An Ensemble Clustering Approach for Modeling Hidden Categorization Perspectives for Cloud Workloads

Mustafa Daraghmeh[1*], Anjali Agarwal[1†] and Yaser Jararweh[2†]

[1]Electrical and Computer Engineering, Concordia University, 1455 De Maisonneuve Blvd. W., Montreal, H3G 1M8, QC, Canada.

[2]Computer Science, Jordan University of Science and Technology, Irbid, 22110, Jordan.

*Corresponding author(s). E-mail(s): Mustafa.Daraghmeh@concordia.ca; Contributing authors: Anjali.Agarwal@concordia.ca; yijararweh@just.edu.jo;
†These authors contributed equally to this work.

Abstract

Effectively managing cloud resources is a complex task due to the interdependencies of various cloud-hosted services and applications. This task is integral to workload categorization, which groups similar cloud workloads to inform scheduling and resource management procedures. Although traditional clustering algorithms can categorize workloads into single data grouping patterns, the multifaceted nature of cloud workloads may conceal multiple valid categorization perspectives, indicating a need for a more adaptable clustering approach. This paper presents a novel ensemble clustering approach to enhance the categorization process for scheduling workloads in cloud data centers. This approach combines various normalization and transformation techniques, including Principal Component Analysis, to create multiple data preprocessing pipelines. The data derived from these pipelines serves as input for different base-clustering models. To improve the accuracy of selecting optimal base-clustering models and preprocessing pipelines, we utilize a combined score based on the Silhouette score, Calinski-Harabasz index, and Davies-Bouldin index. The clustering outcomes of these models are encoded and inputted into a meta-clustering algorithm, effectively capturing complex categorization perspectives.The assessment of this approach, utilizing authentic workload trace data from Microsoft Azure, significantly enhances workload segmentation efficacy. Consequently, this improvement elevates resource management and quality of service standards in cloud data centers. It offers a promising path toward improved workload clustering in cloud systems, demonstrating the practical utility of advanced ensemble clustering techniques to unveil hidden categorization perspectives.

Keywords: Data Preprocessing, Ensemble Clustering, Cloud Workload Categorization, Cloud Computing.

1 Introduction

Cloud computing is a technology that offers unparalleled flexibility, scalability, and cost-effectiveness in processing and managing various applications. The exponential growth in the number and diversity of cloud-hosted services and applications has amplified the complexity of resource management and led to intricate interdependencies. Effectively categorizing workloads is essential for precise decision-making in load scheduling and resource allocation, improving overall infrastructure management [23].

The sheer volume, velocity, and variety of workloads in large-scale cloud environments make it challenging for conventional clustering methods to produce accurate and meaningful results. Although these methods are somewhat effective, they have limitations when dealing with cloud workloads' diverse and multifaceted nature [8]. The scalability and heterogeneity of the cloud environment frequently result in overlapping and nested clusters that are challenging to manage with conventional methods.

In addition, hidden within the complexity of cloud workloads are multiple valid categorization perspectives that single clustering methods may overlook. This obfuscation compromises the efficiency of resource and task scheduling strategies in cloud data centers, resulting in suboptimal performance, increased costs, and diminished service quality. Therefore, a more adaptable and nuanced approach to workload categorization in cloud systems is necessary and timely.

Moreover, it is crucial to preprocess the observed data to ensure the precision of Machine Learning (ML) models. This process involves cleaning, integrating, transforming, and reducing the data to improve its quality and usefulness [28]. In cloud computing, acquiring knowledge can be challenging due to the diverse workloads being executed. In order to improve data quality and obtain valuable insights, it is crucial to have adequate data preparation pipelines. Using efficient ones in clustering and prediction models can lead to more reliable and accurate results, optimizing the Cloud Resource Management System (CRMS) operational outcomes, which ensures efficient and effective resource utilization.

Given the limitations inherent in current clustering methods, this research aims to elevate workload segmentation by leveraging an advanced ensemble clustering approach. This study intends to refine the categorization process by employing enhanced data preprocessing and clustering methodologies tailored to cloud workloads' multifarious nature. The main contributions of this study can be summarized as follows.

- The formulation of a novel ensemble clustering technique that amalgamates diverse data preprocessing pipelines with varied base clustering learners, thereby ensuring a comprehensive understanding of intricate cloud workload patterns.
- Empirical validation of the proposed technique using real-world trace data from Microsoft Azure, attesting to its robustness, precision, and relevance in identifying complex workload attributes.
- Introduction of a unique scoring method that integrates the Silhouette Coefficient (SC), Calinski-Harabasz Index (CHI), and Davies-Bouldin Index (DBI) metrics. This score facilitates the discerning selection of optimal clustering models and preprocessing mechanisms, illuminating the nuanced interplay between distinct configurations.
- Detailed analysis of the interactions between different clustering algorithms and a combination of transformation and normalization methods, offering profound insights into their collective influence on categorization efficacy.
- To increase the adaptability of the reduction of data dimension, we combine the Kneedle method [38] with Principal Component Analysis (PCA), as detailed in Algorithm 3. This novel fusion approach autonomously determines the optimal component count through the inflection in the cumulative variance curve, obviating predefined thresholds and enhancing processing efficacy.

The significance of this study lies in its potential to markedly enhance workload segmentation efficiency, paving the way for nuanced analyses and optimized cloud infrastructure management. By highlighting the efficacy of ensemble clustering in addressing the multifaceted nature of cloud workloads, this research seeks to propel advancements in the field. Furthermore, the proposed ensemble clustering approach underscores a promising avenue for superior cloud workload clustering. It exemplifies the integration potential of sophisticated clustering strategies within real-world cloud resource management contexts.

The remainder of this paper is organized as follows. Section 2 elaborates on different clustering methods in the literature and their use in cloud data centers. Section 3 details our proposed ensemble clustering approach and its components, including the mathematical formulation. In Section 4, we delve into the experimental design, elucidate the resulting findings, and highlight prospective avenues for future research. Section 5 captures the concluding thoughts and reflections.

2 Related Work

The success of cloud computing systems and providers' financial gain depend on efficient resource management [1]. In order to meet the diverse needs of users, cloud services are offered under various categories, such as infrastructure, platform, or application-based, which are tailored to meet distinctive requirements [44]. Given the limited availability of resources, implementing effective resource management strategies is imperative to maximize provider profits and meet users' demands [50]. However, there has been a significant emphasis on enhancing the efficiency and effectiveness of workload categorization in cloud computing systems. This section reviews several prevalent techniques, outlining their strengths and limitations and setting the stage for the innovative approach proposed in this study.

Jayaprakash et al. [23] conducted an in-depth analysis that compares and classifies various clustering, optimization, and ML methods used as a practical approach that manages the resource allocation in the cloud. Clustering methods and optimization techniques are seen as pivotal in resource management due to their potential to offer solutions for efficient resource and energy management. The authors have acknowledged that the current methods being utilized have certain limitations. These include low convergence, a tendency to become entrenched in suboptimal solutions, and overfitting.

Conventional clustering techniques, such as KMeans and hierarchical clustering (with an early stop of tree construction), have been extensively used for the process of cloud workload categorization [3, 12, 14, 24, 26, 34, 35, 40, 47, 53]. These algorithms categorize workloads into specific grouping patterns based on similarity metrics. Implementing these methods has proven highly effective, offering a streamlined and efficient approach to handling extensive datasets. Their simplicity affords a greater degree of manageability, providing a reliable means of data management that can prove invaluable in various professional settings. However, these algorithms have significant limitations. They require a prespecified number of clusters, which may not align with the actual structure of cloud workloads. Also, they can be sensitive to initial starting conditions and outliers, leading to inconsistent results [36].

Density-based clustering algorithms have been proposed to overcome some of the limitations of traditional clustering techniques to cluster cloud workloads efficiently [15, 18, 22, 49]. Compared to more conventional algorithms, these are more versatile because they can detect clusters of arbitrary shapes and do not require a predefined number of clusters. However, high-dimensional data and clusters of varying densities present challenges [6].

Another stream of research focuses on grid-based clustering techniques that have been explored for categorizing cloud workloads [16, 22]. These techniques partition the given data space into a finite number of cells and perform the clustering process on these cells, allowing for rapid processing time. However, the quality of clusters is heavily dependent on the granularity of the grid, and these methods can overlook more refined details at lower granularities [43].

Classification techniques, including Support-Vector Machine (SVM) and Decision Trees, have also been explored for cloud workload categorization [5, 20, 26, 31]. These methods have the advantage of being able to handle non-linear data and offer interpretability. However, they may overfit the data and require careful tuning of parameters. Furthermore, these techniques are primarily concerned with classification rather than clustering, which makes them less suitable for workload categorization in situations where natural groupings still need to be discovered.

Deep learning-based methods such as Autoencoders and Deep Belief Networks (DBN) have emerged, allowing for handling high-dimensional data and discovering complex patterns [17, 19, 21, 30, 48, 51]. However, these methods require large amounts of data for training and can be computationally intensive. Also, they often act as a black box, providing little insight into the underlying clustering process.

Finally, Strehl et al. [41] introduced three heuristics for the cluster ensemble. The Cluster-based Similarity Partitioning Algorithm (CSPA) establishes pairwise similarity among cluster objects to form a unified clustering. The HyperGraph Partitioning Algorithm (HGPA) approximates mutual information, turning the ensemble problem into a hypergraph partitioning task. The Meta-CLustering Algorithm (MCLA) focuses on identifying and merging groups of clusters by addressing cluster correspondence.

Table 1: Comparative Analysis of Leading Methods in Literature for Cloud Workload Categorization.

Methods	**Strengths**	**Limitations**
KMeans used by [3, 14, 26, 35, 40, 47, 53]	• Suits large datasets • Simplified approach	• Predefined cluster count • Initialization sensitivity
Agglomerative hierarchical clustering used by [12, 24, 34]	• Rich dendrogram output • Diverse distance metrics	• Slow on vast datasets • Decisions are permanent
Density-based used by [15, 18, 22, 49]	• Detects diverse cluster shapes • Spatial cluster aptitude • No set cluster count	• Limited in high-dimensions • Parameter sensitivity
Grid-based used by [16, 22]	• Rapid due to cell count • Scalable structure	• Grid granularity matters • May overlook details
Classification techniques used by [5, 20, 26, 31]	• Addresses non-linear patterns • Interpretable models	• Needs labeled data • Overfitting risks
Deep learning used by [17, 19, 21, 30, 48, 51]	• Captures complex patterns • Non-linear modeling • Scalable with data volume	• Demands ample training data • Computationally heavy • ”Black box” nature
Ensemble Clustering used by [9, 13, 29, 41, 45, 52]	• Enhances robustness • Combats individual flaws	• Setup complexity • Potentially resource-intensive

Another significant work in ensemble clustering is by Topchy et al. [45], who introduced a graph-based methodology. The approach combines different clusterings in a hyper-graph, where each hyper-edge represents a cluster. They then proposed a graph partitioning algorithm to find the consensus function. Meta-clustering approach has also been explored [9, 13, 29, 52]. In this method, they treated each base clustering solution as an entity. Subsequently, they are subjected to another clustering round to identify and group analogous solutions, culminating in a consensus solution. However, there is no one-size-fits-all method for all scenarios, so it is essential to carefully select or create a method appropriate for the specific dataset and task at hand, given the variety of available methods.

