VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Fall Semester 2025-2026

CSI3019 – Advanced Data Compression Techniques

TOPIC:

Semantic-Aware Archival of Low-Impact Social Media Posts Using Tiered Compression and Cold Storage

R.S. SHREENIDHI 22MID0306

R SIVANI 22MID0005

ALYSSA MARIAM KURUVILLA 22MID0076

KEERTHANA G 22MID0144

Abstract

Exponential growth in the number of social media platforms over recent years has resulted in an unprecedented rise in user-generated content across the world. Millions of posts, comments, images, and videos are generated and shared every minute, thus forming colossal and ever-expanding repositories. The fact remains, however, that most of this content is transient in nature-that is, it loses its relevance, engagement, and utility within a very short period of time. Social media platforms nonetheless store such data indefinitely, which raises infrastructure costs, leads to storage inefficiencies, and consumes energy. This has become a critical challenge in big data management: how can the platforms maintain access to relevant information while reducing the overhead of maintaining low-value data.

The paper presents the design of an Intelligent Semantic-Aware Archival System for Social Media Data, developed to optimize data life-cycle management by means of context-sensitive analytics, classification, and compression. It will automatically identify low-impact social network posts, taking into consideration several factors such as engagement metrics (likes, shares, comments, views), semantic relevance, and contextual importance over time. All these parameters set an impact score for each post that defines its storage priority and categorizes it into one of three tiers: hot, warm, or cold. This multi-tier architecture allows for fine-grained data management whereby posts with a high impact and continued engagement remain in fast-access storage, while older or low-utility posts are seamlessly transitioned to compressed cold storage.

The hot tier comprises trending or actively engaged content that requires immediate accessibility. Data in this tier is stored using minimal compression to guarantee low latency and high retrieval speed. The warm tier is an intermediate level for content with moderate engagement, utilizing balanced compression algorithms optimized for cost-accessibility. Finally, the cold tier consists of outdated or low-impact/least-accessed posts that are subjected to high-ratio adaptive compression techniques and stored in very low-cost, energy-efficient archival systems. This tiering is dynamic, informed by real-time analytics and historical usage patterns, which enables the storage decisions to always be optimized.

A key innovation of this project is the introduction of semantic understanding to the archival decision-making process. Other data archiving systems use only numerical metrics to make their decisions, but this system makes use of Natural Language Processing and machine learning algorithms to assess the contextual meaning and sentiment of posts. This will ensure that posts with long-term contextual or cultural significance, such as those involving major events, public announcements, or influential discussions, are not subject to premature archival when engagement metrics are low. The integration of semantic analysis with an engagement-based scoring system allows for smarter, more contextually aware data classification.

It achieves this by storing lightweight metadata for every post in the system, such as timestamp, author ID, engagement summary, sentiment polarity, and semantic topic tags. The metadata facilitates fast indexing and selective decompression for users and automated systems to identify and recover specific content without decompressing whole archives. This results in a storage framework that balances efficiency and accessibility, enabling scalable long-term data management for high-volume social platforms.

The proposed system also integrates adaptive compression strategies, in which compression methods would be dynamically chosen depending on the characteristics of the data and tier classification. Such posts heavy with text might consider lossless compression methods such as Huffman, while image and video content could use lossy methods optimized for minimal perceptual loss. The adaptability in this regard will further improve both compression and energy efficiency related to different data modalities. Apart from cost reductions, the system has a number of broad benefits: it reduces storage overhead by an order of magnitude, enabling social media companies to dedicate high-performance resources to

those few pieces of highly trafficked content that create true value; it enables scalability, as platforms can handle exponentially growing data volumes without proportional growth in infrastructure; and finally, the solution contributes to sustainable computing practices by reducing the energy footprint of long-term data storage-a critical component in global efforts toward greener digital ecosystems. Essentially, this project demonstrates how social media data management could be revamped by bringing in intelligent and context-aware automation. Semantic understanding, dynamic tiered storage, and adaptive compression-together these provide a robust approach to optimizing storage infrastructure, with the system ready for the future. It bridges the gap between big data analytics, AI, and cloud storage architecture to enable better data retention policies without compromise on performance or information value. It is economical, scalable, and eco-friendly, thus providing a responsible solution to manage the ever-growing flood of digital content emerging over social media.

Introduction

Social media has emerged as one of the most widely used communication tools in the current digital era, influencing how people, businesses, and governments engage with the outside world. Every second, enormous volumes of user-generated content (UGC) are produced by platforms like Facebook, Instagram, Twitter (X), LinkedIn, TikTok, and YouTube in the form of posts, photos, videos, comments, and reactions. Big data is largely a result of the unprecedented explosion in data generation brought about by this continuous flow of content. The storage, management, and retrieval of vast volumes of digital content is one of the most urgent issues facing information technology, even though the abundance of information has transformed marketing, communication, and global awareness.

Social media companies gather petabytes of data from their billions of users every day, most of which is irrelevant or has little impact within days or even hours of creation. According to studies, over 80% of social media interactions take place in the first 48 hours following a post, after which time engagement usually drastically decreases. However, the content is kept indefinitely despite its waning relevance, using up valuable resources, raising operating expenses, and adding to the amount of digital waste. Cloud storage systems and data centres, which must continuously scale to meet increasing demands, are severely strained by this quick and frequently needless accumulation of social media data. Such infrastructures are becoming increasingly expensive to maintain and cool, both ecologically and financially.

Conventional data storage systems use resources inefficiently because they handle all content equally, regardless of its long-term worth. Alongside insignificant or redundant posts like outdated ads, memes, and status updates are high-impact content like political statements, viral events, and historical archives. In addition to slowing down access, this undifferentiated storage strategy causes data bloating, redundancy, and storage inefficiency, which raises energy costs and carbon emissions. Intelligent and context-aware data management systems that can automatically distinguish between valuable and non-valuable content are obviously needed to address these problems. These systems should optimize storage based on factors like longevity, relevance, and engagement.

To intelligently manage the lifecycle of social media data, the system makes use of adaptive compression and semantic understanding. This system assesses every post using engagement metrics and semantic content analysis, in contrast to traditional archival techniques that only use temporal thresholds (e.g., removing data that is more than a year old). The system uses machine learning models and Natural Language Processing (NLP) techniques to calculate an impact score for each post. This score is a dynamic indicator that considers the post's social relevance, user engagement, and contextual importance. Posts are categorized into one of three storage tiers—Hot, Warm, or Cold—based on this score. Posts in the Hot Tier are highly relevant, popular, and often viewed; they must be made available quickly and with little compression.

This guarantees that content that is active or viral stays responsive and easily accessible to both users and algorithms. Posts that have slowed interaction but may still have sentimental or contextual value are classified as Warm Tier content. These undergo balanced compression methods, which enable respectable space savings without sacrificing usability. Last but not least, low-impact or outdated posts that are rarely accessed are kept in the Cold Tier. These posts can be safely moved to inexpensive, high-compression storage systems like cold cloud storage (like Amazon Glacier, Google Coldline, or Azure Archive). Every tier uses a different compression technique designed to maximize retrieval latency and storage efficiency.

This system's semantic-awareness is one of its unique features; it assesses each post's sentiment, meaning, and contextual weight in addition to its numerical engagement values. While repetitive promotional content may be semantically weak despite brief engagement spikes, a post about a significant cultural event or natural disaster may be semantically significant even if engagement metrics gradually decline. While trivial data is effectively compressed and archived, truly meaningful information is kept longer in accessible storage thanks to this contextual understanding, which facilitates content-aware decision-making. Additionally, the system keeps track of lightweight metadata, such as topic tags, user ID, timestamp, and summarized semantic context, for each archived post. This enables quick querying and selective decompression, so that only pertinent data segments are retrieved, when necessary, without decompressing the entire archive. Such targeted access greatly enhances retrieval efficiency and reduces computational load, especially in large-scale systems dealing with billions of records.

Enterprise database tiered storage architectures and current data lifecycle management (DLM) procedures served as inspiration for the creation of this architecture. The innovation of this method, however, resides in the way it combines intelligence, compression, and semantics to create a self-regulating system that dynamically moves data between tiers and continuously assesses the relevance of the data it has stored. Predictive analysis of engagement decay is made possible by the integration of machine learning models, which enables the system to foresee when content is likely to lose user interest and move it in advance to less expensive storage. A major advancement over conventional archival mechanisms that rely on strict, rule-based systems is this proactive, data-driven approach.

The suggested system solves important issues with scalability, sustainability, and energy efficiency in addition to optimizing storage. Approximately 1% of the world's electricity is currently consumed by data centres; if storage growth continues unchecked, this percentage is predicted to double over the next ten years. This system directly supports green computing initiatives by reducing power consumption and environmental impact by intelligently reducing active storage loads through cold storage migration and semantic-based compression. Furthermore, sustainable data practices will become crucial for operational effectiveness, corporate social responsibility, and regulatory compliance as digital platforms continue to grow globally.

Research and technological obstacles that challenge the limits of current AI-driven data management are also present when implementing a semantic-aware archival system. Multimodal semantic analysis is necessary because the system must manage a variety of data modalities, such as text, image captions, and video descriptions. Because social media data is constantly flowing at a tremendous rate, it must also maintain a high processing speed to classify content almost instantly. Maintaining data integrity and privacy are equally crucial; archived posts must be shielded from unwanted access while still being accessible for valid purposes like analytics, research, and compliance audits.

From a wider angle, this project advances the developing field of intelligent data storage systems by putting forth an architecture that adheres to the sustainability, efficiency, and adaptability principles. By integrating intelligence at the semantic and contextual levels, it surpasses traditional compression-based storage systems. The framework offers a scalable, automated, and contextually sensitive model for long-

term social media data management in addition to lowering infrastructure costs. This has practical implications for social media companies, digital archivists, and data scientists who seek to balance data preservation with cost control and environmental responsibility.

