana sedikit kehilangan data masih dapat diterima asalkan hasil akhirnya tetap dapat dinikmati oleh pengguna. Namun, data hasil kompresi lossy tidak dapat dikembalikan ke bentuk aslinya secara sempurna.

3.1.2 Pentingnya Kompresi Data Lossless

Fokus pada kompresi lossless sangat relevan dalam pengelolaan dokumen digital dan file penting lainnya. Keunggulan utama dari lossless compression adalah kemampuannya untuk mempertahankan keutuhan data, sehingga file yang telah dikompresi dapat didekompresi kembali ke bentuk semula tanpa perubahan apa pun. Hal ini sangat penting untuk dokumen hukum, arsip pemerintah, data penelitian, dan aplikasi lain yang tidak mentoleransi kehilangan informasi.

Selain itu, kompresi lossless juga berperan dalam efisiensi penyimpanan dan transmisi data. Dengan ukuran file yang lebih kecil, kebutuhan kapasitas penyimpanan dapat ditekan dan proses transfer data melalui jaringan menjadi lebih cepat dan hemat bandwidth. Dalam penelitian terbaru, algoritma LZMA2, misalnya, menunjukkan performa kompresi lossless terbaik dengan rasio kompresi tinggi dan penghematan ruang yang signifikan, meskipun efektivitas setiap algoritma sangat bergantung pada karakteristik data yang dikompresi.

3.1.3 Perkembangan dan Implementasi Algoritma Kompresi Lossless

Berbagai algoritma kompresi lossless telah dikembangkan dan diimplementasikan dalam aplikasi nyata. Sebagai contoh, algoritma LZW (Lempel-Ziv-Welch) banyak digunakan dalam format file seperti GIF dan TIFF, sedangkan Huffman Coding sering digunakan dalam kompresi teks dan file arsip. Penelitian terbaru juga menyoroti pentingnya pemilihan algoritma yang tepat sesuai dengan jenis dan karakteristik data, karena tidak ada satu algoritma yang selalu unggul untuk semua tipe data.

Beberapa algoritma lossless yang umum digunakan antara lain:

a. Run Length Encoding (RLE): Efektif untuk data yang memiliki banyak pengulangan karakter.
b. Lempel-Ziv-Welch (LZW): Banyak digunakan pada file gambar dan teks.

DOI: https://doi.org/10.31004/riggs.v4i2.1396

c. Huffman Coding: Cocok untuk kompresi teks dan file dengan distribusi frekuensi karakter yang tidak merata.
d. Arithmetic Coding: Memiliki efisiensi tinggi untuk data dengan distribusi probabilitas tertentu

3.2 *Rasio kompresi*

Berdasarkan hasil analisis, tingkat rasio kompresi sangat bergantung pada jenis data dan algoritma yang digunakan (lihat Tabel 1). Algoritma Huffman Coding, yang bekerja berdasarkan frekuensi kemunculan karakter, mampu mengurangi ukuran data teks antara 45% hingga 60%, terutama pada data yang memiliki pola pengulangan. Hal ini sejalan dengan studi Aruan & Rahayu (2023), yang menyatakan bahwa metode Huffman efektif dalam mengurangi redundansi data teks. Untuk jenis data biner dan gambar digital, Lempel-Ziv-Welch (LZW) menunjukkan performa lebih baik dengan rasio kompresi sekitar 35% hingga 50%, karena kemampuannya dalam mengenali pola-pola yang lebih kompleks. Sementara itu, Run-Length Encoding (RLE) sangat efektif digunakan pada gambar dengan warna yang seragam, dengan rasio kompresi antara 30% hingga 40%, namun kurang efisien ketika diterapkan pada data acak atau yang memiliki entropi tinggi.