As delineated in Table 1, contemporary clustering methods for cloud workload categorization inevitably exhibit intrinsic constraints. The inherent complexity and multifarious nature of cloud workloads often challenge the applicability of conventional clustering algorithms. Furthermore, the efficacy observed with a particular dataset seldom ensures uniform performance across diverse datasets, highlighting the imperative for more adaptive and multifaceted techniques. With the continual progression of technology leading to increased heterogeneity and dynamism in cloud workloads, there is an emerging necessity for robust and adaptive methodologies.

Ensemble clustering, which synthesizes the strengths of various clustering paradigms, presents a promising avenue. By amalgamating various clustering outputs, it not only bolsters the robustness of categorizations but also adeptly counters the deficiencies inherent to individual methodologies. Given the escalating intricacies of cloud ecosystems, the strategic integration of ensemble techniques is poised to play a pivotal role in optimizing cloud workload management in future research and applications.

The current discourse in cloud workload categorization, while rich in its exploration of individual clustering techniques, leaves a discernible gap: the cohesive amalgamation of these techniques, particularly when integrated with advanced data preprocessing pipelines. Ensemble clustering emerges as a beacon of potential in this landscape. Our research is driven by the ambition to bridge this gap, striving for a harmonious integration of diverse techniques and transcending their limitations. While the robustness of ensemble clustering is acknowledged, its nuanced application to cloud workloads via multiple data preprocessing pipelines remains largely uncharted. This research seeks to pioneer in this direction, offering deep insights and practical solutions that align with the multifaceted nature of cloud workloads in large-scale data centers.

Our proposed ensemble clustering methodology stands apart in its innovative amalgamation of multiple clustering paradigms and sophisticated data preprocessing pipelines. This strategic fusion aims to harness the collective strengths of individual methods, revealing intricate cloud workload categorizations that might elude standalone techniques. Grounded in real-world scenarios and rigorously tested on genuine cloud workload trace, this research promises tangible contributions to cloud computing. Moreover, it paves the way for actionable insights and practical guidelines on the merits and modalities of ensemble clustering, tailored specifically for the complexities of cloud workloads in large-scale cloud environments.

3 Proposed Ensemble Clustering Approach

Data preprocessing pipelines are a systematic sequence of steps used to clean and transform raw data into a format suitable for analysis, ensuring consistent and efficient data processing [28]. On the other hand, ensemble clustering merges multiple clustering algorithms or configurations to categorize data [46]. This combination aims to maximize the strengths and minimize the weaknesses of individual methods, yielding more accurate and robust groupings of data points. Both concepts underscore the value of integrating various methods to achieve enhanced results. Figure 1 outlines the proposed ensemble clustering combining base clustering pipelines (multiple clustering algorithms and data preprocessing pipelines) to improve cloud workload categorization.

The approach begins with constructing base clustering pipelines, each comprising a unique combination of data preprocessing and base clustering algorithms. These candidate pipelines are then evaluated and selected based on their ability to partition the data effectively. The selected pipelines generate clusters whose labels are encoded using a count encoder. A PCA-based dimensionality reduction can be applied to handle potential high dimensionality in the label space. Lastly, a meta-clustering algorithm consolidates the results from the selected base

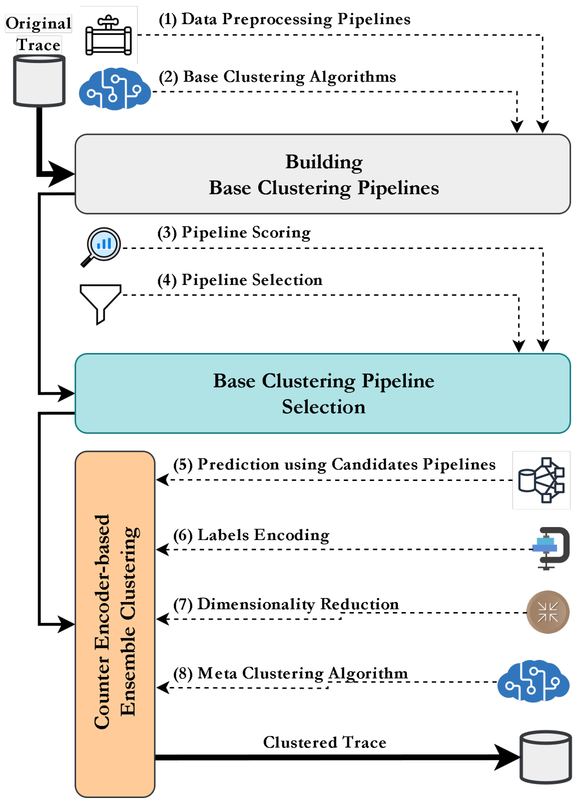

Original Trace
(1) Data Preprocessing Pipelines
(2) Base Clustering Algorithms
Building Base Clustering Pipelines
(3) Pipeline Scoring
(4) Pipeline Selection
Base Clustering Pipeline Selection
(5) Prediction using Candidates Pipelines
(6) Labels Encoding
(7) Dimensionality Reduction
(8) Meta Clustering Algorithm
Counter Encoder-based Ensemble Clustering
Clustered Trace

Fig. 1: Visual representation of the proposed ensemble clustering model showcasing the processes the original workload trace goes through to perform the segmentation process.

clustering pipelines, producing a final ensemble clustering output. Thus, this approach leverages the strengths of various clustering algorithms and data preprocessing methods, providing robust and reliable categorization of cloud workloads.

3.1 Building Base Clustering Pipelines

The Algorithm 1 outlines the first significant step in our ensemble clustering approach - *Building Base Clustering Pipelines*. This step is critical to preparing a comprehensive set of base clustering pipelines that constitute the building blocks of our ensemble approach. These pipelines combine each data preprocessing method, denoted by P, with each base clustering algorithm, denoted by B. Both P and B are lists; P consists of different data preprocessing pipelines such as normalization, standardization, PCA-based dimensionality

Table 2: Table of Definitions

Symbol	**Definition**
X	Workload trace data matrix
P	Set of data preprocessing processes
B	Set of base clustering algorithms
CP	List of combined clustering pipelines
CP_{ij}	Clustering pipeline combining the i-th of P with the j-th of B
X'_{ij}	Data post application of CP_{ij}
L_{ij}	Clustering labels from CP_{ij}
SC_v, CHI_v, and DBI_v	SC, CHI, DBI score lists for CP
SC_{ij}, CHI_{ij}, and DBI_{ij}	Evaluation scores for CP_{ij}
SC'_{ij}, CHI'_{ij}, and DBI'_{ij}	Normalized scores of CP_{ij}
CS	Combined score list for CP
CS_{ij}	Combined score for CP_{ij}
SP	Pipeline selection criteria
T	Threshold based on SP
CP'	Selected top-performing pipelines
M	Meta clustering algorithm
k	Specified cluster count
k'	'Knee' point in variance data
a	Flag for applying PCA to base clustering outputs
X''	Input data matrix for PCA
u	Flag for using Kneedle algorithm [38] in PCA
$\mathbb{E}$	PCA variance ratio
v	Explained variance threshold for PCA
$\mathbb{C}$	Cumulative variance ratio in PCA
L	Label list from selected top-performing pipelines CP'
L^T	Transposed version of L
L_{en}	Counter-encoded labels of L^T
L'_{en}	PCA-transformed L_{en}
L'	Final ensemble labels

reduction, and other feature engineering methods as required based on the data structure. While B comprises various base clustering algorithms such as KMeans, Agglomerative Clustering, MeanShift, and other algorithms as required based on the given data characteristics.

In detail, the algorithm begins by initializing an empty list CP to hold the resulting base clustering pipelines (Line 1). It then proceeds with two nested loops to iterate over every possible combination of elements from lists P and B (Lines 2 to 7). In each iteration, a base clustering algorithm B_j is combined with a preprocessing data pipeline P_i to form a new clustering pipeline, denoted by CP_{ij} (Line 4). The clustering pipeline is basically a sequence of preprocessing steps P_i that need to be applied to the data X and ends with the clustering algorithm B_j. Each created pipeline CP_{ij} is then added to the list CP (Line 5), which includes a pool of pipeline combinations.

This systematic process ensures that every potential combination of preprocessing and base clustering methods is considered, thus increasing the likelihood of finding a set of pipelines that offer high-quality clustering results. At the end of the algorithm, the list CP is returned (Line 8).

The list CP contains base clustering pipelines to be scored and evaluated in the next step of the ensemble clustering approach, known as *Base Clustering Pipeline Selection*. This step is not a mere cursory evaluation. It is a pivotal juncture where the theoretical constructs of our approach meet empirical validation. We undertake a rigorous quality assurance process by scoring and evaluating the pipelines stored in CP. This step ensures that only the most efficacious pipelines, which are in harmony with the intrinsic data structure, are retained. Such precision-driven selection amplifies the robustness of our ensemble clustering, ensuring that our final model is not just a conglomeration of algorithms but a synergized ensemble informed by data-driven insights. This iterative feedback and selection mechanism distinguishes our approach, placing it above conventional ensemble clustering methods.

3.2 Base Clustering Pipeline Selection

Algorithm 2 presents the second major step of our proposed ensemble clustering approach - *Base Clustering Pipeline Selection*. The primary aim of this step is to identify the top-performing base clustering pipelines that will contribute to the final ensemble clustering solution. This process is achieved by evaluating each pipeline's clustering performance on the trace data, X, using a Combined Score (CS) of a set of evaluation metrics. The clustering pipelines are then filtered based on a threshold T calculated based on a criterion defined by the selection policy, SP, such as the top models m or the medium score threshold.

The choice of clustering evaluation metrics is grounded in thoughtful deliberation. We have chosen SC, CHI, and DBI metrics for their proven track record and effectiveness across several studies, detailed in Section 4.1. These metrics provide a comprehensive assessment of clustering quality: SC measures the tightness and isolation of clusters, CHI evaluates intra-cluster cohesion, and DBI inspects the distinctness between clusters. By adopting a multifaceted approach to the evaluation process, we can ensure that clustering results are robust and holistic.