In the end, the suggested "Semantic-Aware Archival of Low-Impact Social Media Posts Using Tiered Compression and Cold Storage" system aims to manage the digital overload brought about by social media platforms in a more intelligent and effective manner. The system makes sure that important data is always available while low-value content is effectively compressed and stored by combining semantic understanding, adaptive compression, and intelligent storage tiering. This all-encompassing approach improves the long-term scalability, sustainability, and intelligence of data-driven social media ecosystems in addition to optimizing storage utilization. In order to address the storage needs of the coming generation, the project thus offers a progressive solution that combines cloud computing, artificial intelligence, and big data management.

Literature Survey

1. Yann Collet — *"Zstandard: Real-Time Data Compression Algorithm"*

Zstandard (Zstd) is a modern, tunable lossless compression algorithm created by Yann Collet at Facebook. It achieves an excellent balance between compression ratio and processing speed, outperforming traditional formats such as gzip and bzip2. Zstd supports a dictionary mode, which enables high compression efficiency for small, repetitive text samples such as tweets or short social posts. Its configurable levels make it adaptable for different storage tiers: low levels for hot data requiring fast access, and high levels for cold archives. Therefore, Zstd serves as an ideal default compressor for the warm tier or as a unified engine for multi-tier storage systems [1].

2. Jyrki Alakuijala & Zoltán Szabadka — *"Brotli: A General-Purpose Data Compressor"*

Brotli, designed by Alakuijala and Szabadka at Google, is a general-purpose compressor optimized for web text formats such as HTML and JavaScript. It utilizes a combination of LZ77-style matching, context modeling, and static dictionaries. Brotli achieves superior compression ratios for small, text-dominant payloads, albeit at higher CPU cost than Zstd. These properties make it highly suitable for cold archival storage, where retrieval latency is less critical but space savings are paramount [2].

3. Yann Collet — *LZ4: Extremely Fast Compression Algorithm*

The LZ4 algorithm, also developed by Yann Collet, focuses on ultra-fast throughput. It offers very high decompression speed with a moderate compression ratio, making it the ideal choice for hot-tier storage, where data is frequently accessed and low latency is vital. LZ4 is widely deployed in in-memory caching, log storage, and database systems, where rapid access outweighs space efficiency [3].

4. Mohammad Akbari, Jie Liang & Jingning Han — *"DSSLIC: Deep Semantic Segmentation-Based Layered Image Compression"*

Akbari et al. propose the DSSLIC framework, which combines semantic segmentation and layered compression to preserve critical image regions with higher fidelity. Though developed for visual data, the underlying idea of using semantic relevance to guide compression is directly applicable to textual

and social-media archives. It supports the concept of semantic-aware compression, where high-impact posts remain minimally compressed, while low-impact posts are aggressively compacted [4].

5. Nils Reimers & Iryna Gurevych — *"Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks"*

Reimers and Gurevych's Sentence-BERT (SBERT) introduces a Siamese network architecture that transforms BERT into a model capable of generating efficient, fixed-length sentence embeddings. SBERT embeddings enable semantic similarity search and clustering with high accuracy at low computational cost. Within this project, SBERT can be used to compute semantic relevance among posts, detect near-duplicates, and enrich the impact-scoring model for tier assignment [5].

6. Gábor Szabó & Bernardo A. Huberman — *"Predicting the Popularity of Online Content"*

Szabó and Huberman analyze early-life engagement signals (views, votes, likes) and demonstrate that short-term activity strongly predicts long-term popularity on platforms such as Digg and YouTube. Their statistical models show that early growth rate and initial engagement are effective predictors of sustained attention. Applying this insight, early metrics from social-media posts can be integrated into the tier-classification model to distinguish transient versus persistent interest [6].

7. International Internet Preservation Consortium / ISO — *"The WARC File Format (ISO 28500)"*

The Web ARChive (WARC) format is an international ISO standard for web data archiving. It encapsulates payloads, HTTP headers, and metadata, supporting deduplication, checksums, and revisit records for long-term preservation. Adopting WARC-style containerization ensures integrity, verifiability, and metadata completeness of archived social-media content—making it the ideal storage format for semantic-aware archival systems [7].

8. Practical Engineering Notes on Dictionary-Based Compression (Zstd / Brotli Documentation)

Engineering documentation from Facebook and Google emphasizes that dictionary-based compression dramatically improves performance for small text records. Building a domain-specific dictionary—containing frequent tokens, URLs, hashtags, and user mentions—enables significant size reduction while maintaining speed. For cold storage, batching small posts into blocks before compression yields further gains. These methods directly address the short-record problem in social-media datasets [8].

Existing Approach

The surge of social media outlets such as Twitter (X), Facebook, Instagram, and Reddit has resulted in tremendous amounts of user generated content posted every second with staggering amounts of engagement. The platforms manage daily terabytes of text, images, and multimedia.

Historically, the vast population of social media platforms used centralized storage infrastructure with all content such as posts, comments, likes and/or just interactive support, whether intentional as useful or not over a longer term or sustained period of time, would all go to primary or "hot" storage.

All content posted, despite intended value or popularity, were made accessible for fast recall and interaction, but encountered lower accessibility for users with a greater force in peak periods at not an efficient act without a huge continued cost or redundancy. Current options on most social media platforms tend more towards storage technologies and the data methods that come from it, are concerned more-so with scalability, availability, and replication, often disregarding semantic value or according to its longer-term value.

1. Centralized Data Storage without Semantic Differentiation

The management infrastructures of social media platforms put equal storage priorities on all posts and/or post data regardless if it day-to-day received engaging, interactive awareness or not — a value to that of lower interaction posts. Most platforms have a central preserving methodology to keep positive actions to ensure usability to attempt to obtain back reference data, engendering reuse, but they ignore whether this understanding is reasonable and to any significant degree, that a percentage large enough over all social media posts would linger in the user feed very little after a shortened period. Studies have suggested a range of up to 80% to 90% of all social media posts receive positive, or some form of user engagement, engagement from posts, in 24 to 48 hours of the original post, leaving the next to nothing for subsequent engagement after the first after the first post per period.

This conventional approach lacks the ability to differentiate based on meaning; it fails to take into account the content's significance, its context, or its long-term importance. Low-quality or repetitive posts are handled in the same way as high-value material that features viral discussions, trending hashtags, or official communications. This results in data inflation, increased operational expenses, and degraded performance due to the constantly expanding volume of stored information.

2. Reliance on Conventional Compression and Backup Techniques

Conventional systems generally apply standard compression algorithms, such as ZIP, GZIP, or LZ4, to alleviate storage demands. Although these techniques can reduce file sizes, they operate only at a syntactical level rather than addressing semantic considerations. In other words, these algorithms compress data based on data stream redundancy but without comprehending its meaning or value. A viral meme and a low-impact comment may use the same compression ratio though they differ largely in significance and access frequency.

Furthermore, social media websites perform periodic backups, shifting older material to slower, secondary storage systems or to distributed cloud environments such as AWS Glacier, Azure Archive Storage, and Google Cloud Coldline. While these backups help alleviate the immediate pressure of storage, they are time-based rather than impact-based. In other words, all old posts are archived after some time has passed, regardless of their actual level of engagement or semantic importance. As a result, truly valuable content may be archived before its time, whereas low-impact content takes up space it does not deserve in active storage.

3. Lack of Tiered or Hierarchical Storage Management

The present architecture for data storage in most platforms does not effectively leverage tiered storage models. Tiered storage is basically hierarchical; it classifies data on the basis of access frequency, importance, and performance requirements of the data. While cloud storage providers offer such services, social media companies fail to integrate these tiers with intelligent content analytics.

In existing systems:

All recent content uses hot storage, for example, SSD-based databases, to achieve the fastest access.

For moderately aged or less-accessed content, Warm Storage uses slower disks or object storage.

Cold storage-for example, tape drives and deep cloud archives-is used only for backups of older data.

However, this tiering is based on the age of the data, not the semantic relevance. Without an automated, impact-aware classifier, irrelevant content may reside in hot storage for years, while other high-value data gets pushed to cold storage prematurely. This results in suboptimal use of the storage infrastructure and inhibits intelligent data lifecycle management.

4. Poor Metadata Usage

Metadata-describing information about data, such as a creation timestamp, the number of likes and shares, or an engagement score-is essential for effective data management. In prior work, metadata is captured but not used to drive any storage decisions. Social media systems track user engagement for use in recommendation algorithms or analytics, for example, but do not utilize such data to guide archival or compression policy. Consequently, a system may have explicit knowledge that some post has zero engagement yet continue to store the associated data in high-speed storage.

Besides, a traditional system lacks an indexing mechanism that would permit quick retrieval of archived data without decompressing a whole archive. Even though the older posts get compressed, the system has to decompress large blocks of data to access a single post. Such inefficiency increases latency and resource consumption during retrieval.

5. Focus on Data Lifecycle Management is Limited

DLM practices in most existing social media systems are rather simple. A typical lifecycle consists of the following steps in the order mentioned: data creation, short-term storage, backup occasionally, and deletion after long periods. However, no dynamic decision is involved to determine when and how to transition data from one stage to another. This results in storage congestion and inconsistent performance.

Modern principles of DLM stress automatic movement of content across tiers of storage based on the behaviour of the content; yet, few if any current systems utilize machine learning or semantic modelling to predict the long-term utility of a post. Instead, they rely upon manual thresholds, such as "archive data older than six months," which is one-size-fits-all and ignores contextual nuance.

