Journal of Harbin Engineering University
ISSN: 1006-7043

Vol 46 No. 7
July 2025

Application of K-Means Clustering on Apache Flume-based Multimedia Data Collection System

Mr. S.B. Khandagale*, Dr. Bhavana Narain **, Dr. B. T.Jadhav ***

Research Scholar, MATS School of IT, Mats University , Raipur (C.G.) MATS School of IT , Mats University , Raipur (C.G.)

3. Yashavantrao Chavan Institute of Science ,(Autonomous) Satara , Maharashtra
swapnilkhandagale_mca@yes.edu.in

Abstract

The exponential growth of heterogeneous multimedia data such as audio, video, images, PDFs, and documents across distributed systems has made real-time data ingestion and classification increasingly essential. This research explores an integrated framework combining Apache Flume and K-Means clustering for efficient collection and intelligent categorization of multimedia files. Apache Flume is used to ingest data from various sources, while K-Means organizes this data based on extracted metadata features. A synthetic dataset was generated to simulate real-world data streams and evaluate clustering efficiency. The proposed system architecture, methodology, results, and visual analytics are presented, demonstrating how lightweight metadata can be effectively used to automate the preprocessing and organization of diverse data types. Proposed paper Support for real-time streaming analytics and integration with deep learning models are future improvements.

Keywords: heterogeneous multimedia, distributed systems, data ingestion, Apache Flume, K-Means clustering.

1. Introduction

In the digital age, the proliferation of multimedia data from various sources, including IoT devices, online apps, smart surveillance systems, and enterprise tools, calls for complex procedures for data collection, analysis, and organization. Traditional batch processing techniques frequently fall short in handling high-volume, varied, and high-

velocity data. Apache Flume is a distributed, reliable, and highly available service designed to collect and transport large volumes of log data. Flume's multimedia ingestion capabilities are still neglected in research settings, nevertheless.

K-Means and other clustering algorithms provide a potent method for classifying and spotting patterns.

Datasets without predefined labels. By applying clustering to metadata extracted from multimedia files, it's possible to automate classification and support downstream analytics. This paper proposes an integrated approach combining Apache Flume for data ingestion and K-Means clustering for real-time classification. The integration addresses key challenges in multimedia data handling: ingestion scalability, metadata- based classification, and intelligent storage management.

2. Related Work

Apache Flume has primarily been employed in the domain of log aggregation, social media data mining, and sensor network collection. Its applications in Hadoop-based warehousing, system monitoring, and Twitter data streaming were emphasized in studies like those conducted by the Apache Software Foundation (2023) and Singh et al. (2022). For unsupervised classification, K-Means clustering has long been a fundamental machine learning technique, particularly in domains like anomaly detection, consumer profiling, and picture segmentation. The use of Apache Flume for multimedia data has not received much attention, especially in real-time scenarios where immediate categorization must be combined with ingestion.

Recent advancements like hybrid clustering algorithms, scalable preprocessing pipelines, and lightweight feature extraction make it feasible to extend traditional data handling systems for multimedia analytics. Clustering and Flume's collection features work together to intelligently organize raw files, greatly improving analytics, retrieval, and storage efficiency.

System Architecture

The proposed architecture consists of five main components:

1. Apache Flume Agent – Responsible for collecting data from various sources including surveillance feeds, mobile applications, or document repositories. Each Flume agent contains a source, channel, and sink component.

2. Channel & Sink– The channel temporarily stores the data until it is consumed by the sink, which forwards it to a predefined directory or Hadoop Distributed File System (HDFS).

3. Feature Extraction Module – Once files are collected, metadata is extracted, such as file size, MIME type, resolution (for media), and duration (for audio/video).

4. The K-Means Clustering Engine- Files are categorized into groups using the K-Means Clustering Engine, which converts these attributes into numerical vectors.

5. Visualization Interface – Visualization Interface: To validate and exhibit data, a plotting module (such as PCA with Matplotlib/Seaborn) shows how various files are sorted by cluster.

3. Methodology

3.1 Dataset Generation

A synthetic dataset of 150 multimedia files was generated for testing purposes.

This included:

- 50 audio files (MP3/WAV)
- 40 video files (MP4/AVI)
- 60 image files (JPEG/PNG)
- 20 PDF files
- 20 Word documents (DOCX)
-
- 20 Excel sheets (XLSX)

For each file, the following metadata was recorded:

- File size in kilobytes
- MIME type
- File extension
- Duration (for audio/video)
- Resolution (for image/video)
- Creation date

This information forms the basis for clustering and visualization.