To initiate, the algorithm first constructs the base clustering pipelines by invoking Algorithm 1, with the original data X, the list of data preprocessing pipelines P, and the base clustering algorithms B as input parameters (Line 1). Following this, it initializes three empty lists, SC_v, CHI_v, and DBI_v, which will, respectively, store the SC, CHI, and DBI scores for each base clustering pipeline (Line 2). In detail, the algorithm performs the following operations for each pipeline CP_{ij} in the constructed list of base clustering pipelines, CP. It first fits the pipeline CP_{ij} on the original data X to produce the cluster labels L_{ij} and transformed data X' (Line 4). It then calculates the SC, CHI, and DBI scores using the transformed data X' and labels L_{ij}, and appends these scores to the respective lists SC_v, CHI_v, and DBI_v (Lines 5 to 6).

Once the evaluation metrics have been calculated for each pipeline, the algorithm calculates a CS (Lines 8 to 13). It first normalizes the SC, CHI, and DBI values using Equations 1, 2, 3, to generate normalized scores $SC'ij$, $CHI'ij$, and $DBI'ij$ respectively. It then calculates the combined score $CSij$ for each pipeline as per Equation 4, and appends each CS_{ij} to the list CS. Following the computation of combined scores, the selection policy SP is applied on the list CS to determine a threshold T for selecting the top-performing pipelines (Line 14). Those pipelines whose combined score CS is above the threshold T are selected and stored in the list CP' (Line 15).

It is crucial to specify the precise context in which the proposed combined score operates optimally. This metric has been judiciously designed to amalgamate the outcomes, namely SC_v, CHI_v, and DBI_v, derived from various candidate clustering pipelines. Through rigorous normalization procedures, the inherent diversity in the value ranges of these metrics is harmonized to fit a consistent scale, ensuring a holistic and equitable synthesis into the combined score. It is noteworthy, however, that this combined metric is tailored for appraising an ensemble of clustering pipelines. For evaluating an individual clustering algorithm, such as the final ensemble clustering algorithm, the utility of the combined score is constrained. In such instances, the conventional metrics, specifically SC, CHI, and DBI, proffer a more immediate and illustrative assessment.

In summary, Algorithm 2 effectively ranks and selects the base clustering pipelines based on their clustering performance, represented by a combined score of multiple validity indices. The output is a subset of the original list of base clustering pipelines, CP', that contains the top-performing pipelines selected for the final ensemble clustering.

3.3 Counter Encoder-based Ensemble Clustering

Firstly, in Algorithm 4 - *Counter Encoder-based Ensemble Clustering*, it takes as input the original workload trace data matrix, X, the list of

Algorithm 1 Building Base Clustering Pipelines

```
Input
    $X$    Original workload trace data matrix
    $P$    $= P_1, P_2, ..., P_s$ Data Pipeline List
    $B$    $= B_1, B_2, ..., B_c$ Base Clustering Algorithms
Output
    $CP$   $= CP_{ij} : CP_{ij}$ is a clustering pipeline that combines $P_i$ and $B_j$      ▷ List of clustering pipelines
1: $CP \leftarrow$ Empty list                                          ▷ Initialize the pipeline list
2: for each $P_i$ in $P$ do                                            ▷ Iterate over preprocessing methods
3:     for each $B_j$ in $B$ do                                        ▷ Iterate over base algorithms
4:         Combine $B_j$ with $P_i$ based on $X$ and obtain $CP_{ij}$  ▷ Create a clustering pipeline $CP_{ij}$
5:         Add $CP_{ij}$ to $CP$                                       ▷ Add the pipeline to the list
6:     end for
7: end for
8: return $CP$                                                         ▷ Return the list of pipelines
```

Algorithm 2 Base Clustering Pipeline Selection

```
Input
    $X$    Original workload trace data matrix
    $P$    $= \{P_1, P_2, ..., P_s\}$ Data Pipeline List
    $B$    $= \{B_1, B_2, ..., B_c\}$ Base Clustering Algorithms
    $SP$   Selection policy
Output
    $CP'$  List of selected clustering pipelines
1: $CP \leftarrow$ Call Algorithm 1 with $X$, $P$, and $B$             ▷ Construct base clustering pipelines
2: Initialize $SC_v, CHI_v, DBI_v$ as empty lists   ▷ Initialization of lists to store clustering validity scores
3: for each $CP_{ij}$ in $CP$ do                                       ▷ Iterate over all pipeline combinations
4:     Fit $CP_{ij}$ on $X$ to get labels $L_{ij}$ and transformed data $X'$   ▷ Process data using current pipeline
5:     Calculate $SC_{ij}$, $CHI_{ij}$, and $DBI_{ij}$ scores using $X'$ and $L_{ij}$   ▷ Evaluate the base clustering pipelines
6:     Append $SC_v$, $CHI_v$, and $DBI_v$ with $SC_{ij}$, $CHI_{ij}$, and $DBI_{ij}$, respectively.
7: end for
8: Initialize $CS$ as empty list                                       ▷ Prepare for combined validity scores
9: for $(SC_{ij}, CHI_{ij}, DBI_{ij})$ in zip$(SC_v, CHI_v, DBI_v)$ do    ▷ Iterate over clustering validity scores
10:    Normalize the validity scores according to Equations 1, 2, and 3 to obtain $SC'_{ij}$, $CHI'_{ij}$, and $DBI'_{ij}$.
11:    Compute Combined Score $CS_{ij}$ according to Equation 4       ▷ Aggregate the normalized scores
12:    Append $CS_{ij}$ to $CS$                                        ▷ Add aggregated score to the list
13: end for
14: Calculate threshold $T$ using the selection policy $SP$ applied on $CS$   ▷ Determine cut-off score for selection
15: $CP' \leftarrow$ base clustering pipelines with $CS$ above $T$ from $CP$   ▷ Filter pipelines surpassing the threshold
16: return $CP'$                                                       ▷ Return the subset of top-performing pipelines
```

selected base clustering pipelines, CP', the meta clustering algorithm, M, and a few additional parameters. If a specific number of clusters, k, has been provided and the meta clustering algorithm M has an attribute $n\_clusters$, the algorithm sets $M.n\_clusters$ to k (Lines 1 to 3). For each selected base clustering pipeline, $CP'ij$, in the list CP', the algorithm fits the pipeline on the data X and retrieves the cluster labels Lij (Line 6). These labels are added to an initially empty list, L (Line 7). After obtaining the cluster labels from all selected pipelines, the algorithm transposes the list L to L^T (Line 9). The counter-encoding method is then applied to L^T, resulting in the encoded labels L_{en} (Line 10).

Following the counter encoding process, Algorithm 3 - *PCA-based Dimensionality Reduction* is applied if the flag a is set to True. As an optional process, this algorithm leverages PCA to reduce the dimensionality of the encoded labels, L_{en}, and transform them into a lower-dimensional space that maintains most of the original data variance. There are two recommended methods to determine the optimal PCA components. The first involves utilizing a Kneedle algorithm [38] to identify the inflection point. Alternatively, a threshold for the explained variance, denoted by

Algorithm 3 PCA-based Dimensionality Reduction

```
Input
    $X'' = \{X''_1, X''_2, ..., X''_n\}$ Input Data Matrix
    $u$   Use Kneedle True or False
    $v$   Explained Variance Threshold
Output
    $X'$  Transformed $X''$ after applying PCA
1: Apply PCA on $X''$ and get explained variance ratio $\mathbb{E}$   ▷ Apply PCA and Compute variance ratios for data
2: Cumulative Explained Variance Ratio $\mathbb{C} \leftarrow$ Cumulative Sum of $\mathbb{E}$   ▷ Get cumulative sum of variances
3: if $u$ then   ▷ Check if Kneedle method is to be used
4:     Use Kneedle to find knee $k'$ from $\mathbb{C}$   ▷ Identify the 'knee' point in the cumulative variance
5:     if knee $k'$ is None then   ▷ Check if no knee point was found
6:         $k \leftarrow$ first index where $\mathbb{C}$ exceeds $v$   ▷ Determine number of components based on threshold
7:     end if
8: else
9:     $k \leftarrow$ first index where $\mathbb{C}$ exceeds $v$   ▷ Determine number of components directly based on threshold
10: end if
11: Apply PCA with $k'$ components to $X''$ and get $X'$   ▷ Apply dimensionality reduction
12: return $X'$   ▷ Return transformed data
```

Algorithm 4 Counter Encoder-based Ensemble Clustering

```
Input
    $X$     Original workload trace data matrix
    $CP'$   List of selected base clustering pipelines
    $M$     Meta Clustering Algorithm
    $k$     number of clusters
    $a$     Apply PCA to base Clustering Algorithms Results {True or False}
    $u$     Use Kneedle {True or False}
    $v$     Explained Variance Threshold
Output
    $L'$    Ensemble Clustering Labels
1: if $k$ is not None and has $M.n\_clusters$ then   ▷ Check if a specific number of clusters is provided
2:     $M.n\_clusters \leftarrow k$
3: end if
4: $L$ as empty list   ▷ Initialize list for base clustering labels
5: for each $CP'_{ij}$ in $CP'$ do   ▷ Iterate over selected base clustering pipelines
6:     Fit $CP'_{ij}$ on $X$ and get labels $L_{ij}$   ▷ Compute labels using each pipeline
7:     Append $L_{ij}$ to $L$
8: end for
9: $L^T \leftarrow$ transpose of $L$   ▷ Transpose to structure data for encoding
10: Encode $L^T$ using **counter encoding method** and get encoded labels $L_{en}$   ▷ Counter encode the labels
11: if $a$ then   ▷ Check if PCA is to be applied
12:     $L'_{en} \leftarrow$ Call Algorithm 3 with $L_{en}$, $u$, and $v$
13: else
14:     $L'_{en} \leftarrow L_{en}$
15: end if
16: Fit $M$ on $L'_{en}$   ▷ Apply meta clustering on encoded labels
17: $L' \leftarrow M$.labels   ▷ Retrieve the final ensemble labels
18: return $L'$   ▷ Return the final ensemble clustering labels
```

the v, can be set to attain the desired outcome. Both approaches are practical and depend on the specific needs of the analysis. However, Algorithm 3 applies PCA on L_{en}, yielding an explained variance ratio, $\mathbb{E}$ (Line 1). The cumulative explained variance ratio, $\mathbb{C}$, is calculated as the cumulative sum of $\mathbb{E}$ (Line 2). If the use Kneedle flag, u, is set to True, a Kneedle algorithm finds a knee point, k', from $\mathbb{C}$ (Line 4). If no knee point is found, or if u is set to False, k' is determined as the first index where $\mathbb{C}$ exceeds the explained variance threshold, v. Lastly, PCA is applied with k' components to L_{en}, resulting in the transformed encoded labels L'_{en} (Line 11).