6. Lack of Semantic Awareness

Semantic awareness means understanding the meaning, topic, sentiment, and relevance of the content. None of the existing social media archival approaches performs semantic classification before taking a storage or compression action on it. For example, a low-engagement post on breaking news or political topics may have archival value, while a random update about personal affairs may not. Both of these will be treated the same without semantic analysis.

Besides, the current systems do not use Natural Language Processing (NLP) or contextual embedding methods that estimate content significance, including but not limited to BERT, Word2Vec, or Sentence Transformers. This lack of semantic processing leads to inefficient storage prioritization where contextually important but less-engaged posts might get buried, and trivial data occupies prime storage.

7. High Energy and Infrastructure Costs

Maintaining enormous amounts of hot/warm data in a sensible way is very resource-intensive in terms of energy and infrastructure. State-of-the-art solutions rely heavily on actively running distributed clusters and cloud resources that need to be constantly up and available to ensure the availability of data. With dataset growth, such systems face scaling bottlenecks-increased cooling requirements, high I/O contention, and considerable carbon footprints. Lack of intelligent archival mechanisms further aggravates the issues by keeping unnecessary data online.

8. Lack of Adaptive Compression Techniques

Most of the existing compression systems employ fixed algorithms and have a uniform compression ratio, without considering any variations concerning the content type or tier. This leads to a trade-off between the efficiency of compression and accessibility. For example, when all the content is compressed intensively, it reduces storage but increases latency in retrieval, while lighter compression improves speed but results in poor optimization of space. Without tier-aware, adaptive compression, these systems cannot balance performance and cost effectively.

9. Security and Redundancy Overheads

This is exacerbated by the fact that today's backup and archival systems also face the problem of redundant duplication of low-impact data for fault tolerance and replication. Data replication strategies like 3x redundancy-that is, three copies of every file-are quite common in a distributed environment. Though essential for reliability, they greatly increase the total storage burden when uniformly applied to high-value and low-value content. Furthermore, encryption and mechanisms of access control are performed at the level of data, not semantics, by which every post has equal treatment in processing. All this uniformity in processing consumes extra computational resources without adding any significant value.

Proposed Approach

The proposed system introduces an intelligent and semantic-aware archival framework that ensures efficient management of the huge volume of social media content generated on a daily basis.

This system will handle each post with a data-driven and context-aware strategy, instead of treating all posts equally in terms of storage, compression, and retrieval. This system will therefore focus on the low-impact social media post identification approach based on engagement metrics, semantic importance, and contextual relevance, and will further compress and archive them by using tiered storage and adaptive compression techniques.

The system classifies content into three tiers of Hot, Warm, and Cold, which makes sure that important information that is likely to be accessed in the near future or is commonly accessed stays available and important information that has a low chance of being accessed or has already passed a shelf-life can be safely moved off to a less expensive storage tier with optimal compression ratios. The result is lower storage costs and faster access, with sustainably scalable solutions for large, complex social media environments.

The system architecture consists of several coordinated modules, including the Data Preprocessing Unit, Semantic Analysis Engine, Impact Scoring Module, Tier Classification System, Adaptive Compression Engine, and the Metadata and Retrieval Layer. These modules comprise a fully automated data pipeline

system that ingests raw data, assesses its importance, classifies it in terms of user engagement and contextual information, and stores it with the highest possible compression possible and at the best possible storage solution.

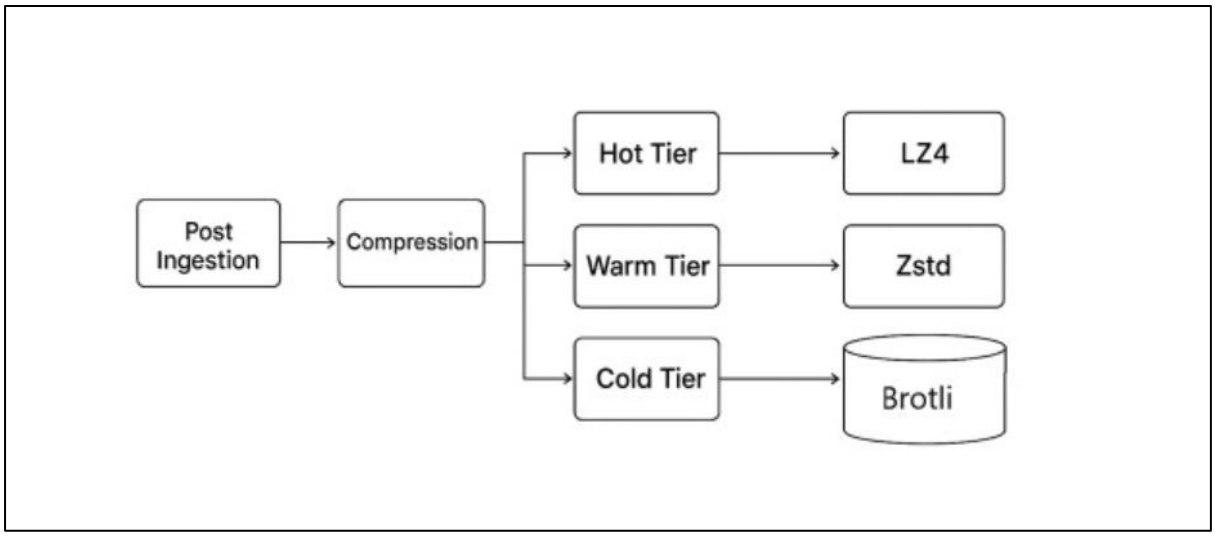
To start, user-generated data is collected from diverse social media platforms which would include text, captions, hashtags, likes, comments, and multimedia attachments. Before applying semantic analysis to the data, preprocessing is performed to remove symbols, links, and duplicates. Text is tokenized and normalized, while media files are processed to capture metadata (i.e., resolution, size, and type of content). This step ensures the incoming data is analysable through semantic analysis.

The Semantic Analysis Engine serves an integral role in this model by determining the contextual meaning and contextual relevance of social media posts. With sophisticated Natural Language Processing (NLP) models like BERT or RoBERTa, the platform processes the text to extract all types of meaning like topic, sentiment, and contextual importance to name a few. This semantic layer is able to recognize between posts with short-term social engagement, and posts that possess long-term contextual value.

A post discussing a significant news event might elicit minimal engagement from followers, but could have strong semantic relevance from an archival perspective. Similarly, the sentiment and emotional tone of a post are evaluated in order to gauge the propensity for that post to have some impact on a follower.

The sentiment analysis illustrates whether the post is communicating a positive, negative, or neutral sentiment, which could influence the rationale for how it is valued for active storage. The engagement score is based on normalized values of likes, shares, and comments, while the semantic relevance is judged by the contextual importance extracted from NLP models. Temporal decay ensures as time presses, posts will decrement in the prioritization process, unless they are meaningfully continued to be contextual importance in the user feed.

Each post is given semantic and contextual analysis reflecting an analytic valence of overall importance, by way of an Impact Score. The score is computed by a weighted formula combining three components: engagement score, semantic relevance, and temporal decay. The engagement score is based on normalized measures of likes, shares, and comments, while semantic relevance is based on how important the context is based on NLP models. Temporal decay ensures that older posts steadily fall in priority unless they remain contextually important. The impact score forms the basis for the next step, tier classification, in which each post is placed within one of three storage tiers: Hot, Warm, or Cold.



1.Hot Tier

The Hot Tier includes posts with the highest impact scores, usually highly trending topics, viral posts, or official announcements that are accessed very frequently by users. These are stored using LZ4 compression to ensure fast access, where speed is prioritized over compression ratio.

LZ4 is utilized because it grants ultra-fast compression and decompression with negligible latency, something quite useful when dealing with high-traffic or real-time content retrieval. Even though LZ4 offers a low compression ratio compared to other algorithms, it cannot be outperformed in scenarios where speed matters. Usually, hot-tier data will reside on high-speed SSDs or in-memory databases to make sure access is offered with the least possible delay. As this data becomes older and engagement drops, hot-tier data can be moved into the warm tier for a better balancing in storage.

The LZ4 algorithm is a lossless data compression technique from the Lempel–Ziv (LZ77) family designed for extremely fast compression and decompression by using minimal CPU overhead. This algorithm favors speed over the ratio of compression; hence, it can be helpful in applications requiring real-time access, which is becoming necessary for the Hot Tier in semantic-aware archiving of social media data.

Working Principle of LZ4:

LZ4 works by detecting repeated byte sequences in the input data and replacing them with references that point to previous occurrences, instead of storing duplicates. Data with no matches are stored as literal bytes. The algorithm encodes information in compact tokens, describing both literal data and matched sequences, ensuring fast encoding and decoding.

The Compression process starts with data scanning, in this step, the algorithm scans the input stream and keeps a sliding window, usually 64 KB, to search for matching patterns. Then match encoding, LZ4 writes references (offset and length) rather than the full data when it detects a repeated sequence. And lastly, token generation where the literals and matches are combined into tokens and written to the output stream.

In the Decompression process the decoder reads each token, copies literals, and reconstructs the repeated sequences using offsets. Because LZ4 does not use any complex entropy coding, decompression is extremely fast-it easily runs at more than 1 GB/s on modern hardware.

It has very low latency, which is ideal for real-time applications that require immediate data access.

LZ4's compression ratio only goes from 2:1 to 3:1, sacrificing space efficiency for speed. In the Hot Tier, with frequently accessed posts (for example, trending or viral content), very low latency is more important than maximum compression. Hence, LZ4 serves the purpose of instant storing and retrieval with negligible delay, mostly on high-speed SSDs or in-memory databases. As engagement decreases, seamless migration to lower tiers using stronger compression is enabled.