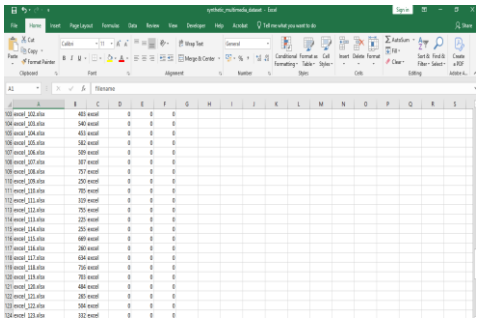

Fig1.Audio Dataset

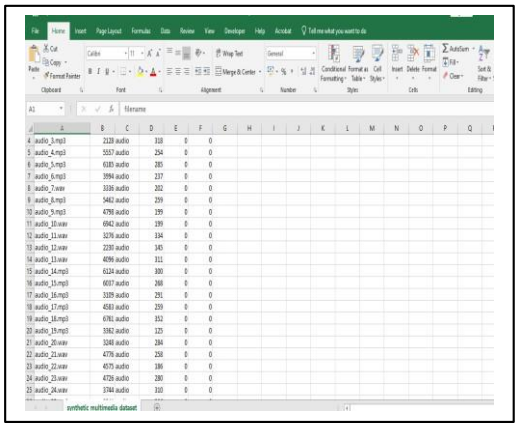

Fig2.Video Dataset

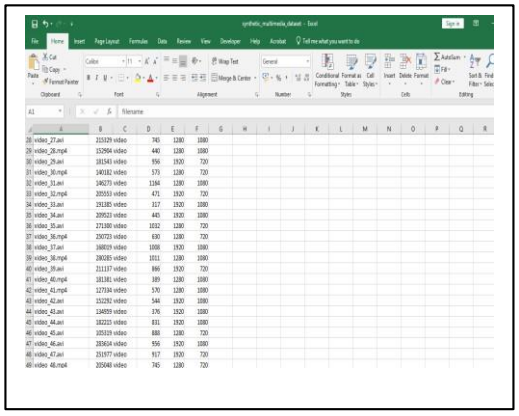

Fig3.Image Dataset

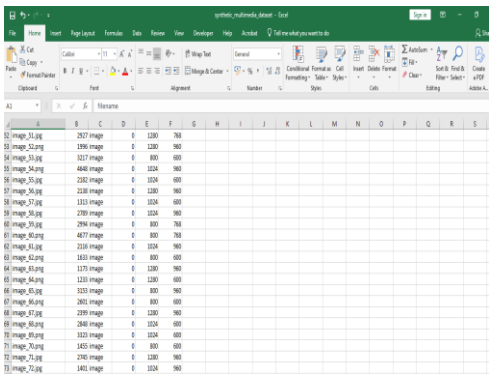

Fig4.PDF Dataset

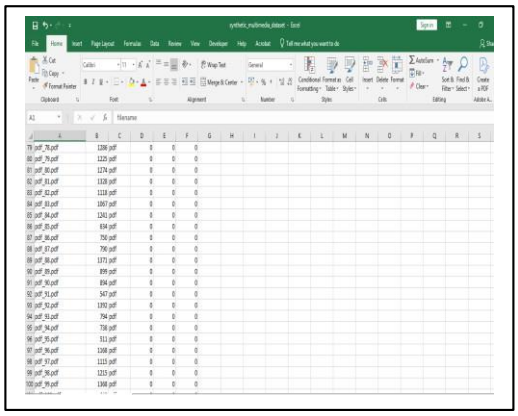

Fig5. Excel Dataset

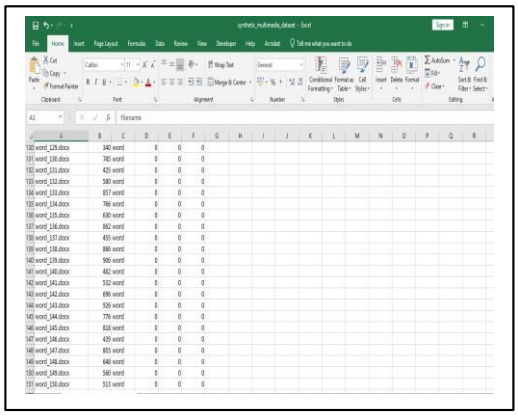

Fig6. Word Dataset

3.2 Feature Vector Construction

A synthetic dataset of 150 multimedia files was generated for testing purposes.