Table Perbandingan Algoritma Kompresi Data

Algoritma	Jenis Kompresi	Rasio Kompresi	Kecepatan Kompresi/Dekompresi	Kelebihan	Kekurangan
Huffman Coding	Lossless	45% – 60% (teks)	120 ms / 95 ms	Efektif untuk data teks Dekompresi cepat, hasil kompresi efisien	- Kompresi lebih lambat dari RLE dan LZW - Kurang adaptif untuk data acak
LZW	Lossless	35% – 50% (biner dan citra)	85 ms / 110 ms	Cocok untuk data biner, citra Transfer data (REST API)	- Dekompresi lebih lambat dari kompresi - Kurang optimal pada data sangat acak
Golomb Coding	Lossless	Tergantung distribusi data	Kecepatan Kompresi/Dekompresi Golomb Coding ini tergantung pada jenis datanya.	Efisien untuk data dengan angka kecil sering muncul	Tidak cocok untuk data dengan distribusi acak atau besar
Elias Gamma	Lossless	Efisien pada angka kecil	kecepatan kompresi dan dekompresi algoritma Elias Gamma juga sangat bergantung pada jenis datanya	Unik, prefix-free, cocok untuk data dengan banyak angka kecil	Tidak optimal untuk angka besar
Deflate	Lossless (gabungan)	Umumnya tinggi (ZIP, gzip)	Tidak paling cepat dalam kompresi, juga bukan yang terbaik dalam rasio kompresi.	Kombinasi LZ77 & Huffman, efisien & cepat, dipakai luas (ZIP/gzip)	Kompleksitas lebih tinggi
RLE	Lossless	30% – 40% (data homogen)	45 ms / 38 ms	Sangat cepat, sederhana	Tidak efektif untuk data acak, bisa memperbesar ukuran data

Rasio kompresi yang tinggi sangat bermanfaat dalam konteks penyimpanan, memungkinkan kapasitas media penyimpanan digunakan secara lebih efisien. Selain itu, data yang lebih kecil juga mempercepat proses transfer, terutama pada jaringan dengan bandwidth terbatas.

3.3 *Kecepatan kompresi*

Kecepatan dalam proses kompresi dan dekompresi merupakan salah satu faktor kunci dalam pemilihan algoritma kompresi, terutama pada aplikasi yang menuntut waktu respons sangat singkat seperti sistem real-

DOI: https://doi.org/10.31004/riggs.v4i2.1396

time, pengiriman data instan, dan layanan berbasis cloud. Selain efisiensi rasio kompresi, performa waktu eksekusi menjadi pertimbangan utama agar tidak terjadi bottleneck pada sistem.

3.3.1 Analisis Kecepatan Kompresi dan Dekompresi

Berdasarkan pengujian pada file berukuran 40KB, terdapat perbedaan signifikan dalam kecepatan kompresi dan dekompresi antara beberapa algoritma populer:

a. *Run Length Encoding (RLE)*
RLE menonjol dalam hal kecepatan, dengan waktu kompresi rata-rata 45 ms dan dekompresi 38 ms. Kecepatan tinggi ini didapat berkat mekanisme kerjanya yang sederhana, yaitu hanya mencatat jumlah pengulangan karakter secara berurutan. Namun, keunggulan ini harus dibayar dengan efisiensi kompresi yang rendah, karena *RLE* hanya optimal pada data yang memiliki banyak pola pengulangan. Pada data yang tidak berpola, *RLE* bahkan bisa memperbesar ukuran file (Prabiantissa et al., 2023).

b. *Lempel-Ziv-Welch (LZW)*
LZW membutuhkan waktu sekitar 85 ms untuk kompresi dan 110 ms untuk dekompresi. Waktu eksekusi yang lebih lama dibanding RLE disebabkan oleh proses pembangunan dan pemeliharaan struktur kamus (*dictionary*) yang digunakan untuk merepresentasikan pola data. Meski demikian, *LZW* menawarkan rasio kompresi yang lebih baik pada data teks dan gambar yang berpola, serta banyak digunakan dalam aplikasi pengiriman data melalui jaringan (Aswar, 2023).

c. *Huffman Coding*
Huffman membutuhkan waktu kompresi sekitar 120 ms dan waktu dekompresi 95 ms. Algoritma ini memerlukan waktu lebih lama pada tahap kompresi karena harus membangun pohon kode berdasarkan frekuensi kemunculan karakter. Namun, proses dekompresinya relatif cepat karena hanya perlu menerjemahkan kode biner ke karakter aslinya. Huffman sangat efektif untuk data dengan distribusi karakter yang tidak merata (Lazuardi Imani et al., 2021).