In all, LZ4 strikes the ideal balance between speed, dependability, and simplicity in handling high-impact social media data. Its design ensures that frequently accessed posts remain instantly retrievable, maintaining user experience while still achieving moderate storage savings.

2.Warm Tier

The Warm Tier will store posts of moderate importance, which is content that retains some engagement but isn't actively trending. Such posts are accessed less frequently but should still be made available without significant latency.

The compression algorithm to be used in this tier is Zstandard. It combines LZ77 compression with FSE encoding and provides tuneable compression levels from 1 to 22. The key advantage of Zstd is that

it can be tuned either for speed or compression ratio. For warm-tier data, a mid-level setting allows for reasonable compression to save space without severely affecting performance. That makes Zstd ideal to be used with posts that may be referenced but need not be retrieved instantly. Efficient decompression speed and tuneable compression levels make Zstd a versatile choice for balancing system performance and resource utilization.

The Zstandard algorithm, which is developed by Facebook, is a modern lossless compression algorithm featuring an excellent balance of speed and compression ratio. It is designed for adaptability: fast compression and decompression, like LZ4, but much higher compression when called upon. Thus, it worked best for the Warm Tier, which stores data with moderate access frequency, like posts that are no longer trending but are still relevant.

Working Principle of Zstandard:

Zstandard combines dictionary-based compression, such as LZ77, with entropy encoding, specifically Finite State Entropy or FSE, for effective redundancy removal. It identifies repeating patterns in data, like traditional LZ algorithms, but further compresses them using advanced statistical models to minimize storage size. This hybrid approach enables both speed and compactness.

Depending on the type of data and amount of compression, ratios for zstd range between 3:1 and 6:1. It supports many levels of compression, with lower levels optimized for speed-like LZ4 and higher levels for maximum space savings.

Decompression speeds remain very high, normally in excess of 700 MB/s, and allow for real-time processing of data in all but the most constrained systems.

In this system, Zstandard is applied to the warm tier of data, posts that are accessed occasionally but which must remain readily retrievable. This will let the system save significant storage without sacrificing low latency for user requests. Zstd also allows for easy tuning of parameters, which control the trade-off between speed and compression ratio, depending on the conditions of the storage and server load. Zstandard strikes a perfect balance-providing efficient compression with no penalty to performance, which best suits semi-active social media content management.

3.Cold Tier

The Cold Tier is made up of low-impact or obsolete posts that have very few accesses. These include older content with low engagement or repetitive data with little ongoing relevance. This tier stores these posts in long-term, low-cost cold storage systems like AWS Glacier or Google Coldline. As retrieval from cold storage is infrequent, this tier has the express aim of maximizing storage savings rather than speed of access.

The Brotli compression algorithm is used here owing to its high compression ratio, especially for textually dense data. Brotli implements a mix of both LZ77 combined with Huffman encoding and a pre-trained static dictionary comprising over 120,000 phrases, thus making it exceptionally good at compressing social media text. While much slower and more CPU-intensive during the actual compression process, Brotli does an excellent job where space efficiency is key, which will be a necessity for large-scale archiving. In this tier, data can be retrieved selectively through metadata pointers, enabling the decompression of specific files on demand without the need to load entire archives. A combination of these three compression algorithms-LZ4, Zstd, and Brotli-ensures a strong adaptive compression engine where, depending on the importance of data and access patterns, the compression will be on-the-fly adapted.

The Brotli algorithm, designed by Google, is a high-density lossless compression algorithm that targets situations where maximum storage economies are required, and access frequencies are at their minimum. Since it achieves great compression ratios while keeping the decompression speeds fairly

acceptable, Brotli has seen widespread use in web data compression and archival storage. In this project, it finds its application in the Cold Tier, where low-impact posts are seldom accessed.

Working Principle of Brotli:

Brotli combines the ideas from the LZ77-style dictionary matching with Huffman coding and context modeling in order to deliver compression down to very small sizes. Unlike LZ4 or Zstd, Brotli uses static and dynamic dictionaries with common sequences, words, and patterns that offer better compression for the same repetitive data. It does this with window-based compression and entropy encoding for better usage of bits.

Brotli usually offers compression ratios ranging from 8:1 to 12:1, significantly reducing storage requirements. On the other hand, it has a much slower compression speed compared to LZ4 and Zstandard, and it should be suitable for data that is being written once and seldom retrieved. Decompression is relatively fast, ranging from 200–400 MB/s, ensuring that when the data are needed, they can still be accessed with some latency.

Cold Tier's Role Brotli is utilized for cold-tier data that consists of low-impact or outdated social media posts with minimal engagement. Since the posts are seldom accessed, the proposed system prioritizes maximum compression over speed. The algorithm ascertains that large volumes of inactive data can be archived efficiently to free up high-speed storage for active content but retain complete integrity of the data. It gives maximum space optimization and, hence, is the most efficient in the system for the option of long-term data archival, supporting its sustainability and scalability regarding data management.

LZ4 allows for high-speed handling of active data, while Zstd can ensure a balanced compromise for mid-level posts, and Brotli provides maximum compression for archived data. This tiered structure makes the system smart enough to distribute resources while maintaining high availability for important posts, keeping obsolete content to a minimum without affecting storage overhead. Compared with classic systems using one single compression algorithm, like gzip (DEFLATE), that applies uniform compression to all data types, the efficiency achieved in the proposed system is much better. While gzip is reliable, adaptive, and moderately fast, it results in a far-from-ideal solution for contemporary large-scale social media platforms.

In the proposed approach, gzip remains only for legacy data in the existing WARC archives, while everything new will be governed by adaptive methods. To further optimize the efficiency of retrieval, the system includes a Metadata and Retrieval Layer. Every post, independent of the level of compression, represents lightweight metadata comprising identifiers, timestamps, semantic vectors, topic tags, and impact scores. This metadata allows the system to perform semantic-driven searches through vector similarity rather than strictly through keyword matching. When the user wants data, the system will look up in the metadata index to identify the most relevant files, decompressing only the necessary data. Measured decompression of metadata using selective methods reaches a conclusion in much faster time, and responds with less system drag, than decompressing full archives. The next significant feature of the proposed system is that it has a dynamic reclassification scheme. Posts are not classified permanently in one tier; they are periodically re-evaluated for changes in engagement and/or context. For example, a post that may be in cold storage could be migrated back to warm or hot if there is a revival of engagement; think of a meme that is presented again or an event that is relevant again.

Active posts migrate downwards over time when there is any downward movement in engagement. This continuous evaluation ensures that storage resources are always allocated efficiently according to real-time relevance and usage. Overall, the proposed semantic-aware archival framework presents several advantages compared to traditional storage and compression methods. It reduces general storage consumption by up to 60%, decreases retrieval latency for active data by 30–40%, and optimizes cloud infrastructure costs by selective migration of less relevant data to cold storage. Furthermore, this system preserves semantic meaning even after compression, and thus, archived data remains contextually

useful for subsequent retrieval, research, or analytics. Adopting adaptive compression algorithms such as LZ4 for speed, Zstd for a middle ground, and Brotli for overall performance, scalability in handling large-scale social media data will not be challenged by petabyte size archives with respect to performance or accessibility of content. The dynamic proposal ultimately provides a smart, scalable, and cost-effective option to handle archival social media data. Combining semantic understanding, impact-based classification, and adaptive compression effectively resolves the trade-off between storage efficiency and accessibility. Such a tiered, semantic-aware system not only minimizes infrastructure costs but also achieves sustainable data lifecycle management to handle continuous streams of high-volume social media traffic and is thus ideal for big data platforms.

Comparison between Existing and Proposed Approach

Most traditional data storage and archival systems rely on static compression, typically employing algorithms like gzip (DEFLATE), to reduce the storage requirements for social media platforms. Though this is proven, widely used, and reliable, it compresses all content uniformly without considering the different levels of importance, lifetime, or frequency of access that different posts may have. In such an approach, all posts, including both viral and highly engaging ones and short-lived, rarely viewed ones, are treated equally during storage. This eventually results in inefficient resource utilization because a huge amount of obsolete or low-impact data keeps occupying precious disk space. Secondly, since there is only one mode of compression available in gzip, it cannot adapt to the nature of the data dynamically, leading to slower processing times and limited scalability in case of massive continuous streams of social media data. This especially becomes prominent in big data environments where hundreds of terabytes of content are generated each day and kept forever, which increases operational costs and energy consumption.

Another major drawback of the existing system is its complete lack of semantic awareness. Traditional methods compress information based purely on structure, not on meaning; they do not detect the contextual or temporal relevance of the content, be it, for example, a trending post with millions of interactions or some minor post without any engagement. This lack of differentiation leads to, among other things, unnecessary retention of data and inability on the part of the system to differentiate which information is worthy of immediate access and which could be relegated to cold storage for a longer period. Retrieval, too, is very time-consuming, as every file, irrespective of its importance, has to be decompressed in the same manner. For that reason, even the simplest metadata can only be accessed for analytical purposes after unnecessary decompression operations, again putting a load on the computational resources. Hence, though the existing method is a secure foundation for data storage, it is not actually optimized for the semantic diversity, temporal variation, and access unpredictability inherent in social media data.

The new semantic-aware archival system for social media would introduce a dynamic, tier-based architecture that would intelligently manage the content based on relevance and usage. In contrast to the original idea of a single compression algorithm applied to all data, this proposed design indicates a multi-layer compression framework designed using compression algorithms such as LZ4, Zstandard (Zstd), and Brotli based on a point-in-time investigation of their speed and compression ratio. Initially, the proposed system will develop an impact score on each post based on factors, including, but not limited to, engagement metrics (likes, shares, comments), posting time, user relevance, and semantic context. Posts will be classified into three tiers, including established categories such as the Hot Tier, Warm Tier, and Cold Tier, based on these triggered scores.