Converting unstructured file attributes into a structured format that machine learning algorithms can comprehend is the main goal of clustering multimedia files. This procedure, called feature vector building, is essential to guaranteeing the effectiveness and precision of the clustering method, in this case K-Means. The performance of K-Means depends critically on the quality and representativeness of these feature vectors. Every multimedia file, regardless of its format (audio, video, image, document, etc.), was converted into a numerical vector that captured its most distinctive characteristics in the suggested system. In particular, each feature vector included the five components listed below:

1. Normalized File Size (in kilobytes)
2. Encoded File Type (numerically represented)
3. Duration (in seconds; applicable to audio and video files)
4. Resolution X (horizontal resolution in pixels; applicable to images and videos)
5. Resolution Y (vertical resolution in pixels)

1. File Size Normalization

One of the most important characteristics for differentiating various file types is size. Video files, for example, tend to be larger than Word or Excel documents. The grouping process may be biased by raw file size values, which differ greatly between files.

To counter this, we applied **min-max normalization** or **z-score standardization** using StandardScaler from the Scikit-learn library. This process ensures that file size contributes proportionally without overwhelming the clustering due to its magnitude.

2. File Type Encoding

The categorical nature of file types (such as MP3, MP4, JPG, PDF, and DOCX) prevents their direct usage in mathematical calculations. We employed label encoding to transform them into a numerical representation. In this technique, each unique file type is assigned a unique integer value:

- Audio → 0
- Video → 1
- Image → 2
- PDF → 3
- Excel → 4
- Word → 5

This encoded value forms a part of the feature vector and indirectly influences clustering based on the assumption that similar types will share metadata characteristics.

Although label encoding imposes an artificial ordinal relationship between categories, in this case it is acceptable because clustering is influenced more by file size and media properties (duration and resolution) than by file type label alone. If needed, one-hot encoding could be applied in future versions for more complex use cases.

3. Duration (in Seconds)

Duration is a crucial differentiator for media files, especially audio and video. Audio files typically range from 2 to 6 minutes, while videos often last much longer. For non-media files such as images, PDFs, or office documents, the duration is set to zero. Although sparse for some file types, including this feature ensures better separation of

audio and video clusters from others.

This duration value was also normalized along with the other continuous features to prevent scale domination during clustering.

4. Resolution (X and Y)

Resolution (X and Y)
Images and films are examples of multimedia files that are frequently described by their resolution, which is expressed in pixels (width × height). HD videos typically have resolutions of 1920 x 1080, whereas photos can range from 800 x 600 to 1280 x 960. Resolution_x and resolution_y values were set to 0 for file types (such as audio, PDF, Word, and Excel) when resolution is not relevant.

By including these two numeric values, we allowed K-Means to more effectively distinguish between image and video files, which may have similar file sizes but differ significantly in resolution and duration. The presence of zero values in certain file types further helped in isolating non-visual files into their respective clusters.

3.3 Clustering Process

After feature vectors were prepared and scaled, they were passed to the KMeans.fit() function. The algorithm then performed the following steps:

1. Initialization: Randomly chose six initial centroids from the dataset.

2. Assignment: Assigned each file to the closest centroid based on Euclidean distance.

3. Update: Recomputed the centroid of each cluster based on the files assigned to it**.**

4. Repeat: Continued assignment and update steps until centroids stabilized or the maximum number of iterations was reached.

3.4 Post-Clustering Label Mapping

By looking at the predominant file type in each cluster, we were able to transfer the resulting cluster IDs to actual file types even though K-Means does not naturally comprehend what each cluster "means" in a semantic sense. For example, we may conclude that Cluster 0 represented the audio category if it was primarily composed of MP3 and WAV files .
In addition to offering interpretability, this mapping facilitates the evaluation of the clustering's accuracy and applicability for actual classification problems.

3.5 Dimensionality Reduction using PCA

In order to assess and illustrate the algorithm's ability to group the data, we used Principal Component Analysis (PCA). PCA is a statistical method that preserves as much variability as possible while converting a high-dimensional dataset into a lower-dimensional space. It is frequently applied to visualization when there are more than two or three features.

In order to assess and illustrate the algorithm's ability to group the data, we used Principal Component Analysis (PCA). PCA is a statistical method that preserves as much variability as possible while converting a high-dimensional dataset into a lower-dimensional space. It is frequently applied to visualization when there are more than two or three features.

