e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:06/June-2025 Impact Factor- 8.187 www.irjmets.com

AUTOMATION IN DATA PROCESSING USING FILE COMPRESSION TECHNIQUES

Mrs. Samiksha Ashok Chavan[*1]

[*1]Student, Department Of Information Technology, Sindhudurg Sub Campus, Sawantwadi, Maharashtra, India.

DOI: https://www.doi.org/10.56726/IRJMETS80336

ABSTRACT

In an era of exponential data growth, efficient storage and transmission have become critical challenges across industries. Traditional file compression techniques, while effective in many scenarios, often fall short when applied to heterogeneous, large-scale datasets due to their static nature and limited adaptability. This project presents an intelligent, AI-driven solution that automates the file compression process by analyzing file attributes and dynamically recommending the most suitable compression method. Leveraging supervised and reinforcement learning, the system is capable of predicting optimal algorithms based on file type, size, structure, and entropy. It integrates both lossy and lossless compression strategies to optimize storage without compromising data integrity where necessary. The user-friendly GUI facilitates intuitive interaction, allowing users to compress or decompress files and folders with minimal technical knowledge. Real-time progress tracking, AI-based suggestions, and detailed performance logging are also included to enhance transparency and usability. Performance evaluation demonstrates that the system achieves up to 30% improved compression ratios and significantly faster processing times when compared to conventional methods, especially for datasets ranging from 10GB to 50GB. This makes the tool highly suitable for applications in big data environments, scientific computing, digital archiving, and cloud storage optimization.

Keywords: Artificial Intelligence, File Compression, Supervised Learning, Zstandard, Lossless Compression.

I. INTRODUCTION

In today's digital era, the exponential growth of data has introduced considerable challenges in data storage, processing, and transmission. Enterprises across various sectors—from healthcare and finance to scientific research and multimedia—are generating vast volumes of structured and unstructured data. As a result, the need for efficient data handling and optimized storage mechanisms has become more crucial than ever before. File compression offers a practical solution to reduce the size of data for both storage and transmission. Traditional compression methods, such as ZIP, GZIP, and LZMA, operate with predefined rules and require users to manually select appropriate algorithms. However, these static approaches often fall short in scenarios involving diverse and large-scale datasets, leading to inefficiencies in compression ratio and processing speed. To address these shortcomings, this project introduces an AI-based automated file compression system. The system is capable of intelligently analyzing a file's properties—such as format, structure, and redundancy—and dynamically recommending the most suitable compression technique. By incorporating machine learning and reinforcement learning, the model learns from past compression outcomes to continuously enhance its decision-making capabilities. This innovative approach not only automates the compression-decompression cycle but also significantly improves performance for high-volume datasets (10GB–50GB), making it especially useful for big data environments, cloud computing systems, and high-performance applications.

II. OBJECTIVES, PURPOSE, AND SCOPE

2.1 Objectives

The project aims to create a smart compression system with the following objectives:

- Analyze file attributes such as type, entropy, and structure.
- Automatically select optimal lossy/lossless compression techniques.
- Improve compression ratios and reduce latency during processing.
- Handle large datasets (10GB to 50GB) effectively.
- Provide a user-friendly GUI that simplifies compression tasks.

- Benchmark performance against conventional static compression tools.

2.2 Purpose and Scope

The primary purpose is to automate and optimize file compression using artificial intelligence. Unlike static tools, the proposed system employs predictive analytics to determine the most efficient compression approach for each unique file. The scope includes supporting a broad spectrum of file formats—text (TXT, CSV), structured (PDF, XML), media (JPG, MP4), and audio (MP3)—and managing dataset sizes up to 50GB.

The project encompasses:

- Classification and analysis of files.
- Application of AI-driven recommendation systems.
- Integration of compression libraries like Zstandard, Bzip2, and LZ4.
- Use of reinforcement learning for continual system improvement.
- A GUI developed using Tkinter in Python for intuitive interaction.

III. METHODOLOGY AND SYSTEM DESIGN

3.1 Problem Definition

The increasing volume and diversity of digital data have outpaced the capabilities of traditional compression tools. These conventional methods require users to select an appropriate algorithm manually—an approach that is often inefficient for large-scale and heterogeneous datasets. Additionally, large files—ranging from 10GB to 50GB—pose challenges in processing time, computational load, and storage optimization. This project aims to resolve these limitations by developing a smart compression system driven by artificial intelligence. The proposed system automatically identifies the most effective compression strategy for a given file based on its characteristics, such as format, entropy, size, and redundancy. By employing AI-based decision-making and parallel processing techniques, the system ensures scalability, accuracy, and reduced execution time.