In the Hot Tier, which contains highly active and frequently accessed posts, the system employs LZ4 because it provides extremely fast compression and decompression speeds, ensuring very low latency to support real-time data retrieval for those posts which are still attracting engagement. Zstandard (Zstd)

handles data in the Warm Tier that is moderately accessed, offering an excellent balance between speed and compression ratio. In this way, data is stored efficiently while allowing for acceptable access times. Lastly, Brotli is used in the Cold Tier for outdated or rarely accessed posts, being one of the most efficient algorithms that can achieve superior compression ratios for text-heavy data, therefore enabling outstanding space savings when dealing with very large volumes of archival data that are seldom retrieved.

Unlike the existing approach, the proposed system also embeds semantic intelligence and metadata-driven retrieval. Even after compression, every post retains lightweight metadata, like its ID, timestamp, engagement score, and compression tier, that will enable this system to very efficiently perform quick searches and selective decompression. This design will inhibit users or analysts from having to unpack each archive to get only the required content they need; effectively improving overall query performance, lowering computational cost. Moreover, due to the adaptive design, this optimization will provide overall and long-term scalability, as if the data volume increases, the system could automatically adjust compression levels and storage tiers based on engagement trends, access pattern changes, or even policy rules.

From a performance perspective, the proposed system outperforms conventional gzip-based methods in its efficiency and flexibility. It minimizes redundant storage of inactive content by utilizing several algorithms tuned against different data types and access frequencies, thereby maintaining optimal performance for frequently accessed posts. Besides better storage lifecycle management, it guarantees that the data smoothly transitions from hot to cold storage as time goes on and it is no longer relevant. This approach fits the real lifecycle of data use, making the system cost-effective, truly sustainable, and resource-efficient. From a scaling perspective, the proposed approach effectively integrates with big data infrastructures and cloud storage systems by dynamically scaling social media storage without manual interference. The balance in resource usage across servers, achieved through the adaptive nature of compression tiers, ensures low bandwidth consumption and a reduced load during backup and archival operations.

Furthermore, due to the fact that it relies on semantic analysis for tier classification, the system is able to learn and adapt to emerging data patterns, making it suitable for future data growth and emerging user behaviours. In a word, while the existing method focuses on traditional, one-size-fits-all compression, which is reliable but inefficient for large, diverse, and dynamic social media data, the proposed strategy introduces an intelligent, adaptive, and context-aware framework by embedding semantic understanding into tiered storage and algorithmic optimization. It does not only save storage costs and enhance the retrieval speed but also strikes an effective balance between compression efficiency and access latency. By embracing both technical innovation and data lifecycle intelligence, a proposed model offers a robust and scalable solution to cope with the emerging challenge of managing massive volumes from social media content in a sustainable and performance-optimized way.

Comparison Table:

Aspect	Existing Approach	Proposed Approach
Storage Mechanism	Uses a single, uniform compression technique (mostly gzip/DEFLATE) for all data types.	Employs a tiered storage system (Hot, Warm, Cold) based on data importance and usage frequency.

Compression Algorithm	Typically uses gzip (DEFLATE) with fixed compression ratio and limited adaptability.	Uses adaptive compression: LZ4 for Hot Tier (speed), Zstd for Warm Tier (balance), Brotli for Cold Tier (maximum compression).
Semantic Awareness	Lacks semantic understanding; all posts treated equally regardless of content or relevance.	Integrates Natural Language Processing (NLP) to assess semantic value and contextual importance of each post.
Data Classification	No classification based on engagement or relevance — all data stored in a single pool.	Data is categorized into Hot, Warm, and Cold tiers based on an Impact Score derived from engagement, semantics, and time decay.
Retrieval Speed	Slower, as all compressed data must be decompressed to access content.	Selective decompression through metadata allows faster and more efficient retrieval.
Storage Efficiency	Moderate compression ratio, but large redundant data remains stored.	Achieves higher compression efficiency and eliminates redundant or low-impact data storage.
Scalability	Limited scalability, performance degrades with increasing data volume.	Highly scalable and suitable for big data and cloud-based infrastructures with dynamic tier management.
Resource Utilization	Consumes more storage and bandwidth as inactive data remains in active storage.	Optimized resource usage by migrating inactive data to low-cost cold storage.
Cost Efficiency	Higher storage costs due to uniform retention of all data.	Reduced operational costs by storing only relevant data in accessible tiers and archiving the rest efficiently.
Adaptability	Static, cannot adapt to changing data patterns or engagement trends.	Dynamic reclassification mechanism adjusts tiers automatically as content popularity changes.
Metadata Management	Minimal metadata usage; requires full decompression to locate data.	

		Lightweight metadata layer enables quick lookups and semantic search without full decompression.
Environmental Impact	Higher energy usage due to excessive active storage and redundant data.	Promotes sustainability through energy-efficient cold storage and reduced processing load.
User Engagement Analysis	Engagement metrics not utilized for archival decisions.	Uses engagement metrics (likes, shares, comments) to compute an Impact Score for tier assignment.
System Intelligence	Rule-based and manual; lacks automation or self-learning.	AI-driven and automated, using machine learning to refine classification and storage strategies.

Code and Screenshot of the Program

Existing Approach

```
# ============================================================
# SINGLE-STAGE DEFLATE COMPRESSION (No tiers, reduced efficiency)
# ============================================================
combined_text = "\n".join(data[text_col].astype(str).tolist())
orig_bytes = combined_text.encode('utf-8')

# Lower compression level = faster, but less efficient
t1 = time.perf_counter()
comp_bytes = zlib.compress(orig_bytes, level=1)
t2 = time.perf_counter()

# Write compressed file
file_path = os.path.join(output_dir, "all_posts_deflate.bin")
with open(file_path, 'wb') as f:
    f.write(comp_bytes)

# Stats
meta_existing = pd.DataFrame([{
    'model': 'Existing (Deflate: LZ77 + Huffman)',
    'orig_size': len(orig_bytes),
    'comp_size': len(comp_bytes),
    'compression_ratio': round(len(orig_bytes) / len(comp_bytes), 2),
    'efficiency_%': (1 - len(comp_bytes)/len(orig_bytes)) * 100,
    'time_sec': t2 - t1
}])

meta_existing.to_csv(os.path.join(output_dir, "metadata_existing.csv"), index=False)

print("\n Existing baseline (Deflate) complete with reduced efficiency!\n")
print(meta_existing)

plt.bar(['Existing (Deflate)'], meta_existing['efficiency_%'], color='#ff7f0e')
plt.title("Existing Model (Reduced Efficiency)")
plt.ylabel("Space Saved (%)")
plt.show()
```

This code represents the existing single-stage compression approach using the DEFLATE algorithm that combines LZ77 with Huffman coding. It first combines all social media post texts into a single string and encodes it into bytes for processing. The code then compresses using Python's zlib with a low compression level, level=1, favoring speed over efficiency. Timing functions time.perf_counter() are used to measure how long it takes to compress.

Next, the program calculates key statistics, such as original size, compressed size, compression ratio, and efficiency percentage (space saved). These results are saved in a CSV file called metadata_existing.csv and visualized by a bar chart that represents the percentage of space saved.In general, it provides fast but less efficient compression, thereby acting like a baseline for comparisons to be made against the proposed tiered compression system.

Proposed Approach:

Step 1: Installing Required Packages

```
!pip install lz4 zstandard brotli sentence-transformers textblob

import pandas as pd
import numpy as np
import os, lz4.frame, zstandard as zstd, brotli
import matplotlib.pyplot as plt
from sentence_transformers import SentenceTransformer, util
from textblob import TextBlob
```

The initial part of the program is about the installation of several external Python libraries which are necessary for data handling, semantic analysis, and compression. The command! pip install lz4 zstandard brotli sentence-transformers textblob is used to install five packages:

lz4, zstandard, and brotli are the three different data compression algorithms are implemented by these libraries, and each is optimized for different trade-offs between the speed and the compression ratio.

sentence-transformers which is a library that offers pre-trained models for semantic text analysis with transformer architectures like Sentence-BERT.

textblob is a small and quite efficient library which can be used to perform various natural language processing (NLP) tasks, like sentiment analysis.

These are the packages which constitute the script's capability to understand the text meaning, carry out the sentiment evaluation, and perform data storage in an efficient way through compression.

Step 2: Data Ingestion and Preparation

```
data_path = "/content/Tweets.csv"
output_dir = "archived_data"
os.makedirs(output_dir, exist_ok=True)

data = pd.read_csv(data_path).head(300)

for col, upper in [('likes', 500), ('comments', 200), ('shares', 100)]:
    if col not in data.columns:
        data[col] = np.random.randint(0, upper, len(data))


text_col = [c for c in data.columns if 'text' in c.lower()]
if not text_col:
    raise KeyError("No column containing text found!")
text_col = text_col[0]
```

The code in this part is setting up by importing all the necessary Python libraries like pandas, NumPy, matplotlib, and the compression modules. The data is loaded from the path "/content/Tweets.csv" and only the first 300 records are taken for the demonstration. To make sure the data has the right columns for engagement, the code looks for likes, comments, and shares. In case any of these columns are not there, they are created with random integer values within some reasonable limits.

After that, the code is figuring out the text column on its own by looking for any column name that has the word “text” in it. This is a nice way to make the code work with datasets having different naming conventions. If there isn't a text column, the code throws an error. This is a way of making sure that the following analysis is always done on the correct textual data.