3.3.2 Adaptabilitas dan Kesesuaian Algoritma

Tidak ada satu algoritma yang konsisten unggul dalam semua skenario data. Adaptabilitas algoritma sangat dipengaruhi oleh karakteristik data dan kebutuhan aplikasi:

a. *Huffman* Adaptif
Versi adaptif dari *Huffman* mampu meningkatkan efisiensi kompresi sekitar 15–20% dibandingkan versi statis, khususnya pada data dengan pola distribusi karakter yang berubah-ubah. Hal ini membuatnya lebih fleksibel dalam menangani data dinamis (Mansyuri, 2021).

b. *LZW* untuk Transfer Data Jaringan
LZW sangat efektif dalam aplikasi pengiriman data melalui jaringan, seperti *REST API*, dengan peningkatan efisiensi transfer sebesar 11,67–16,67% dibandingkan data tanpa kompresi. Ini menjadikan *LZW* pilihan utama untuk aplikasi yang membutuhkan efisiensi *bandwidth* dan kecepatan transfer (Aswar, 2023).

c. RLE untuk Data Berpola
RLE sangat ideal untuk data yang memiliki banyak pengulangan, seperti bitmap sederhana atau data sensor yang stabil. Namun, pada data yang tidak berpola atau memiliki variasi tinggi, RLE tidak hanya kurang efisien, tetapi juga berpotensi memperbesar ukuran file.

3.4 *Faktor-faktor yang mempengaruhi performa*

Beberapa aspek penting yang menentukan kinerja algoritma kompresi meliputi:

a. Jenis data: Algoritma Huffman lebih optimal digunakan pada data teks yang memiliki banyak pengulangan, sementara LZW lebih cocok diterapkan pada data biner dan gambar digital.

DOI: https://doi.org/10.31004/riggs.v4i2.1396

b. Ukuran file: Semakin besar ukuran file, proses kompresi dan dekompresi memerlukan waktu lebih lama, meskipun algoritma yang efisien mampu mengatasi peningkatan tersebut.
c. Sifat data: Data dengan tingkat pengulangan atau redundansi tinggi dapat dikompresi dengan lebih baik. Sebaliknya, data yang acak atau memiliki entropi tinggi cenderung sulit dikompresi secara efektif.

3.5 *Implikasi untuk penyimpanan*

Proses kompresi data memiliki peran yang sangat penting dalam meningkatkan efisiensi penyimpanan, karena dapat memperkecil ukuran file secara signifikan tanpa menghilangkan informasi yang esensial, khususnya apabila algoritma yang digunakan sesuai dengan jenis data yang dikompresi. Sebagai contoh, untuk data teks, algoritma seperti Huffman Coding, LZ77, dan LZW sangat efektif dalam mengurangi elemen yang berulang (redundansi) sehingga ukuran file menjadi lebih kecil. Sementara itu, untuk data dalam bentuk gambar dan audio, metode seperti JPEG, PNG, MP3, serta AAC diaplikasikan berdasarkan sifat visual maupun suara dari data tersebut.

Di sisi lain, algoritma kompresi juga memiliki peran penting dalam menjaga keseimbangan antara kualitas data dan ukuran file, terutama pada kompresi berjenis *lossy* dan *lossless*. Kompresi *lossless* sangat diperlukan ketika data harus dipulihkan secara utuh tanpa kehilangan informasi sedikit pun, seperti pada file sistem atau dokumen penting. Sebaliknya, kompresi *lossy* lebih cocok digunakan pada data seperti gambar atau suara yang masih dapat diterima meskipun terjadi sedikit penurunan kualitas. Kompresi menjadi lebih efisien ketika diterapkan pada data yang memiliki tingkat pengulangan tinggi, karena algoritma dapat mengenali pola yang sama untuk menghemat ruang penyimpanan. (Aruan, M. C., & Wanti Rahayu. (2023).