In Algorithm 3, the number of PCA components is determined primarily using the Kneedle algorithm. This technique is tailored to detect the knee point in the cumulative variance curve, configured as ”concave” and ”increasing”. The knee represents a critical inflection where further inclusion of components offers diminishing explanatory returns. However, we adopt an intuitive selection approach when the Knee Locator fails to identify a clear knee. We can select a variance threshold v, typically between 95% and 99%, which a systematic trial-and-error tuning process can guide to achieve optimal performance.

After the dimensionality reduction, in Algorithm 4, the final step is to fit the meta clustering algorithm M on L'_{en} (Line 16). The ensemble clustering labels, L', are obtained as M.labels (Line 17). The algorithm then returns these labels (Line 18), thus completing the ensemble clustering process based on multi-perspective data preprocessing and clustering pipelines.

In summary, the *Counter Encoder-based Ensemble Clustering* algorithm and the *PCA-based Dimensionality Reduction algorithm* work synergistically to transform the base clustering results into a suitable format, reduce their dimensionality, and generate the final ensemble clustering labels. This combination provides a practical and adaptable approach to ensemble clustering based on multi-perspective pipelines. Consequently, employing various clustering algorithms, in this context, can significantly improve the overall performance compared to relying solely on a singular clustering algorithm.

3.4 Mathematical Formulations

Our ensemble clustering method uses a set of mathematical procedures to achieve robust clustering outcomes. These processes encompass pipeline evaluation, score normalization, counter encoding, and the final meta-clustering. In order to measure the effectiveness of the essential clustering pipelines, we utilize three commonly used clustering metrics (SC, CHI, and DBI). These metrics are detailed in Section 4.1. Each of these metrics provides valuable insights into the quality of the clustering generated by a base clustering pipeline CP_{ij} on a given dataset X, resulting in a corresponding label set L_{ij}. These metrics for each pipeline coalesce into metric vectors, namely SC_v, CHI_v, and DBI_v. However, these metrics vary in their ranges, making direct comparisons challenging. Thus, we normalize these metrics to ensure they reside on a uniform scale, as follows:

$$SC'ij = \frac{SCij - \min(SC_v)}{\max(SC_v) - \min(SC_v)} \tag{1}$$

$$CHI'ij = \frac{CHIij - \min(CHI_v)}{\max(CHI_v) - \min(CHI_v)} \tag{2}$$

$$DBI'ij = 1 - \frac{DBIij - \min(DBI_v)}{\max(DBI_v) - \min(DBI_v)} \tag{3}$$

Each normalized score now falls within the interval [0, 1], facilitating their amalgamation into a combined score CS_{ij} using weights α, β, and γ for each respective metric. The optimal value corresponds to the maximum score in the context of these normalized scores and the resulting combined one. The maximum value represents the highest performance level achievable among the clustering pipelines. It serves as the ultimate goal in the pipeline selection process. The combined score is calculated as follows:

$$CS_{ij} = \alpha \cdot SC'ij + \beta \cdot CHI'ij + \gamma \cdot DBI'_{ij} \tag{4}$$

where the weights α, β, and γ reflect the relative importance of each metric in the combined score. With CS_{ij} for each pipeline, we choose base clustering pipelines that surpass a certain threshold T for the final ensemble clustering step. The user-defined selection policy SP influences the threshold T by using, for instance, the median of the CS vector or top n clustering pipelines.

Following the selection of the base clustering pipelines, their produced labels are encoded using the *Counter Encoding* method. This method treats each unique set of labels across the pipelines as a separate group. It replaces them with the count of their occurrences, effectively transforming the categorical labels into numerical features. These encoded labels are then subjected to PCA for dimensionality reduction. PCA is a sophisticated mathematical technique that can minimize the number of correlated variables in a given dataset. This process transforms these variables into a smaller set of uncorrelated variables called principal components. The first principal component captures as much data variability as possible, and each subsequent component captures as much of the remaining variability as possible. It is an effective tool for simplifying complex datasets and extracting meaningful information.

Finally, these transformed and reduced labels serve as input to the chosen meta-clustering algorithm M, which applies its mathematical formulation to produce the final ensemble clustering labels L'. The specific formulation varies based on the type of meta-clustering algorithm used. Overall, the mathematical formulations in the proposed ensemble clustering approach are span normalization, weighting, encoding, dimensionality reduction, and clustering, creating a robust and adaptive framework for clustering tasks.

3.5 Time and Space Complexity Analysis

Understanding the time and space complexity of the Counter Encoder-based Ensemble Clustering method is crucial for assessing its efficiency and scalability. The algorithm consists of multiple steps, each with its associated computational complexity. However, the time complexity of a base clustering pipeline (CP_{ij}) is the sum of the time complexities of its data preprocessing steps (P_i) and its base clustering algorithm (B_j). Similarly, the space complexity of a base clustering pipeline is the maximum of the space complexities of its data preprocessing steps and its base clustering algorithm.

The counter-encoding step transforms the categorical labels into numerical values. The time complexity of this operation is $O(n \cdot m)$, where n is the number of data instances and m is the number of selected base clustering pipelines. The space complexity remains $O(n \cdot m)$ as well, as we need to store the encoded labels for each data instance from each pipeline. If PCA is applied to reduce the dimensionality of the encoded labels, the time complexity would be $O(n \cdot m^2)$, as PCA requires calculating the covariance matrix and performing eigenvalue decomposition. The space complexity here depends on the number of principal components selected but would be at most $O(n{\cdot}m)$. Upon transforming the labels, the time complexity of fitting the meta-clustering algorithm M is subject to variation based on the selected algorithm, where the space complexity is $O(n)$ as we store the final clustering labels for each data instance.

By denoting the time complexity of M as T_M, we can represent a formal representation as follows. For instance, if M were to be instantiated as the KMeans algorithm, T_M would manifest as $O(I \times k \times n)$, where I represents the number of iterations, k the number of clusters, and n the number of data instances. Conversely, if M was a hierarchical clustering algorithm, the upper bound on the complexity could be $O(n^3)$, albeit more efficient variations could proffer a complexity of $O(n^2 \log n)$. Thus, the definitive characterization of T_M remains contingent upon the specific algorithmic nature of M. In practical applications, it is crucial to recognize the variability and choose an appropriate value for M that balances computational efficiency and clustering accuracy while also considering the operational constraints of the application domain.

In conclusion, the overall time complexity of the ensemble clustering approach is thus the time complexity of applying all base clustering pipelines plus the time complexity of the meta-clustering algorithm. The overall space complexity is the maximum of the space complexities of all base clustering pipelines and the space complexity of the meta clustering algorithm. Nevertheless, it is essential to note that the specific time and space complexity can vary greatly depending on the specific details of the datasets and the algorithms used. While the method may have relatively high computational requirements due to multiple stages of computations, its advantages in terms of robustness and flexibility can justify the computational cost in many practical scenarios, mainly when efficient clustering algorithms are used for base clustering pipelines and the meta-clustering process.

4 Experimental Setup and Evaluation

Our proposed ensemble clustering approach for workload categorization is evaluated using actual cloud data center workloads. Different clustering algorithms and data pipeline setups are employed. Additionally, various intrinsic performance metrics are used to assess the quality of the clustering results. More details are outlined as follows.

4.1 Clustering Evaluation Metrics

The clustering performance is crucial, as clustered data are often evaluated manually and subjectively to ascertain relevance. In the absence of knowledge regarding the valid clustering data label, various intrinsic metrics can be employed to assess the efficacy of the clustering technique. In this paper, we employed the most popular metrics described as follows.

4.1.1 Silhouette Coefficient

The SC score, introduced in [37], is used to evaluate the clustering quality by determining the degree of internal coherence and separation among clustering outcomes. This score clearly shows how clustering is performed within and between clusters. The score ranges from [-1 to 1], assessing the consistency within data clusters. A higher score indicates that the cluster data points are farther away from neighboring points, while a negative score implies that the cluster data points may have been assigned to the wrong group. When the score is around zero, the data points of the cluster are close to the boundaries and may overlap with other clusters. The overall clustering performance can also be assessed by computing the mean score. The SC score is calculated using the following formula.

$$SC = \frac{b-a}{max\left(a,b\right)} \tag{5}$$

where a represents the mean intra-cluster distance (*mean distance between a data point and all others within a cluster*), and b represents the mean nearest-cluster distance (*mean distance between a data point and all other points in other clusters*). The mean distance can be calculated using Euclidean distance. Typically, for two points p and q represented by Cartesian coordinates in n-dimensional Euclidean space, the distance $d(\mathbf{p},\mathbf{q})$ can be calculated as the following equation [42].

$$d(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2} \tag{6}$$

4.1.2 Calinski-Harabasz Index

The CHI, also known as the Variance Ratio Criterion, was introduced by Calinski in [7] and has since proven to be a valuable tool for assessing clustering outcomes. This metric assesses the dispersion ratio within a cluster to the average dispersion between clusters, which can be calculated by summing the squared distances. A higher CHI value signifies well-defined clustering with dense and distinct groups. The CHI can be easily computed using the following equation.

$$CHI = \frac{\text{tr}\left(B_k\right)}{\text{tr}\left(W_k\right)} \times \frac{n_E-k}{k-1} \tag{7}$$

where n_E represents the size of a given sample dataset E that has been clustered into k groups, $\text{tr}\left(B_k\right)$ is the trace matrix of the dispersion between clusters, and $\text{tr}\left(W_k\right)$ is the trace matrix of the dispersion within a cluster. The W_k and B_k matrixes can be calculated as follows.

$$W_k = \sum_{q=1}^{k}\sum_{x\in C_q}\left(x-c_q\right)\left(x-c_q\right)^T \tag{8}$$

$$B_k = \sum_{q=1}^{k} n_q\left(c_q-c_E\right)\left(c_q-c_E\right)^T \tag{9}$$

where C_q refers to a collection of data points within a specific cluster labeled as q, the variable n_q represents the total number of data points within that cluster, c_q indicates the center point of the same cluster, c_E refers to the center point of the entire sample dataset labeled as E, and T means the transpose of the calculated matrix.

4.1.3 Davies-Bouldin Index

The DBI is another evaluation metric for clustering algorithms, introduced in [25]. It estimates the similarity between two clusters by computing the ratio of within-cluster distances to the distance between the other clusters. The minimum value is zero, and the model with a lower value implies further apart and less scattered within the data points, designating better clustering performance. This index is calculated as follows.

$$DBI = \frac{1}{k}\sum_{i=1}^{k}\max_{i\neq j} R_{ij} \tag{10}$$

where k is the total number of clusters, R_{ij} is a measurement that represents how good the clustering scheme is between i^{th} and j^{th} clusters, which is calculated using the following equation.