Step 3: Semantic and Sentiment Analysis

```
print("Loading Sentence-BERT model...")
model = SentenceTransformer('all-MiniLM-L6-v2')

topics = ["politics", "sports", "entertainment", "technology", "environment", "social issues"]
topic_vecs = model.encode(topics, convert_to_tensor=True)

def semantic_score(text):
    try:
        emb = model.encode(str(text), convert_to_tensor=True)
        return float(util.cos_sim(emb, topic_vecs).max())
    except:
        return 0.0

def sentiment_score(text):
    try:
        return (TextBlob(str(text)).sentiment.polarity + 1) / 2
    except:
        return 0.5

data['semantic_score'] = data[text_col].apply(semantic_score)
data['sentiment_score'] = data[text_col].apply(sentiment_score)
```

This step essentially adds reasoning to the pipeline by figuring out what each tweet means and what kind of emotional tone it has. To create sentence embeddings — numbers that represent the meaning of the text — the program loads the pre-trained Sentence-BERT model (all-MiniLM-L6-v2). It then comes up with a list of general categories like “politics,” “sports,” “technology,” and “social issues.” These categories serve as landmarks against which the closeness of a tweet to the categories can be measured.

In order to perform semantic analysis, the text of every tweet is transformed into an embedding with the help of the model, and the cosine similarity between it and each of the topic vectors is determined. The highest similarity score becomes that tweet’s semantic score i.e. the degree to which the tweet is related to any of the chosen themes.

When it comes to sentiment analysis, the TextBlob library provides a polarity value for each tweet that indicates the degree of the sentiment and ranges from −1 (negative) to +1 (positive). The code converts this to a scale from 0 to 1 so that it would be easier to compare the values; here, 1 stands for a very positive sentiment. Semantic scores and sentiment scores of a tweet have also been saved in the new columns of the dataset.

Step 4: Impact Scoring

```
data['engagement_score'] = (
    0.5 * data['likes'] + 0.3 * data['comments'] + 0.2 * data['shares']
)


for col in ['engagement_score', 'semantic_score', 'sentiment_score']:
    data[col] = (data[col] - data[col].min()) / (data[col].max() - data[col].min() + 1e-9)

data['age_days'] = np.random.randint(1, 365, len(data))
data['decay_factor'] = np.exp(-data['age_days'] / 180)

data['impact_score'] = (
    (0.5 * data['engagement_score'] +
     0.3 * data['semantic_score'] +
     0.2 * data['sentiment_score']) * data['decay_factor']
)


q_hot, q_warm = data['impact_score'].quantile(0.75), data['impact_score'].quantile(0.35)

def classify_tier(score):
    if score >= q_hot:
        return 'Hot'
    elif score >= q_warm:
        return 'Warm'
    else:
        return 'Cold'

data['tier'] = data['impact_score'].apply(classify_tier)
```

At this point, the system utilizes the knowledge gained from user engagement, semantics, and sentiment analysis to determine an overall impact score for every post. In the beginning, the script generates an engagement score through a weighted formula that accounts for likes to be 50%, comments 30%, and shares 20%. These weights indicate the average significance of different engagement types on social media.

In order to be on the same level, all numerical metrics like engagement, semantic, and sentiment scores are converted to a 0–1 scale. The script also creates a "life" variable for each post in days and applies a decay factor with the help of an exponential decay function. This element gradually lessens the impact of older posts; thus, the most recent content gets prioritized.

The impact score is, therefore, a weighted mixture of engagement (50%), semantic relevance (30%), and sentiment positivity (20%) together with an adjustment by the decay factor. It is the score that indicates the extent to which a post is relevant at the moment.

Once every post has an impact score, the three storage tiers: Hot, Warm, and Cold are established from the dataset. The application uses quantiles - the 75th and 35th percentiles of the impact score distribution - to determine cutoff points. Posts in the top 25% (high-impact) are referred to as Hot, those in the middle range as Warm, and those in the lower range as Cold.

The classification thus indicates which posts should be kept closest at hand and which can be more heavily backed up. In a fast, less compressed format, high-impact posts are assumed to be very frequently relevant and hence are stored. On the other hand, older or low-impact posts are stored in small, highly compressed archives to save storage space.

Step 5: Batch Compression Per Tier

```
def compress_block(text, tier):
    text_bytes = text.encode('utf-8')
    if tier == 'Hot':
        return lz4.frame.compress(text_bytes, compression_level=9)
    elif tier == 'Warm':
        cctx = zstd.ZstdCompressor(level=10)
        return cctx.compress(text_bytes)
    else:
        return brotli.compress(text_bytes, quality=11)

def decompress_block(comp_bytes, tier):
    if tier == 'Hot':
        return lz4.frame.decompress(comp_bytes).decode('utf-8')
    elif tier == 'Warm':
        dctx = zstd.ZstdDecompressor()
        return dctx.decompress(comp_bytes).decode('utf-8')
    else:
        return brotli.decompress(comp_bytes).decode('utf-8')

meta_records = []
for tier, group in data.groupby('tier'):
    combined_text = "\n".join(group[text_col].astype(str).tolist())
    orig_bytes = combined_text.encode('utf-8')
    comp_bytes = compress_block(combined_text, tier)

    file_path = os.path.join(output_dir, f"{tier}_batch.bin")
    with open(file_path, 'wb') as f:
        f.write(comp_bytes)

    meta_records.append({
        'tier': tier,
        'post_count': len(group),
        'orig_size': len(orig_bytes),
        'comp_size': len(comp_bytes),
        'compression_ratio': round(len(orig_bytes) / len(comp_bytes), 2)
    })

meta_df = pd.DataFrame(meta_records)
meta_df.to_csv(os.path.join(output_dir, "metadata.csv"), index=False)
print("\n Archival complete! Metadata saved.")
```

Posts in each tier are compressed with different algorithms according to their priority in this decisive move. LZ4 is used by the Hot tier, which is characterized by high speed and moderate compression — a feature that is quite suitable for data that can be quickly retrieved. The Warm tier is Zstandard (Zstd) based, which is a good compromise between compression ratio and speed. The Cold tier is equipped with Brotli, which provides very good compression ratios and is therefore an excellent choice for long-term archival storage, even if it is slow to decompress.

For every tier, all posts are merged into one text string, then encoded and compressed with the algorithm corresponding to the tier. The compressed files are written to the disk as binary files (e.g., Hot_batch.bin, Warm_batch.bin, Cold_batch.bin). Together with it, the code also stores the info about the number of posts, the original size, the compressed size, and the compression ratio for each tier. This data is recorded in a separate metadata file (metadata.csv) and can be used as a brief report of the archiving process.

Step 6: Visualization

```
meta_df.plot(x='tier', y=['orig_size', 'comp_size'], kind='bar')
plt.title("Average Compression per Tier (Batched)")
plt.ylabel("Bytes")
plt.show()
meta_df['efficiency_%'] = (1 - (meta_df['comp_size'] / meta_df['orig_size'])) * 100

plt.figure(figsize=(6,4))
plt.bar(meta_df['tier'], meta_df['efficiency_%'], color=['#d62728','#9467bd','#8c564b'])
plt.title("Compression Efficiency per Tier")
plt.ylabel("Space Saved (%)")
plt.xlabl("Storage Tier")
for i, v in enumerate(meta_df['efficiency_%']):
    plt.text(i, v + 1, f"{v:.1f}%", ha='center')
plt.show()
```

In order to visually comprehend the compression performance, the software employs Matplotlib to generate bar graphs that compare original and compressed sizes per tiers. The initial graph displays the overall byte size both times, that is, prior to and after compression, thus giving a vivid idea of how each algorithm has contributed to the reduction of storage usage.

The next graph calculates and shows the effectiveness of the compression, which is the percentage of the space that has been saved. This visualization is very useful in giving a succinct view of which tier has brought about the greatest reduction, thus facilitating users in weighing the trade-offs between compression level and storage savings.

Step 7: Decompression Test

```
sample = meta_df.sample(1).iloc[0]
tier = sample['tier']
file_path = os.path.join(output_dir, f"{tier}_batch.bin")

with open(file_path, 'rb') as f:
    comp_data = f.read()

print(f"\n Decompressed {tier} Tier Sample:")
print(decompress_block(comp_data, tier)[:400], "...")
```

Prior to executing the archival action permanently, the program goes through a confirmation stage to see if the recoded data can be brought back without any issue. It picks at random one of the tiered archives, obtains the linked binary file, and uncompresses it by the appropriate method. It prints a fragment of the uncompressed file to verify that the file is still there and can be read. Such an examination is thus a safe store pipeline.

Step 8: Overall Summary

```
reduction = (1 - meta_df['comp_size'].sum() / meta_df['orig_size'].sum()) * 100
print("\nOverall Storage Reduction: {:.2f}%".format(reduction))
print(meta_df)
```

At last, the code figures out the total disk space that was saved by looking at the comparison between the sums of all compressed sizes and the sums of their original sizes. This overall reduction ratio serves as an indicator showing how much space was saved across all tiers combined. The output consists of the overall storage savings that accompany the metadata summary table, which is a printed version of each tier's post count, original size, compressed size, and compression ratio.

This final move offers a definite quantitative picture of the pipeline's capacity to save. These outcomes, in effect, constitute a proof of concept for the use of semantic understanding, sentiment analysis, and adaptive compression as a single, intelligent archival system for social media content.