Algoritma seperti LZW dan Huffman memberikan kombinasi yang seimbang antara efektivitas kompresi dan kecepatan proses, sehingga sangat sesuai digunakan dalam sistem penyimpanan yang memerlukan respons cepat atau akses data secara real-time. Sebaliknya, algoritma Arithmetic Coding memang mampu menghasilkan file yang lebih kecil, namun memerlukan waktu pemrosesan yang lebih panjang, sehingga penggunaannya harus dipertimbangkan secara matang pada perangkat dengan kapasitas prosesor atau memori yang terbatas.

Oleh karena itu, pemilihan algoritma kompresi yang tepat tidak hanya berperan dalam menghemat kapasitas penyimpanan, tetapi juga penting dalam menyesuaikan kinerja sistem, baik dari sisi kecepatan pengolahan data maupun efisiensi penggunaan sumber daya yang tersedia. (Baidoo , P. K. (2023).

3.6 *Diskusi dan keterkaitan dengan literatur*

Temuan dalam penelitian ini memperkuat pandangan bahwa efektivitas algoritma kompresi sangat dipengaruhi oleh jenis dan karakteristik data, serta konteks penggunaannya. Sejumlah studi terdahulu juga menyimpulkan bahwa pemilihan algoritma harus disesuaikan dengan tipe data serta tujuan spesifik dari proses kompresi itu sendiri (Aruan & Rahayu, 2023; Xu et al., 2024). Karena tidak ada satu algoritma pun yang unggul dalam semua kondisi, pendekatan yang adaptif maupun berbasis kombinasi algoritma menjadi alternatif yang layak untuk menjawab tantangan penyimpanan dan pengiriman data pada masa kini.

Tidak ada algoritma kompresi tunggal yang unggul dalam semua aspek, melainkan pemilihan algoritma sangat bergantung pada karakteristik data dan tujuan penggunaan.

3.7 *Rekomendasi dan arah pengembangan*

Berdasarkan temuan dalam penelitian ini, beberapa rekomendasi yang dapat diterapkan meliputi:

a. Penerapan algoritma Huffman sangat disarankan untuk pengolahan dokumen teks serta data dengan pola pengulangan karakter yang konsisten.
b. Untuk data biner, file citra digital, dan kebutuhan pengiriman data melalui jaringan, algoritma LZW menjadi pilihan yang efisien.
c. Algoritma RLE, atau kombinasi dengan metode lain seperti Huffman, cocok diterapkan pada data visual dengan area warna seragam.
d. Perlu dilakukan pengembangan lebih lanjut terhadap algoritma kompresi adaptif atau berbasis hibrida agar lebih optimal saat digunakan pada data dengan karakteristik campuran.

DOI: https://doi.org/10.31004/riggs.v4i2.1396

Seluruh pendekatan ini selaras dengan perkembangan teknologi sistem penyimpanan dan komunikasi data masa kini, yang menuntut efisiensi tinggi, kecepatan pemrosesan, serta kemampuan beradaptasi terhadap beragam jenis data (Xu et al., 2024).