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{11}$$

where s indicates the average distance between the data points of a cluster and its centroid value, which this calculation applies to the i^{th} and j^{th} clusters. The d_{ij} is the distance between i^{th} and j^{th} clusters centroids. The optimal partitioning clustering outcomes are determined by the values of small s and large d_{ij}, which provide a minimum DBI value. These factors are crucial in the clustering evaluation process to ensure that the resulting clusters are accurate and efficient.

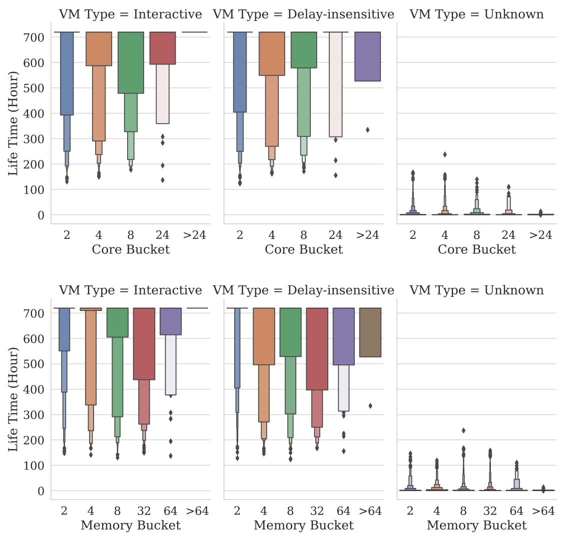
4.2 Workload Description and Initial Data Preparation

In this section, we delve into the specifics of the workload utilized in our research, derived from the Microsoft Azure Public Workload, and the methods employed for data preparation. We first introduce Microsoft Azure's role as a cloud computing platform and describe its functionality and capabilities. Then, we present the specifics of the dataset used in our study, including its origin, constituent elements, and key fields. Subsequently, we provide an overview of the standard data preparation process, encompassing characteristic identification, timestamp extraction, categorical to numerical conversion, and feature selection processes to ensure that only the most relevant and informative features are retained.

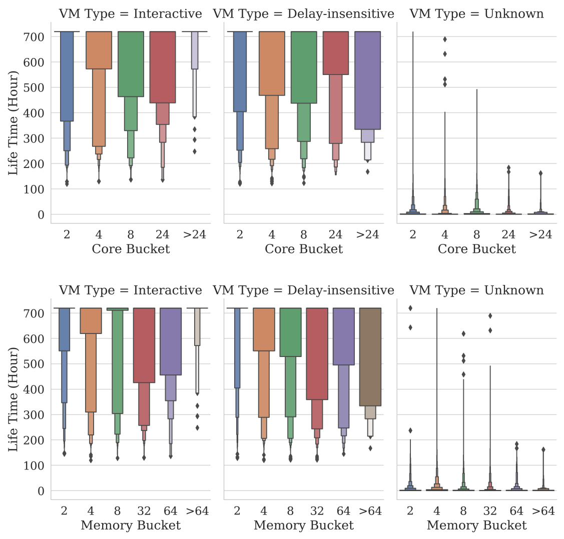
4.2.1 Microsoft Azure Public Workload

Microsoft Azure is a cloud computing platform enabling businesses to build, deploy, and manage cloud-based applications globally by providing computing, storage, analytics, networking, and other diverse cloud-based services [27]. Cloud users can submit multiple tasks to any regional data center using single or multiple subscriptions. Every task operates on a Virtual Machine (VM) within a deployment that supplies the necessary resources for each task to run efficiently, considering its specific requirements.

The dataset used in this study is derived from the Azure Public Dataset v2 [11], a rich and comprehensive trace from Azure's first-party VMs in specific geographical regions. This subset includes



(a) Sample dataset of 15,000 VMs



(b) Original dataset of 2,695,548 VMs

Fig. 2: Boxen plots illustrating the distribution and relationships among VM lifetime, VM type, CPU core, and memory buckets for selected sample and original datasets.

data from 2,695,548 VMs across 6,687 subscriptions collected over 30 consecutive days. The dataset schema contains key fields that describe various aspects of each VM. These include the subscriptions and deployments indexes, which are high cardinal nominal categories, including the timestamps in seconds of VM creation and deletion. The CPU core and memory buckets of each VM are represented as ordinal categories. The VM type is defined as a nominal category. Also, the schema includes CPU utilization, such as the maximum, average, and 95th percentile of maximum, which are presented as numerical features.

Given the extensive size and detail of the dataset, we employed a stratified sampling approach to ensure manageable computational requirements while maintaining representativeness. A sample of 15,000 VMs is chosen for our examination, comprising an even distribution across the three VM types: Interactive, Delay-insensitive, and Unknown. Specifically, 5,000 instances from each type are randomly selected, ensuring a diverse and comprehensive representation of workloads for our clustering tasks. The lifetime distribution of VM in hours for the selected sample and original datasets are depicted in Figure 2, taking into account the VM type along with the CPU core and memory buckets. In particular, the unknown VM type exhibits the shortest lifespan among the other types in both datasets.

4.2.2 Standard Preprocessing Pipeline

A standardized preprocessing pipeline is employed to prepare the dataset for subsequent analysis. It identifies additional characteristics, including the difference between the peak and average CPU usage (referred to as "Diff 1") and the difference between the 95^{th} percentile of maximal and average CPU usage (referred to as "Diff 2"). The pipeline also extracts various representative time-related data from the VM creation and deletion timestamps, including the year, month, day, hour, minute, and weekday indexes.

The categorical variables are then converted to numerical values using three types of encoders.

- *Ordinal Encoder:* This encoder assigns a distinct integer to each CPU core and memory bucket category based on their respective sizes. This encoder is commonly used in data analysis to simplify ordinal categories while preserving their magnitudes.
- *One-hot Encoder:* This encoder generates a binary representation for each VM type. This approach ensures that nominal categories are structured and organized, facilitating efficient handling and processing.
- *Counter Encoder:* This encoder replaces each category label with a numerical count of its appearances while maintaining the frequency-based ordering of the categories. Using this approach, we can better understand the distribution of these high cardinal nominal indexes within the dataset, facilitating more representative numerical details.

Finally, we identified features with high inter-correlation using the absolute Pearson correlation coefficient. Any features with a correlation value greater than the 0.99 threshold are deemed excessively correlated; of these, only the first feature in each correlated set is retained. Additionally, we eliminated features with zero variance, as they fail to provide meaningful differentiation. These feature selection measures ensure the retention of only the most relevant and informative features, thereby enhancing the efficiency and effectiveness of subsequent analyses.

4.3 Experimental Setup

This section details the methodologies employed for constructing and assessing clustering pipelines. Initially, the base clustering pipelines and selection setups are created using a combination of transformation and normalization methods and a PCA-based dimensionality reduction technique. Three clustering algorithms serve as base clustering methods, and the optimal pipelines are chosen using a combined score derived from various normalized metric scores. The selected pipelines form a refined list, serving as inputs to the ensemble clustering task. Furthermore, we introduce the ensemble clustering approach setups based on the counter encoder. Finally, we introduce a strategy to select the ideal number of clusters using a distortion score, quantifying the divergence between data points and their respective cluster centers. The optimal elbow value, indicative of the optimal cluster count, is automatically identified through the knee point detection algorithm, providing a complete setup for our ensemble clustering task.

4.3.1 Base Clustering Pipelines and Selection Setups

The clustering pipelines are built on different data pipeline setups, each utilizing a standard preprocessing pipeline as described in Section 4.2.2. The pipelines are formulated with PyCaret, a top-tier

Table 3: The number of PCA components and cumulative explained variance utilizing various combinations of transformation and normalization methods.

P	Transformation	Normalization	PCA Components	$\mathbb{C}$
P_1	None	None	3	0.99
P_2	None	Z-Score	8	0.84
P_3	None	Min-Max	8	0.89
P_4	None	MaxAbs	8	0.89
P_5	None	Robust	6	0.86
P_6	Yeo-Johnson	None	4	1.00
P_7	Yeo-Johnson	Z-Score	8	0.89
P_8	Yeo-Johnson	Min-Max	8	0.91
P_9	Yeo-Johnson	MaxAbs	8	0.91
P_{10}	Yeo-Johnson	Robust	8	0.90
P_{11}	Quantile	None	7	0.92
P_{12}	Quantile	Z-Score	8	0.87
P_{13}	Quantile	Min-Max	7	0.92
P_{14}	Quantile	MaxAbs	7	0.92
P_{15}	Quantile	Robust	6	0.90

Python ML library [2]. These pipelines use various transformation and normalization techniques and instances where they are not used.

Among the transformations utilized is the Yeo-Johnson method, a power transformation designed to stabilize variance and make the data more closely follow a Gaussian distribution. This transformation is remarkably versatile as it can handle zero and negative values, making it suitable for different data aspects in many datasets. Another transformation employed is the Quantile Transformation, which transforms the features to follow a uniform or a Gaussian distribution. This method is beneficial in mitigating the effects of outliers, hence improving the performance of subsequent clustering algorithms [36].

For normalization, we employed multiple techniques to ensure data consistency and comparability. The Z-Score normalization standardizes the features by removing the mean and scaling to the standard deviation, which results in features with a mean of zero and a variance of one. Min-Max normalization scales the data between a specific range, in our use case [0 to 1], ensuring every feature equivalently contributes to distance computations in clustering. MaxAbs normalization scales each feature by its maximum absolute value, ensuring that the maximal absolute value of each feature in the training set will be 1.0. It is noteworthy that MaxAbs does not shift or center the data, preserving any inherent sparsity in the dataset. The Robust normalization method, resilient to outliers, scales features using the median and the interquartile range, making it an optimal choice for datasets with notable outliers [36].

These setups may differ in the inclusion/exclusion of specific methods for each base clustering pipeline. The data pipelines incorporate Algorithm 3 to reduce the dimensionality of the original data. It applies a Kneedle method by setting True to the u parameter to determine the PCA components. In situations where the identification of the knee index is not feasible, we set a threshold of 95% explained variance v to attain the intended objective. This crucial step entails transforming the data into a lower-dimensional space while preserving most of the original variance, considering the selected transformation and normalization setups.

Table 3 shows the results of PCA components and cumulative explained variance ($\mathbb{C}$) for various combinations of transformation (None, Yeo-Johnson, and Quantile) and normalization methods (None, Z-Score, Min-Max, MaxAbs, and Robust). This table provides insights into how different combinations of methods can affect the optimal number of PCA components and the $\mathbb{C}$ that indicates the variance in the data explained by the selected components. It allows for comparing the performance of different setups and identifying the configurations that yield higher explained variance, serving as a reference for understanding the influence of the given setups.