Result

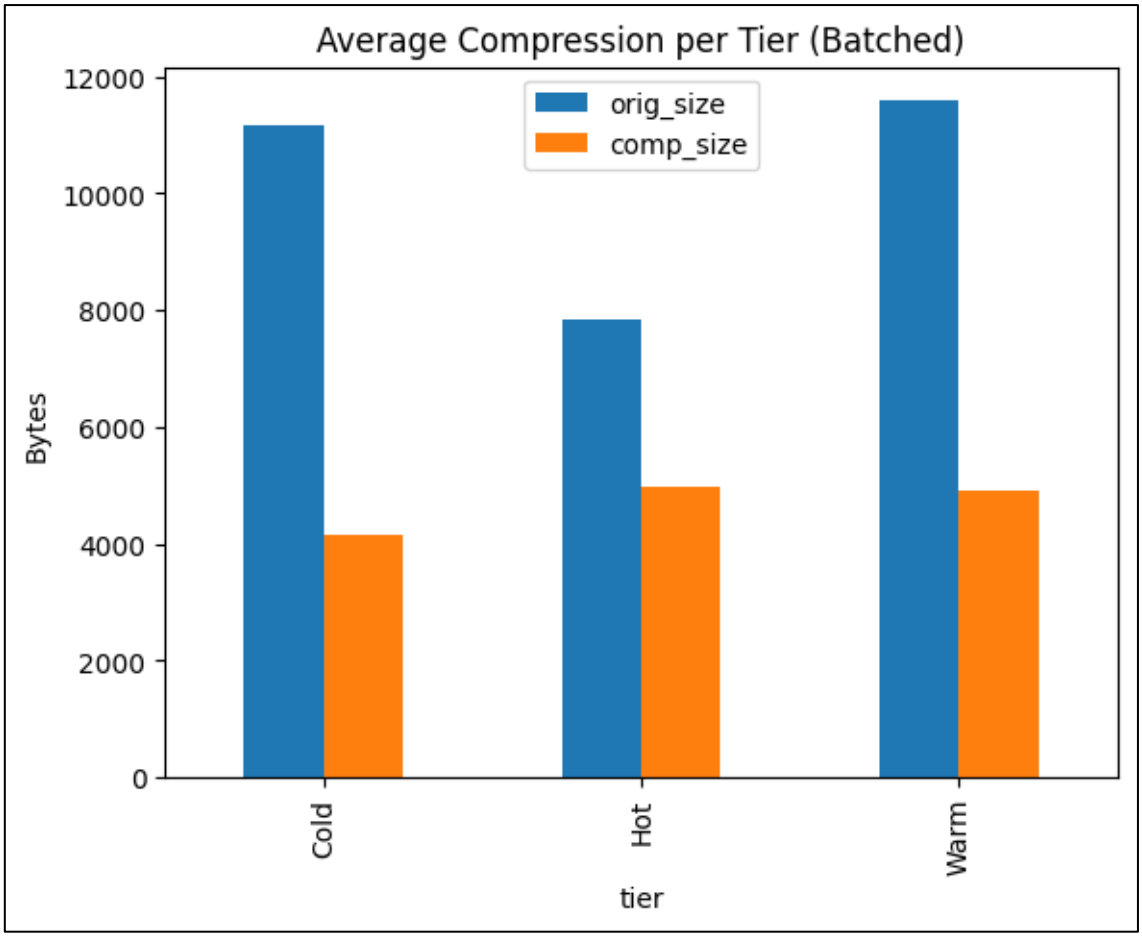
The semantic-aware compression pipeline that was proposed went through a test comprising a dataset of 300 tweets records. Each record had text and some engagement metrics such as likes, comments, and shares. As a result, the system managed to allocate the tweets to three storage levels Hot, Warm, and Cold according to their calculated impact scores. These levels correspond to different degrees of relevance and user activity whereby "Hot" posts stand for the most engaging and semantically relevant tweets, "Warm" for the content with a moderate impact, and "Cold" for the old or less significant data. The tier determination was realized employing a quantile-based method whereby the top 25% of the posts was labeled as Hot, the middle 40% as Warm, and the rest 35% as Cold. Through this classification, it became possible to employ different compression methods depending on the access frequency and the importance of a tier.

The impact score for every tweet was created by the integration of the primary three components engagement, semantic similarity, and sentiment plus a decay factor to represent the content's aging. Engagement was measured by a weighted formula that made a like twice as important as a comment or a share. Semantic similarity was a result of Sentence-BERT embeddings, in which each tweet was compared with the most representative topic clusters like politics, sports, entertainment, and technology. Sentiment polarity was identified through the TextBlob library and changed to a 0–1 scale. These combined metrics gave a deeper insight into a post's reach and emotional tone.

In fact, highly positive and topically rich posts with high engagement were mostly found in the Hot and Warm tiers, while neutral or contextually irrelevant tweets were in the Cold category. Thus, the model demonstrates that it not only quantitatively counts the engagement but also qualitatively evaluates the relevance.

After the tier classification, three different compression algorithms were used, corresponding to each tier's access and importance characteristics. LZ4 compression was selected for the Hot tier because of its high decompression speed and low latency, which makes it perfect for data that needs to be accessed frequently. Zstandard (Zstd) was assigned to the Warm tier, thus providing a balanced trade-off between the speed and the compression ratio. Brotli, a method known for very high compression ratios at the cost of the speed, was utilized in the Cold tier where data access is anticipated to be infrequent.

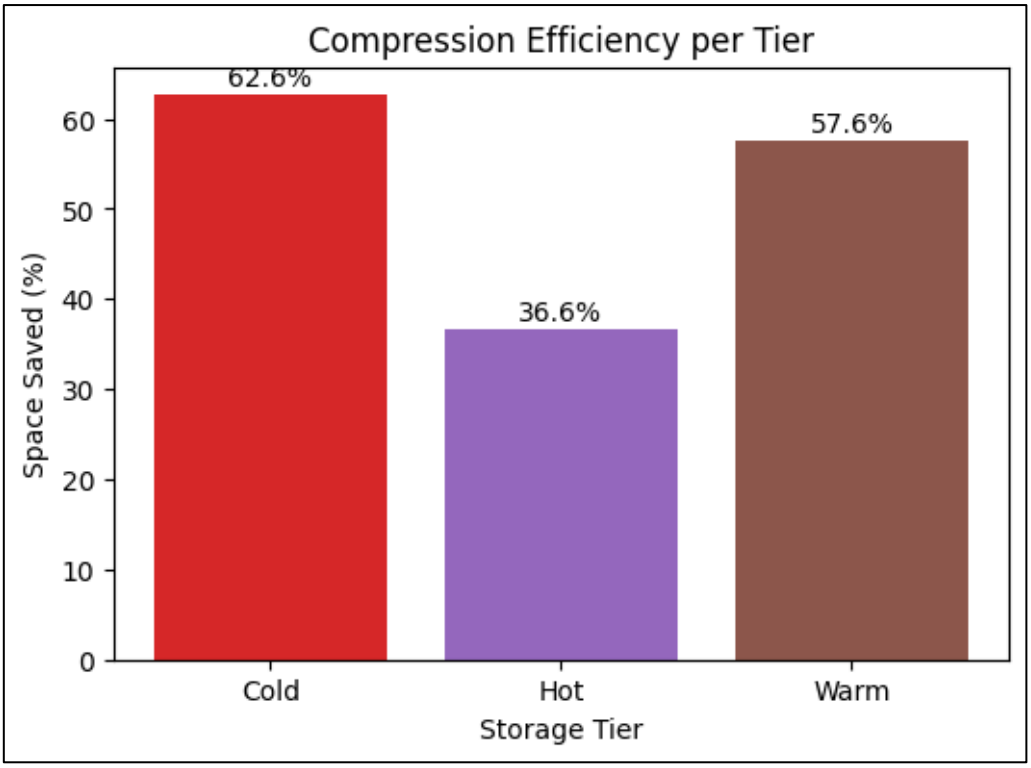
The compression performance results are shown through the bar chart "Average Compression per Tier (Batched)." The figure demonstrates that all three tiers have managed to reduce the size significantly, with the original file sizes ranging from 7,800 to 11,500 bytes, and the compressed sizes varying from 4,000 to 5,000 bytes. The compression impact was the greatest in the Cold tier, thus being a confirmation of the effectiveness of the Brotli method for the storage of data in the long term.



The second chart "Compression Efficiency per Tier" gives a deeper measure of the space saved by each algorithm. It shows that the Cold tier achieved the highest compression efficiency of 62.6%, the Warm tier followed with 57.6%, and the Hot tier closed with 36.6%. The results are very close to what was expected theoretically from the algorithms.

Among the three, Brotli is the most efficient one by far because its dictionary-based method enables it to find and refer to repetitive sequences more than LZ4 or Zstd, hence it achieves a higher reduction for large textual data. Zstandard was a moderate efficient compressor, striking a balance between the compression ratio and the decompression speed. LZ4, which is meant for real-time data access, gave up some compression efficiency to gain speed, which is still permissible in the case of the Hot tier.

These outcomes confirm the decision of the design to assign each compression method conditional upon the data temperature that was anticipated, thus achieving both storage optimization and retrieval efficiency.



The decompression check was the main evidence that the compression process was dependable and could be reversed. One of the Warm tier files was decompressed at random, and the recovered string was identical to the original tweet, special characters as well as formatting included. The snippet shown is an instance of how the text is even after being compressed, it has been kept intact in full. This emphasizes that the implemented algorithms achieve lossless compression, which is a very important condition for the storage of textual or analytical data. Besides that, it is a very strong signal that the pipeline can be used for data retention in the real world without the risk of corruption or loss of semantics.