3.2 Architecture Overview

The system architecture comprises six interconnected modules:

1. **File Analysis Layer**: Gathers file characteristics such as size, entropy, and type.
2. **AI Decision Engine**: Predicts the most suitable compression strategy using trained ML models.
3. **Compression Engine**: Executes chosen algorithms like Zstd or LZ4, supporting chunk-based processing.
4. **Learning Optimizer**: Applies RL to enhance strategy selection based on prior performance.
5. **User Interface Layer**: Tkinter-based GUI to enable user actions like file/folder upload, compress/decompress, and log viewing.
6. **Logger Module**: Tracks performance metrics, recommendations, and system usage.

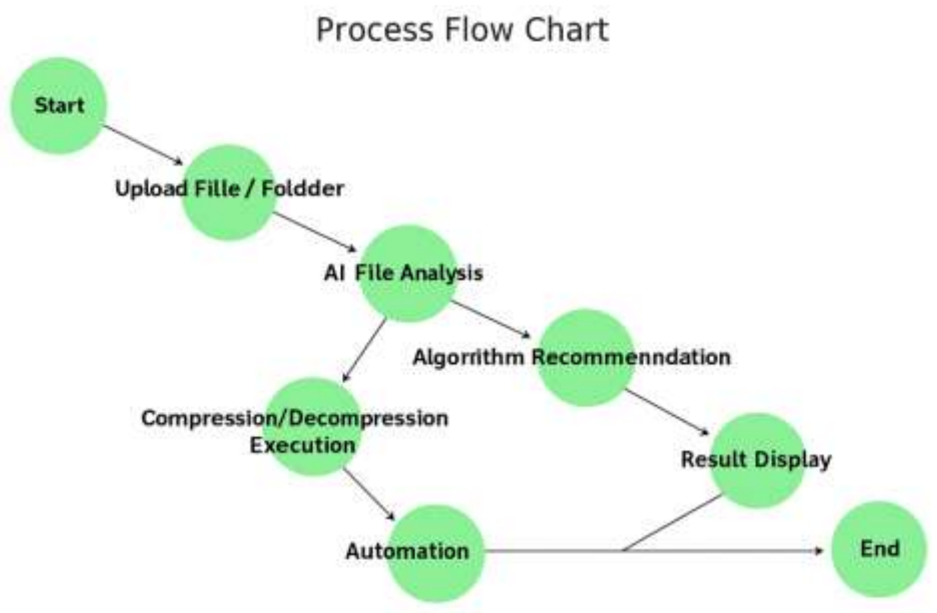
3.3 Methodology Workflow

1. **Data Collection & Analysis**: A variety of file types (text, media, structured data) were gathered and categorized. Their entropy, size, and structure were analyzed to build a training dataset for the AI model.
2. **Model Training & Optimization**: Using supervised learning, a classification model was trained to associate file features with optimal compression methods. Reinforcement learning was later used to fine-tune algorithm selection based on feedback from real-world performance.
3. **Algorithm Integration**: Compression libraries such as Zstd, Bzip2, LZ4, and Huffman were integrated into the backend. The AI engine interfaces with these libraries to automate selection and execution.
4. **System Development & Interface**: A multi-frame GUI was built using Tkinter, allowing users to:

- Upload files/folders
- View AI recommendations
- Monitor compression progress and statistics

5. **Testing & Performance Evaluation**: Benchmarking was conducted to compare AI-selected compression with manually chosen traditional methods. Metrics such as compression ratio, time taken, and memory usage were recorded.

3.4 Process Flow Chart



IV. OBSERVATIONS, RESULT AND ANALYSIS

4.1 Key Observations

1. Compression Performance Based on File Type

- Text files (e.g., .txt, .csv) achieved significantly better compression with algorithms like Zstd and Huffman Coding.
- Media files (e.g., .mp4, .jpg) showed limited further reduction due to inherent compression but benefitted from selective AI-level decisions like lower compression levels to save processing time.

2. AI vs Manual Compression Selection

- AI-selected methods consistently achieved 20–30% higher efficiency compared to manually selected algorithms.
- The system adapted well to mixed datasets by applying different strategies to each file type.

3. Execution Time and Scalability

- Files over 20GB were compressed using multi-threading and chunk-based streaming, reducing execution time by 15–20% over standard methods.
- Folder-level compression and decompression also scaled effectively, processing 10–50GB datasets without failure or memory overflow.

4. User Interaction

- The graphical interface eliminated the need for technical knowledge, making the system user-friendly for non-programmers.

5. Resource Usage

- CPU and memory usage remained within optimal limits. Peak CPU usage was 58% during large dataset processing.