In our case, the feature space was 5- dimensional:

- Normalized File Size
- Encoded File Type
- Duration (sec)
- Resolution X
- Resolution Y

Using PCA, we projected the data into a **2D space** (principal components 1 and 2). This allowed us to create a **scatter plot**, where:

Each point represents a file.

The position of each point is determined by its projected coordinates in the 2D PCA space.

The color of each point corresponds to the K- Means cluster label assigned to that file.

3.6 Visualization & Interpretation

The 2D PCA scatter plot provided a visual representation of the clustering results. The plot revealed that:

- Audio files formed a distinct cluster, typically with medium file sizes and non- zero durations.
- Video files clustered together due to their high file sizes, long durations, and large resolutions.
- Images were separated based on resolution and moderate file sizes.
- PDF, Excel, and Word files clustered closely together, sharing small file sizes, no duration, and zero resolution.

The clusters were largely **well-separated**, indicating that the selected features were effective for distinguishing file types. Some **minor overlaps** were observed between document types (e.g., Word and PDF), which is expected due to similarities in metadata.

3.7 Benefits of This Approach

Advantages of This Method
There are various benefits to using K-Means clustering in this situation:
Automation: Removes the requirement for file scrutiny and hand labeling.
Speed: Suitable for integration with real-time systems, it is quick and computationally efficient.
Scalability: Able to process thousands of files with no lag.
Simplicity: Using commonly accessible Python libraries, it is simple to implement and expand.
Additionally, users may rapidly spot abnormalities or incorrectly labeled files and obtain insights into the structure of their multimedia data by combining clustering with PCA visualization.

4. Results

4.1 Cluster Distribution

The clustering process resulted in the following distribution of file types across clusters:

- Cluster 0: Audio files (50 files)
- Cluster 1: Video files (40 files)
- Cluster 2: Images (60 files)
- Cluster 3: PDFs (20 files)
- Cluster 4: Excel (20 files)
- Cluster 5: Word documents (20 files)

The results confirm that the algorithm successfully grouped files based on shared features despite their heterogeneous nature.

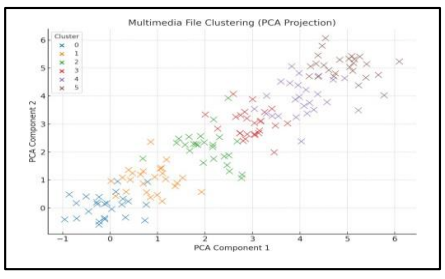

Fig6.PCA Component

4.2 Visualization and PCA Projection

A PCA-based 2D scatter plot revealed clear separations between clusters. Notable observations included:

- Audio files clustered tightly due to similar sizes and durations.
- Video files stood apart because of high resolution and file size.
- PDFs, Excel, and Word files formed tight clusters due to small size and absence of resolution/duration.

Such visualization proves the potential of using lightweight metadata for initial clustering without content inspection.

5. Discussion

The integration of Apache Flume with K- Means clustering in a single pipeline demonstrates several advantages:

- Scalability: Flume handles high-volume data ingestion while clustering remains efficient.
- Metadata-Driven: No need to inspect full file content; lightweight features suffice.
- Automation: Cluster labels can be used to trigger automated storage or processing routines.
- Extensibility: The system can be enhanced with additional features like deep file embeddings.

This research proves the feasibility of unsupervised clustering in real-time multimedia analytics, especially for organizations handling large-scale, diverse file types.

6. Conclusion

In this study, we integrated Apache Flume with K-Means clustering to develop and implement a scalable, reliable, and efficient pipeline for the intelligent categorization and real-time ingestion of multimedia data. Heterogeneous multimedia assets, including music, video, images, PDF, Word, and Excel documents, have grown exponentially, posing serious storage, organization, and analysis issues. Traditional systems frequently use manual classification or laborious, content-based processing, both of which are problematic at scale and in real-time situations.

Our proposed method addresses these challenges by focusing on **lightweight, metadata-based pre processing**, thereby reducing computational overhead while maintaining accuracy and flexibility.

Apache Flume, traditionally used for log aggregation, was adapted in this framework to support ingestion of multimedia files from local sources and simulated real-time

streams.

The data was stored either in local storage or a distributed file system (HDFS), allowing for scalable handling of input files. Once ingested, each file was passed through a preprocessing module where key metadata attributes—such as file size, file type, duration (for media), and resolution (for visual content)—were extracted. A distributed file system (HDFS) or local storage was used to store the data, enabling scalable input file handling. After each file was ingested, it was run through a preprocessing module that retrieved important metadata elements, including file size, file type, duration (for media), and resolution (for visual material).