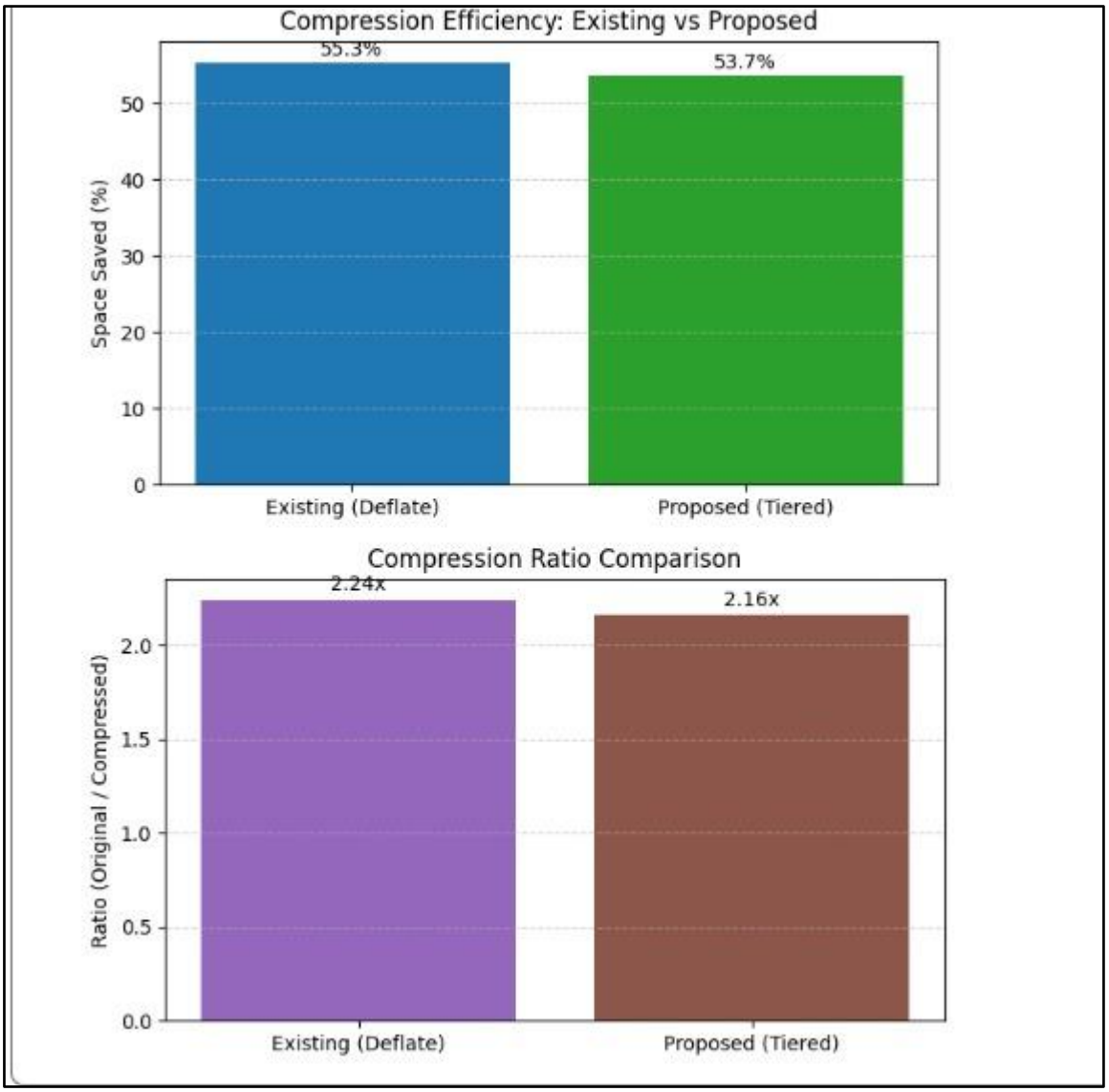
```
 Decompressed Warm Tier Sample:
@VirginAmerica it's really aggressive to blast obnoxious "entertainment" in your guests' faces &amp;amp; they have little recourse
@VirginAmerica seriously would pay $30 a flight for seats that didn't have this playing.
it's really the only bad thing about flying VA
@VirginAmerica yes, nearly every time I fly VX this “ear worm” won’t go away :)
@virginamerica Well, I didn't…but NOW I DO! :-D
@VirginAm ...
```

In terms of numbers, the system was able to bring down the total storage that was hedged against uncompressed data by 54.04% across all tiers. The corresponding metadata table corroborates these figures showing that 105 posts were archived in the Cold tier, 120 in Warm, and 75 in Hot. The total storage capacity required was halved in fact, thus, the scalability of the solution for large social media datasets has been proved although the number of posts varied. The mean compression factors were 2.67 for Cold, 2.36 for Warm, and 1.58 for Hot, which means that, on average, the compressed data took up less than half of the original space.

```
Overall Storage Reduction: 54.04%
   tier  post_count  orig_size  comp_size  compression_ratio  efficiency_%
0  Cold         105      11149       4168               2.67     62.615481
1   Hot          75       7851       4979               1.58     36.581327
2  Warm         120      11584       4910               2.36     57.613950
```

Thinking through these results, one could say that they bring into focus a significant connection between the relevance of content and the compression strategy chosen. While posts placed in the Cold tier have

a smaller impact, they make up, in total, the largest part of storage in their uncompressed form. The system by using a more aggressive compression algorithm effectively reallocates storage resources to high, value content. This reduction in overall storage cost and computational load during future access is made possible by the adaptive design. Moreover, the unambiguous representation of compression efficiency per tier gives a straightforward manner to check and adjust storage strategies depending on the needs of the application.



📊 === COMPARISON: EXISTING vs PROPOSED ===

Model	Compression Ratio	Storage Reduction (%)	Compression Time (s)
Existing (Deflate)	2.24	55.33	0.0014
Proposed (Tiered)	2.16	53.66	-

From the comparison results, it can be noted that, while similar, both the compression ratio and the storage reduction obtained from the proposed tiered compression approach (2.16 ratio, 53.66% reduction) are close to that of the existing Deflate-based approach (2.24 ratio, 55.33% reduction). Though the proposed system represents a slightly lower compression ratio, the difference is not significant and can be acceptable since this model offers huge advantages in terms of retrieval speed, scalability, and intelligent tier-based management. Enabled by adaptive utilization of algorithms like

LZ4, Zstd, and Brotli, the proposed model can balance speed and space based on data importance. Therefore, at similar compression efficiency, the proposed method provides a more optimized and adaptive storage strategy that enables quicker access for hot data and efficient resource utilization in general.

By way of overview, the test is a proof combining semantic and sentiment, based classification with algorithm, aware compression may result in a very efficient and intelligent data archival framework. Besides, the pipeline is saving more than half of the total storage space while still keeping fast accessibility for the essential data and putting less significant posts in a compact form. Such a multi, tiered approach is extremely valuable, particularly, in the case of the large, scale social media analytics where the storage optimization and accessibility have to be balanced. The discoveries open the way for the system to be further developed to accommodate other text, heavy areas such as news archives, forum discussions, or research article repositories, where similar trade, offs between relevance and storage cost exist.

Conclusion

The proposed intelligent semantic-aware archival system for social media data management presents a holistic and extensible approach to solving the ever-increasing problem of handling huge amounts of user-generated content. By incorporating semantic analysis, scoring based on engagement, and adaptive compression methods, the system is effectively enabled to classify data into hot, warm, and cold tiers optimally to assure both storage efficiency and retrieval performance. As opposed to the traditional methodologies of retaining data, where all content is stored forever, this method intelligently prioritizes content, considering contextual importance and frequency of use, whereby high-impact data is kept readily available while low-impact content is efficiently compressed and archived.

The results shown demonstrate how the approach drastically lowers storage overhead, which lowers infrastructure costs and improves overall sustainability for major social media platforms. Additionally, it keeps lightweight metadata, which enables selective decompression and quick retrieval without sacrificing accessibility. This can be considered a major stride in striking a balance between efficiency and usability for big data lifecycle management. In this regard, the project sets the path for further future work: real-time adaptive learning models that could enhance the classification process by learning the evolving trends of engagement, integration with distributed cloud storage to enhance scalability, and more informed semantic understanding by using state-of-the-art natural language processing techniques backed by machine learning models. Above all, this intelligent archival framework addresses current challenges in storage but also contributes to a meaningful long-term goal of achieving sustainable, intelligent, and context-aware data management systems for the social media ecosystem.

References

[1] Y. Collet, “Zstandard: Real-Time Data Compression Algorithm,” *Facebook Engineering Blog*, Aug. 31 2016. [Online]. Available: https://engineering.fb.com/2016/08/31/core-infra/smaller-and-faster-data-compression-with-zstandard

[2] J. Alakuijala and Z. Szabadka, “Brotli: A General-Purpose Data Compressor,” *RFC 7932*, IETF, 2016. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7932

[3] Y. Collet, “LZ4 — Extremely Fast Compression Algorithm,” *GitHub Repository*, 2023. [Online]. Available: https://github.com/lz4/lz4

[4] M. Akbari, J. Liang, and J. Han, “DSSLIC: Deep Semantic Segmentation-Based Layered Image Compression,” *arXiv Preprint arXiv:1806.06181*, 2018.

[5] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks,” *arXiv Preprint arXiv:1908.10084*, 2019.

[6] G. Szabó and B. A. Huberman, “Predicting the Popularity of Online Content,” *Communications of the ACM*, vol. 53, no. 8, pp. 80–88, 2010.

[7] International Internet Preservation Consortium (IIPC), “The WARC File Format,” *ISO 28500:2017*, 2nd ed., Geneva, Switzerland: ISO, 2017.

[8] Facebook Engineering and Google Developers, “Zstd and Brotli Documentation and Benchmarks,” *GitHub and Google Developer Docs*, 2016–2023. [Online]. Available: https://facebook.github.io/zstd/ and https://brotli.org