Three clustering algorithms (KMeans [39], Hierarchical clustering - Agglomerative (with an early stop of tree construction) [33], and Mean-Shift [10]) are employed as base clustering methods using the implementation of sklearn [36]. Accordingly, Algorithm 1 generates all possible clustering pipelines based on the given setups. This process is considered the first step in the Algorithm 2 to formulate a comprehensive set of pipelines incorporating various setups.

In our selection of clustering algorithms, we are driven by a desire to encompass diverse algorithmic philosophies that cater to varied data characteristics and use cases. The KMeans algorithm, as elucidated by Sculley [39], offers a centroid-based clustering mechanism renowned for its scalability and efficiency, making it indispensable for large datasets. The Hierarchical clustering - Agglomerative method, showcased in [33], lends a tree-structured representation qualified for interpreting hierarchical relationships within data. Lastly, MeanShift, referenced in [10], complements the prior methods by offering a density-based clustering paradigm, adept at discerning clusters of arbitrary shapes, and excelling in scenarios with intricate data densities. These methods provide a robust and comprehensive toolkit, each contributing a unique perspective and addressing specific clustering challenges.

Nevertheless, Algorithm 2 selects the optimal base clustering pipelines using a combined score. Equation 4 calculates this score, with weightings assigned to each normalized metric score of the base clustering pipelines. The weights α, β, and γ are set to 34%, 33%, and 33%, respectively, indicating equal importance for all contributed metrics. In order to be considered for selection, pipelines must exceed a predetermined threshold T, which is determined by the selection policy SP. This paper has set T to the median of the combined score vector, allowing for a flexible practice in selecting the most appropriate pipelines for the final ensemble clustering task, enabling consideration of various data preprocessing perspectives.

In summary, Algorithm 1 generates a list of all potential clustering pipelines according to the given specifications, denoted as CP. Algorithm 2 accurately evaluates and chooses the best base clustering pipelines by considering their performance in various validity indices. The outcome is a refined list, denoted as CP', which comprises the top-performing pipelines chosen for the ultimate ensemble clustering solution.

4.3.2 Counter Encoder-based Ensemble Clustering Setup

As a meta-clustering method, we examined the same clustering algorithms used in building the base clustering pipelines (KMeans, Agglomerative (with an early stop of tree construction), and MeanShift) based on the implementation of sklearn [36]. Once the cluster labels are acquired for each selected base clustering pipeline, a counter encoder based on the implementation introduced in [32] transforms them into a numerical flavor. The cluster labels for each pipeline are replaced with their frequency, resulting in the encoded labels L_{en}. Similar to the dimensionality reduction that is used in building the data pipelines, we use Algorithm 3 as a second-stage encoding for the L_{en} to reduce the dimensionality into a lower space that maintains most of the original variance of the base clustering outcomes. Setting True to the u parameter applies a Kneedle method to determine the PCA components. If it is unattainable to accurately and precisely determine the knee index, the explained variance threshold is used instead, in which v is set to 95% to achieve the desired goal.

4.3.3 Selecting the Ideal Number of Clusters

Determining the optimal number of clusters is among the most challenging aspects of the clustering process. For example, the KMeans and hierarchical clustering algorithms require the number of clusters to be determined to group the data points accordingly. The distortion score helps identify the best number of clusters needed for clustering algorithms that require it. The score calculation involves determining the degree of divergence between each data point and its respective cluster center, with the result being the sum of squared distances. A higher score means the clustering outcome is less effective, while a lower score indicates the best effectiveness. Optimal cluster sizes can be determined by plotting the score against different cluster sizes and identifying the elbow point, the point at which the plot begins to level off. According to [4], the distortion score is mathematically expressed as:

$$D = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2 \tag{12}$$

where k is the number of clusters, C_i is the set of data points allocated to cluster i, x data points $\in C_i$, and μ_i is the centroid of cluster i. The goal of clustering is to minimize the score as much as possible. Using this score in this study, we employed the knee point detection algorithm proposed in [38] to automatically identify the optimal elbow value for the number of clusters k, simplifying this critical process.

4.4 Results and Discussion

4.4.1 Base Clustering Pipelines and Selection Results

We evaluated different data pipelines incorporating KMeans, Agglomerative, and MeanShift as base clustering models, analyzing their effectiveness with and without the implementation of PCA for dimensionality reduction, with setups as specified in Table 3.

The use of KMeans as a base clustering model, as shown in Table 4a and Table 4b, resulted in an overall enhancement of the structure of clusters with the application of PCA, as evidenced by the improved SC. In particular, P_1 showcased the most substantial improvement, from 0.7117 without PCA to 0.7539 with PCA. Furthermore, the CHI increased across several pipelines when PCA is used, suggesting denser and more separated outcomes. For instance, P_1's CHI value rose from 99388.6100 without PCA to 130024.7564 with PCA. However, some pipelines such as P_2, P_7, and P_{12} showed only minor improvements. The DBI, which has lower values indicating better clustering, generally decreased with PCA, signifying improved cluster compactness and separation. The most significant improvement is in P_1, as its DBI value decreased from 0.4755 to 0.4165.

In comparing Agglomerative clustering pipelines, we observed consistent improvements across all evaluation metrics when PCA is applied. As shown in Table 5a and Table 5b, there is an upward trend in the SC values with the application of PCA, indicating more coherent clusters. For instance, in P_1, the SC rose from 0.7101 to 0.7275, improving cluster quality outcomes. The CHI values also saw a general increase, suggesting that the clusters are better separated and denser with the implementation of PCA. Specifically, in pipeline P_1, the CHI increased from 98356.3247 to 103705.2861. Moreover, the DBI values generally decreased with PCA, indicating an enhancement in cluster compactness and separation. For example, the DBI in P_1 decreased from 0.4758 to 0.4620, showing a better-defined cluster structure.

By comparing various data pipelines using MeanShift clustering as a base model, we can observe a similar trend to KMeans and Agglomerative clustering pipelines, with PCA generally improving clustering performance across all evaluation metrics. As shown in Table 6a and Table 6b, pipelines with PCA show higher SC values, indicating that the PCA-reduced data produced more distinct and compact clusters. For instance, in P_1, the SC increased from 0.7117 to 0.7539. The CHI values also generally increased, indicating that clusters are better separated and denser when the data underwent PCA. It is particularly noticeable in pipeline P_1, where the CHI increased from 99382.9519 to 129966.3486 after PCA reduction. The DBI values generally showed a decrease with PCA reduction, demonstrating an improvement in the compactness and separation of clusters. In P_1, the DBI decreased from 0.4756 to 0.4165.

However, not all clustering pipelines benefited from PCA reduction and transformation using the MeanShift algorithm based on the default sklearn implementation, which estimates the bandwidth parameter that dictates the region's size to search through based on a heuristic technique. Pipelines P_2, P_{10}, and P_{12} are unable to find meaningful clusters (k=1), with SC, CHI, and DBI all at 0.0. In the case of P_{15}, it is observed that the number of clusters increased from 11 to 14 after applying PCA. This finding could complicate the interpretation of the results and render them less desirable for specific applications.

Although PCA improved the overall clustering performance in many instances, it is worth noting that not all pipelines saw enhancement, especially when the original data structure is not well suited for linear transformation. The performance largely depends on the specific nature of the data

Table 4: Comparison of various pipelines using KMeans as a base clustering model.

(a) Data pipelines without PCA reduction

P	k	SC	CHI	DBI
P_1	4	0.7117	99388.6100	0.4755
P_2	5	0.2186	3620.2148	1.8650
P_3	5	0.3437	7154.0491	1.5209
P_4	5	0.3441	7162.4272	1.5148
P_5	4	0.3514	7669.0127	1.3448
P_6	5	0.6020	18893.5850	0.9479
P_7	5	0.2185	4133.8027	1.7617
P_8	5	0.3037	6794.3161	1.4906
P_9	5	0.3041	6726.1254	1.5256
P_{10}	5	0.2264	4206.1589	1.5352
P_{11}	4	0.4624	11827.3598	1.2293
P_{12}	5	0.2045	3627.9506	1.8595
P_{13}	4	0.4624	11827.3598	1.2293
P_{14}	4	0.4624	11827.3598	1.2293
P_{15}	5	0.3264	4040.4576	1.3211

(b) Data pipelines with PCA reduction

P	k	SC	CHI	DBI
P_1	4	0.7539	130024.7564	0.4165
P_2	4	0.2626	5793.9210	1.6221
P_3	5	0.4105	10377.3437	1.2641
P_4	5	0.4110	10401.4950	1.2573
P_5	4	0.4471	11913.9534	1.0607
P_6	5	0.6079	19117.8225	0.9420
P_7	5	0.2527	5186.1798	1.5283
P_8	5	0.3537	8986.5988	1.2605
P_9	5	0.3560	8869.1916	1.2949
P_{10}	5	0.2577	5349.1903	1.3634
P_{11}	4	0.5415	16482.6017	1.0515
P_{12}	5	0.2427	4635.1435	1.6448
P_{13}	4	0.5415	16482.6017	1.0515
P_{14}	4	0.5415	16482.6017	1.0515
P_{15}	5	0.4000	5080.3153	1.1497

Table 5: Comparison of various pipelines using Agglomerative as a base clustering model.

(a) Data pipelines without PCA reduction

P	k	SC	CHI	DBI
P_1	4	0.7101	98356.3247	0.4758
P_2	6	0.1891	3033.9224	1.8813
P_3	4	0.3747	7851.7142	1.4039
P_4	4	0.3746	7845.6106	1.4036
P_5	4	0.3037	6929.9167	1.5351
P_6	5	0.5753	16729.9033	1.0706
P_7	5	0.1953	3749.2588	1.8909
P_8	4	0.3373	7569.2856	1.4782
P_9	4	0.3403	7457.5812	1.4719
P_{10}	5	0.1630	3603.6385	1.7260
P_{11}	5	0.4942	11072.0702	1.1251
P_{12}	6	0.1779	2889.3170	1.8894
P_{13}	5	0.4942	11072.0702	1.1251
P_{14}	5	0.4942	11072.0702	1.1251
P_{15}	5	0.3278	3504.2283	1.4003

(b) Data pipelines with PCA reduction.