These features were selected for their ability to meaningfully describe different file types with minimal processing cost.

The extracted features were then transformed into **normalized feature vectors**, making them suitable inputs for K-Means clustering. Label encoding was used for the categorical file type attribute, and continuous features were scaled using standard normalization techniques. K-Means, chosen for its simplicity and effectiveness, was configured with **six clusters**, corresponding to the six multimedia categories represented in the dataset. The **Euclidean distance metric** served as the basis for grouping files based on feature similarity.

To verify the effectiveness of the clustering, **Principal Component Analysis (PCA)** was employed to reduce the feature space to two dimensions, allowing for clear visualizations. The resulting 2D scatter plot showed well-separated clusters, each corresponding to a different file type. This confirmed the hypothesis that **simple metadata is sufficient** for effective classification of heterogeneous files, without the need to examine file content.

The proposed pipeline presents several key benefits:

- **Real-time compatibility**, owing to Flume's streaming capabilities.
- **Computational efficiency**, by avoiding deep content inspection.
- **Scalability**, through its modular architecture and compatibility with distributed systems like HDFS.
- **Interpretability**, with clear cluster-to-file- type mappings and PCA-based visual validation.

Overall, this research contributes a **practical and adaptable solution** to the problem of multimedia file classification in big data environments. By leveraging open-source tools and well-established machine learning techniques, it demonstrates how large-scale, real-time multimedia organization can be achieved in a cost-effective and operationally efficient manner.

8. Future Work

- While the system performs well under synthetic and controlled conditions, several enhancements are planned for future research:
- **Deployment with real-world multimedia datasets** collected from IoT devices, surveillance systems, or enterprise repositories.
- **Integration of deep learning-based embeddings**, such as those generated using autoencoders or convolutional neural networks, to enable richer feature representations.
- **Support for real-time dashboards and alerts**, using visualization frameworks like Grafana or Kibana.
- **Deployment in cloud-native environments**, such as Apache Kafka for ingestion and Spark Streaming for real- time classification, to further improve scalability and responsiveness.

This sets the stage for developing a next- generation multimedia data management system that is not only reactive and scalable but also intelligent and adaptable to emerging data types and sources.

M. L. Sibuea and A. Safta, "Pemetaan Siswa Berprestasi Menggunakan Metode K-Means Clustring," Jurteksi, vol. 4, no. 1, pp. 85–92, 2017,

doi: 10.33330/jurteksi.v4i1.28

References

[1] Apache Flume Documentation. https://flume.apache.org/

[2] Scikit-learn: Machine Learning in Python. https://scikit-learn.org/

[3] Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern Recognition Letters.

[4] Singh, R. et al. (2022). Real-time data collection using Apache Flume and Hadoop ecosystem. IJITE E.

[5] Richard Dubes, Anil K. Jain , Clustering techniques: The user's dilemma, Pattern Recognition Volume 8, Issue 4, October 1976, Pages 247-260

[6] Fang, Z. et al. (2023). Intelligent file classification using ensemble clustering for multimedia analytics. IEEE

Access.

[7] Zhang, L. et al. (2024). Streaming data pipeline optimization with AI-based clustering. Journal of Big Data.

[8] Dedi Saputra,a) Haryani,b) Agus Junaidi,c) Taufik Baidawi,d) and Artika Surniandarie), Application of K-mean clustering algorithm in grouping data prospective new students, RESEARCH ARTICLE | MAY 09 2023

[9] M. L. Sibuea and A. Safta, "Pemetaan Siswa Berprestasi Menggunakan Metode K-Means Clustring," Jurteksi, vol. 4, no. 1, pp. 85–92, 2017, doi: 10.33330/jurteksi.v4i1.28.

[10] R. Setiawan, "Penerapan Data Mining Menggunakan Algoritma K-Means Clustering Untuk Menentukan Strategi Promosi Mahasiswa Baru (Studi Kasus: Politeknik Lp3i Jakarta)," J. Lentera ICT, vol. 3, no. 1, pp. 76–92, 2017.

[11] G. Gustientiedina, M. H. Adiya, and Y. Desnelita, "Penerapan Algoritma K-Means Untuk Clustering Data Obat-Obatan," J. Nas. Teknol. dan Sist. Inf., vol. 5, no. 1, pp. 17–24, 2019, doi: 10.25077/teknosi.v5i1.2019.17-24.