4. Kesimpulan

Berdasarkan hasil analisis, dapat disimpulkan bahwa kinerja algoritma kompresi sangat dipengaruhi oleh sifat data yang diproses serta kebutuhan spesifik penggunaannya. Tidak ada satu pun algoritma yang secara mutlak unggul dalam semua kondisi. Misalnya, algoritma Huffman sangat cocok digunakan untuk jenis data teks yang memiliki banyak pengulangan karakter, karena mampu menghasilkan rasio kompresi yang efisien serta proses dekompresi yang cepat. Sementara itu, algoritma LZW lebih efektif digunakan untuk data biner dan gambar digital, serta mendukung dengan baik proses transmisi data berbasis web. Di sisi lain, algoritma RLE memang memiliki kecepatan tinggi dalam proses kompresi dan dekompresi, namun kurang optimal saat diterapkan pada data dengan keragaman tinggi. Pendekatan gabungan seperti Huffman-RLE dapat memberikan hasil yang optimal dalam kondisi tertentu, khususnya pada data seperti gambar yang memiliki area seragam. Faktor-faktor seperti ukuran file, pola distribusi data, dan jenis informasi juga sangat berpengaruh terhadap efektivitas metode yang dipilih. Oleh karena itu, pemilihan algoritma kompresi harus mempertimbangkan konteks penggunaan secara menyeluruh agar diperoleh hasil yang paling optimal.

Referensi

Implementasi Algoritma Shannon-Fano untuk Kompresi File Teks. (2020). Integer Journal.

Aruan, M. C. (2023). Analisis Performa Algoritma Kompresi Data dalam Penyimpanan dan Transfer Data. *Jurnal Inovasi dan Tren*, 1(2), 228-232.

Kadhim, D. J. (2024). Exploring Text Data Compression: A Comparative Study of Adaptive Huffman and LZW Approaches. . *BIO Web of Conferences*, 97, 00035.

Kombinasi Metode Run-Length Encoding (RLE) dengan Algoritma Arithmetic Coding untuk Kompresi Data Lossless. (2023). *Jurnal Interaksi Saintek*.

LodDos. (2021). Perancangan Aplikasi Kompresi File MP3 dengan Algoritma LZW. *Institute Journal* .

Mansyuri, A. (2021). Kompresi Data Lossless dan Lossy. *Jurnal Informatika*.

Marzuki, I. (2017). Aplikasi Kompresi Untuk Pengiriman Data Menggunakan Metode LZW(Lemple Ziv Welch). *Energy: Jurnal Ilmiah Ilmu-Ilmu Teknik,* , 38-42.

Nurcahyo, S. a. (2014). Rainfall Prediction in Kemayoran Jakarta Using Hybrid Genetic Algorithm (GA) and Partially Connected Feedforward Neural Network (PCFNN). *Information and Communication Technology (ICoICT)*, (pp. 166-171).

Penerapan Algoritma Huffman Code untuk Kompresi dan Dekompresi Data. (2024). *CoSIE Journal*.

Prabiantissa, e. a. (2023). Analisis Perbandingan Kompresi Citra Menggunakan Algoritma Run Length Encoding dan Huffman Coding. *Seminar Nasional Teknologi Informasi (TIN),*.

Tangerang, U. M. (2025). Analisis dan Perancangan Aplikasi Algoritma Huffman untuk Kompresi Data. *JIKA (Jurnal Informatika)*.

Xu, J., Niu, K., Liang, Z., Zhang. (2024). Sinematic huffman coding using synonymous mapping. *arXiv preprint aXiv*.

Aruan, M. C., & Wanti Rahayu. (2023). Analisis Performa Algoritma Kompresi Data dalam Penyimpanan dan Transfer Data. *LANCAH: Jurnal Inovasi Dan Tren,* 1(2), 228–232. https://doi.org/10.35870/ljit.v1i2.2157

Baidoo , P. K. (2023). Comparative Analysis of the Compression of Text Data Using Huffman, Arithmetic, Run-Length, and Lempel Ziv Welch Coding Algorithms. *Journal of Advances in Mathematics and Computer Science,* 38(9), 144–156. https://doi.org/10.9734/jamcs/2023/v38i91812

Marzuki, I. (2017). Aplikasi Kompresi Untuk Pengiriman Data Menggunakan Metode LZW (Lemple Ziv Welch). *Energy: Jurnal Ilmiah Ilmu-Ilmu Teknik*, 7(2), 38-43.

DOI: https://doi.org/10.31004/riggs.v4i2.1396