P	k	SC	CHI	DBI
P_1	4	0.7275	103705.2861	0.4620
P_2	5	0.2462	4701.0041	1.7023
P_3	5	0.4041	10154.6949	1.2841
P_4	4	0.4401	10905.5294	1.2039
P_5	4	0.3858	10287.4223	1.1625
P_6	4	0.5823	17582.1233	1.0915
P_7	5	0.2337	4799.6949	1.6786
P_8	4	0.3846	9646.3802	1.3170
P_9	4	0.3886	9505.3736	1.3100
P_{10}	5	0.1930	4501.7227	1.4896
P_{11}	5	0.5806	16557.4135	0.9216
P_{12}	6	0.2215	3904.0858	1.6708
P_{13}	5	0.5806	16557.4135	0.9216
P_{14}	5	0.5806	16557.4135	0.9216
P_{15}	5	0.3457	4202.0616	1.2989

transformation and normalization methods used within each pipeline. Thus, applying PCA should be considered in conjunction with these factors for optimal performance. Accordingly, the base clustering pipelines that incorporate applying PCA are scored and evaluated using Algorithm 2 - *Base Clustering Pipeline Selection*. This algorithm ranks and selects the best clustering pipelines based on their combined score of multiple validity indices representing their clustering performance. The output is a subset CP' from the initial grouping of base clustering pipelines, consisting of the highest-performing pipelines chosen for the final ensemble clustering solution.

Table 7 shows the selected clustering pipelines CP', sorted by their combined scores. These pipelines are selected based on a threshold set to the median of the combined score vector. Here, the combined score measures the overall performance of each pipeline considering all three normalized metrics (SC', CHI', and DBI'). As seen in the table, the pipeline P_1 with KMeans as the base model has the highest combined score and thus is the top selected model for the meta-clustering in the ensemble process. It is closely followed by the pipeline P_1 with MeanShift and Agglomerative as the base models, which also have high combined scores. On the lower end of the selection

Table 6: Comparison of various pipelines using MeanShift as a base clustering model.

(a) Data pipelines without PCA reduction

P	k	SC	CHI	DBI
P_1	4	0.7117	99382.9519	0.4756
P_2	1	0.0000	0.0000	0.0000
P_3	2	0.4027	10750.5777	1.1190
P_4	2	0.4021	10718.7755	1.1199
P_5	2	0.6520	8809.2248	0.5717
P_6	2	0.6390	27583.7521	0.6433
P_7	2	0.3412	7833.7724	1.3410
P_8	2	0.4066	11688.5227	1.0823
P_9	2	0.4011	11176.1000	1.0955
P_{10}	1	0.0000	0.0000	0.0000
P_{11}	1	0.0000	0.0000	0.0000
P_{12}	1	0.0000	0.0000	0.0000
P_{13}	1	0.0000	0.0000	0.0000
P_{14}	1	0.0000	0.0000	0.0000
P_{15}	11	0.3375	918.1624	1.2530

(b) Data pipelines with PCA reduction.

P	k	SC	CHI	DBI
P_1	4	0.7539	129966.3486	0.4165
P_2	1	0.0000	0.0000	0.0000
P_3	2	0.4426	13342.4127	0.9922
P_4	2	0.4420	13304.9449	0.9928
P_5	3	0.4553	12577.8194	0.9602
P_6	3	0.5849	18447.7587	1.2694
P_7	2	0.3685	9344.9197	1.2211
P_8	2	0.4383	13870.9523	0.9830
P_9	2	0.4321	13218.1198	0.9952
P_{10}	1	0.0000	0.0000	0.0000
P_{11}	3	0.5167	16090.8737	0.8242
P_{12}	1	0.0000	0.0000	0.0000
P_{13}	3	0.5167	16090.8737	0.8242
P_{14}	3	0.5167	16090.8737	0.8242
P_{15}	14	0.3356	914.3008	1.1252

Table 7: The selected clustering pipelines based on their combined scores: Results from Algorithm 2.

P	Base Model	k	SC'	CHI'	DBI'	CS
P_1	KMeans	4	100.0 %	100.0 %	75.53 %	91.93 %
P_1	MeanShift	4	100.0 %	99.96 %	75.53 %	91.91 %
P_1	Agglomerative	4	96.5 %	79.76 %	72.86 %	83.17 %
P_6	KMeans	5	80.63 %	14.7 %	44.66 %	47.01 %
P_{14}	Agglomerative	5	77.01 %	12.73 %	45.86 %	45.52 %
P_{13}	Agglomerative	5	77.01 %	12.73 %	45.86 %	45.52 %
P_{11}	Agglomerative	5	77.01 %	12.73 %	45.86 %	45.52 %
P_{13}	MeanShift	3	68.54 %	12.38 %	51.58 %	44.41 %
P_{11}	MeanShift	3	68.54 %	12.38 %	51.58 %	44.41 %
P_{14}	MeanShift	3	68.54 %	12.38 %	51.58 %	44.41 %
P_6	Agglomerative	4	77.24 %	13.52 %	35.88 %	42.56 %
P_{11}	KMeans	4	71.83 %	12.68 %	38.23 %	41.22 %
P_{13}	KMeans	4	71.83 %	12.68 %	38.23 %	41.22 %
P_{14}	KMeans	4	71.83 %	12.68 %	38.23 %	41.22 %
P_6	MeanShift	3	77.58 %	14.19 %	25.43 %	39.45 %
P_5	MeanShift	3	60.39 %	9.67 %	43.59 %	38.11 %
P_8	MeanShift	2	58.14 %	10.67 %	42.25 %	37.23 %
P_3	MeanShift	2	58.71 %	10.26 %	41.71 %	37.11 %
P_4	MeanShift	2	58.63 %	10.23 %	41.68 %	37.06 %
P_9	MeanShift	2	57.32 %	10.17 %	41.54 %	36.55 %
P_5	KMeans	4	59.3 %	9.16 %	37.69 %	35.63 %

spectrum, we see that the pipelines using MeanShift on P_8, P_3, P_4, and P_9, and KMeans on P_5 have combined scores around 35-37%, still making the cut threshold for the ensemble process. This table represents the most promising combination of preprocessing and clustering algorithms for the final ensemble process. The aim is to combine these selected pipelines to leverage their strengths, thus creating an ensemble model with potentially better performance than any single model.

4.4.2 Counter Encoder-based Ensemble Clustering Results

The efficacy of the proposed ensemble clustering approach is evaluated using the selected clustering

Table 8: Comparative performance of various meta-models with the proposed ensemble clustering: A juxtaposition of outcomes from base clustering and original data.

	Comparison with	Meta Model	k	PCA	SC	CHI	DBI
(a)	Respect to the Base Clustering Outcomes	KMeans	4	False	0.7297	25159.6013	0.8914
		KMeans	4	True	0.7371	27000.0676	0.8669
		Agglomerative	4	False	0.7223	23192.8589	0.8567
		Agglomerative	4	True	0.7171	23115.8426	0.7872
		MeanShift	64	False	0.9254	44697.3547	**0.1536**
		MeanShift	44	True	**0.9283**	**61871.5461**	0.3304
(b)	Respect to the Original Data	KMeans	4	False	0.7605	102978.2505	4.4805
		KMeans	4	True	0.7605	102978.2505	4.4805
		Agglomerative	4	False	0.7762	148892.2541	12.7522
		Agglomerative	4	True	**0.7877**	**188882.5009**	**2.8508**
		MeanShift	64	False	0.0793	22850.4584	96.6447
		MeanShift	44	True	0.2371	29348.1181	217.7144

pipelines CP', shown in Table 7, employing several meta-clustering models, including KMeans, Agglomerative, and MeanShift algorithms. Our approach's effectiveness is evaluated using three clustering metrics (SC, CHI, and DBI) and with and without implementing dimensionality reduction based on PCA to the base clustering labels as a second encoding stage.

For clarity, it is imperative to note that the metrics presented here have been intentionally presented in their unnormalized form. This decision is based on the rationale that a direct and unambiguous juxtaposition of raw scores across distinct models would be more illuminating. The proposed normalization methodology is meticulously formulated to serve the nuanced requirements of the clustering pipeline selection context rather than the general application.

Table 8a details the results of our proposed ensemble clustering approach concerning the base clustering outcomes. As we can see, each ensemble model version performs differently in terms of the used evaluation metrics. The MeanShift meta model with PCA produces the highest SC (0.9283) and CHI (61871.5461), indicating high separation and compactness among the clusters. Also, its DBI (0.3304), though not the lowest, is relatively low, which means there is less overlap between clusters. This finding suggests that the MeanShift meta model with PCA and 44 clusters yields the best overall performance considering the base clustering outcomes. However, it is essential to note that MeanShift results in a relatively high number of clusters (k) compared to KMeans and Agglomerative models. This case can increase the complexity of interpretation and may only be appropriate for some use cases. By utilizing the KMeans and Agglomerative models, we can achieve fewer clusters and fairly excellent scores, even without implementing dimensionality reduction. This outcome makes them a suitable and more accessible option for interpretation.

In Table 8b, we present the results of our ensemble clustering approach, taking into account the original data projected into singular dimensions using PCA, as well as the clustering outcomes of all the meat clustering models with and without PCA consideration. The results show that using hierarchical clustering with PCA reduction resulted in the best performance. Specifically, it produced the highest SC (0.7877) and CHI (188882.5009), which indicates a high degree of separation and compactness among the clusters. The DBI (2.8508) is the lowest among all models, suggesting minimal overlap between the clusters. Meanwhile, using the MeanShift model shows considerably lower SC and CHI scores and notably higher DBI with and without PCA consideration. This case indicates that the MeanShift as a meta-clustering model produces less cohesive and more overlapping clusters than the other models due to the highest cluster numbers.