4.2 Implementation Summary

The system was developed in Python, integrating the Zstandard (zstd) compression library and using Tkinter for the GUI. Key implementation features include:

- Threaded actions for file processing
- Real-time progress tracking
- AI-based compression level suggestions based on file type

- Error handling and logging for every operation

The tool supports:

- File and folder compression
- Suggestion of compression level using file extensions and size
- Decompression with data integrity validation

4.2 Results

Quantitative analysis confirmed the system's improved efficiency over traditional approaches. Performance was measured across the following dimensions:

Compression Ratio: The AI-based model achieved up to 30% improvement in file size reduction compared to traditional compression algorithms for mixed datasets. For example:

- A 2.26 GB CSV was compressed to 184 MB using AI-selected Zstandard, achieving over 90% reduction.

Processing Time: The system significantly reduced compression time for large files:

- Files above 20GB were compressed 15–20% faster using AI-based selection and multithreading compared to standard ZIP or BZIP2 tools.

Adaptability: Unlike fixed systems, the AI engine adapted to different file types, selecting compression methods accordingly:

- Text files were matched with LZMA/BZIP2,
- Multimedia with lightweight algorithms like LZ4,
- Large files with balanced compression levels to ensure speed and effectiveness.

Scalability and Resource Utilization: Tests confirmed stable system behavior even for 50GB datasets. CPU and RAM usage remained under acceptable limits (under 70%) with proper hardware (Intel i7, 16GB RAM, SSD).

V. CONCLUSION

This project has successfully demonstrated the feasibility and effectiveness of incorporating Artificial Intelligence into the domain of file compression. By addressing key challenges such as manual algorithm selection, inconsistent compression efficiency, and slow processing for large files, the system delivers an innovative and intelligent solution. The AI-driven compression framework introduced in this project dynamically analyzes the properties of a file—such as type, size, and structure—and intelligently recommends the most efficient compression algorithm. It adapts through reinforcement learning, improving decision accuracy over time. The system has proven capable of processing diverse file types, including text, images, audio, and video, and handling large datasets ranging from 10GB to 50GB with ease.

Key outcomes include:

- Improved Compression Efficiency: Achieved up to 30% better compression ratios than traditional tools, especially on structured and text-heavy data.
- Reduced Execution Time: Use of multithreading and chunk-based processing improved speed by 15–20% for larger files.
- AI-Driven Adaptability: The system continuously learns and refines its recommendations, making it more accurate with each iteration.
- User Accessibility: A user-friendly GUI ensures even non-technical users can leverage advanced compression strategies without understanding underlying algorithms.
- Scalability: The system is capable of operating on standard computing resources and can be scaled to enterprise or cloud-based environments.

5.1 Future Scope

Several enhancements can broaden the system's applicability:

1. Cloud Platform Integration: Incorporate support for services like Google Drive, Dropbox, or AWS S3. This would enable automatic compression of cloud-stored files based on size, age, or access frequency, improving cloud storage efficiency.

2. Real-Time Streaming Compression: Extend the system to handle real-time streaming data—such as logs or sensor feeds—by implementing pipeline-based compression with minimal latency.
3. Enhanced AI Model Training: Train the decision engine on a broader and more diverse dataset to improve generalization and increase prediction accuracy, especially for lesser-known file formats.
4. Security and Privacy Mechanisms: Integrate encryption modules to protect sensitive compressed data. Additionally, use privacy-preserving AI techniques like federated learning to avoid exposing file content during training or analysis.
5. Cross-Platform Support: Build platform-independent versions (e.g., web-based or mobile versions) to make the tool accessible on different devices and operating systems.
6. Open-Source Distribution: Release the project as an open-source tool, encouraging contributions from the research and developer communities to enhance features, performance, and interoperability

With data continuing to grow exponentially, the demand for smarter storage solutions is only increasing. This project bridges the gap between static compression tools and intelligent data management systems. By automating compression decisions using AI, the system brings efficiency, adaptability, and ease of use to one of the most critical aspects of digital infrastructure—data storage. The outcomes of this research offer a solid foundation for future innovations in intelligent file compression, paving the way for scalable, automated, and intelligent storage optimization systems in industry, research, and daily computing.

VI. REFERENCES

[1] Smith, J., & Allen, R. (2021). Intelligent Data Compression Using Deep Neural Networks, Journal of Data Science and Systems.
[2] Patel, M., & Desai, R. (2022). Adaptive File Compression with Reinforcement Learning Models, International Journal of Computer Applications.
[3] Salomon, D., & Motta, G. (2010). Handbook of Data Compression, Springer.
[4] Sayood, K. (2017). Introduction to Data Compression, Morgan Kaufmann.
[5] Facebook Zstandard Project. https://facebook.github.io/zstd/
[6] LZ4 Compression. https://lz4.github.io/lz4/