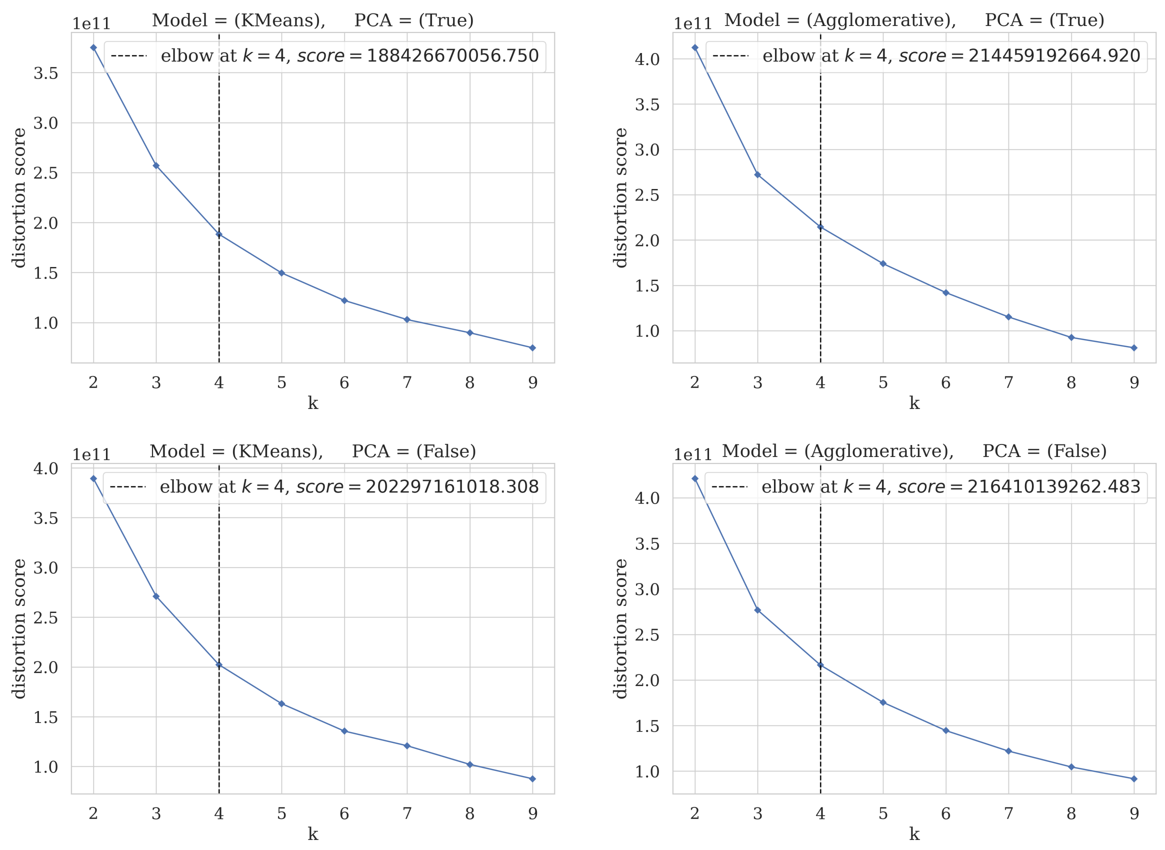


Fig. 3: K-Elbow plots display the distortion score and fitting time in seconds for KMeans and Agglomerative clustering methods with and without applying dimensionality reduction using PCA.

Interestingly, the PCA and non-PCA variants of the KMeans model yield identical results concerning the projected original data into a singular dimension. It suggests a robust inherent structure in the data, where PCA's dimensionality reduction does not significantly alter the primary clustering patterns. Such congruence implies that the clusters identified are defined by substantial structural differences that remain consistent irrespective of the PCA transformation. The stability in clustering outcomes across both variants underscores the strong, intrinsic data patterns and the KMeans model's sensitivity to these overarching patterns.

Nevertheless, each model has distinct strengths and limitations. The selection between them should be judiciously based on the intricacies of the dataset in use and the overarching goals of the clustering endeavor. For instance, in a dataset where the first few principal components primarily capture the variance, the PCA variant might be more efficient by reducing computational costs without sacrificing clustering accuracy. Conversely, for a dataset where vital information is dispersed across multiple components, the non-PCA version could be better suited to capture the nuances in the data.

However, in Figure 3, we present the elbow analysis for KMeans and Agglomerative meta clustering models, both with and without dimensionality reduction. The graph displays the distortion score for various cluster numbers (k) ranging from 2 to 10. Importantly, the chosen k is shown as a dashed line for each model.

Figure 4 visually represents the intercluster distance in a two-dimensional space. This figure showcases the outcomes of KMeans and Mean-Shift meta-clustering models, both with and without dimensionality reduction. The feature space is embedded using a Multi-Dimensional Scaling (MDS) algorithm presented in [4]. It aims to

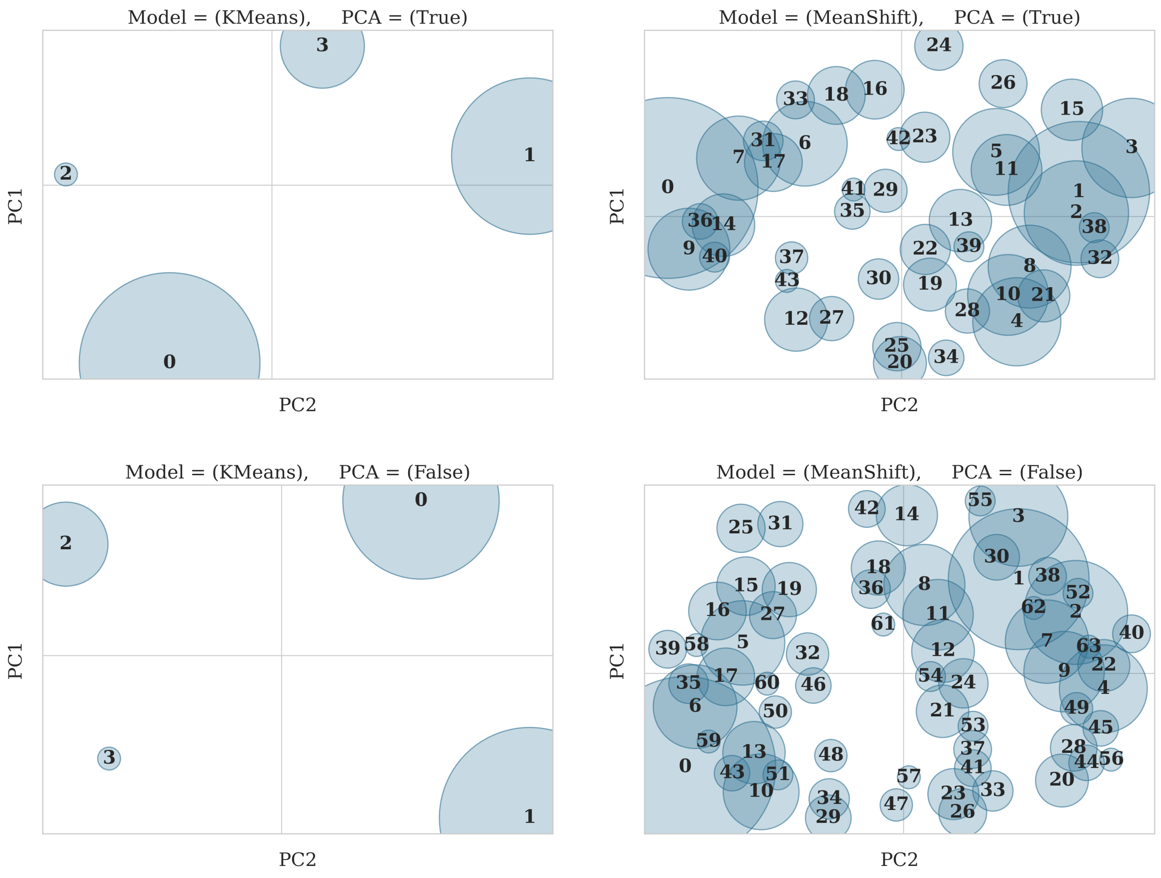


Fig. 4: Intercluster distance map for KMeans and MeanShift clustering methods with and without implementing dimensionality reduction, embedded via the MDS.

transform the base clustering outcomes into a low-dimensional space while preserving the pairwise distances between the original points as much as possible. Correspondingly, it is paramount to underscore that the overlapping of two clusters in the 2D space does not inherently signify an overlap in the original feature space. The clusters' size indicates membership, allowing for a visual gauge of the relative importance of each cluster. This visual aid assists in identifying the most significant clusters in the data and facilitates decisions based on their unique traits.

Lastly, Figure 5 displays silhouette analysis plots for KMeans meta clustering both with and without the application of dimensionality reduction. The dashed line in the figure indicates the average scores, which are observed to be nearly equal. It is noteworthy to mention that despite the dimensionality reduction, the clustering results of both versions are congruent. This consistency underpins the robustness of the underlying data structure and the model's ability to discern intrinsic patterns, regardless of the application of dimensionality reduction.

4.5 Merits, Challenges, and Future Prospects

In this section, we explore the proposed ensemble clustering model in-depth to demonstrate its inherent benefits and potential challenges, thereby determining its overall efficacy. By synthesizing our observations, we aim to chart the anticipated implications of this model on the broader research landscape and identify potential pathways for subsequent investigations.

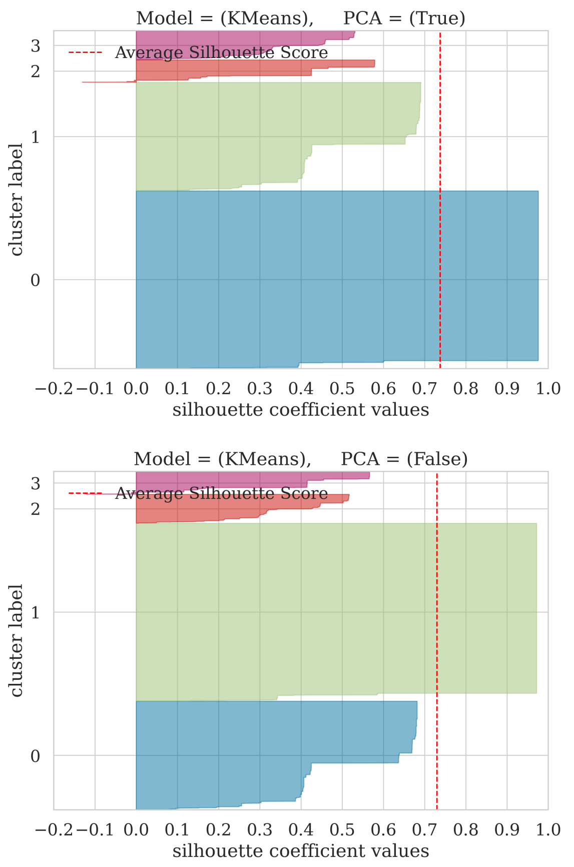


Fig. 5: Silhouette analysis plot of KMeans clustering method with and without implementing dimensionality reduction.

4.5.1 Merits of the Model

Cloud data center management is fundamentally dependent on effective workload identification and classification. The proposed model addresses this by illuminating latent workload classes and revealing underlying categorization perspectives within the data center. This insight facilitates a nuanced understanding of the distinct characteristics inherent in active workloads, thereby streamlining subsequent management processes.

A notable distinction of our ensemble model is its ability to employ homogeneous or heterogeneous base clustering models. Such flexibility permits the assimilation of diverse clustering outcomes and segmentation paradigms, setting our approach apart from specific ensemble clustering methods that may present constraints.

For enhanced accuracy in selecting optimal base-clustering models and preprocessing pipelines, our method adopts a combined score that amalgamates the merits of the SC, CHI, and DBI metrics. Moreover, an intrinsic feature of our model is its autonomous cluster number determination, which proves invaluable in settings that demand grouping workloads into coherent segments for informed decision-making. This nuance becomes pivotal when navigating expansive and multifaceted workloads.

Our ensemble model stands out in high-dimensional spaces, where specific density-based approaches falter, maintaining unwavering robustness and consistent performance. Unlike some grid-based techniques that may offer superficial insights, our method dives deeper, ensuring meticulous data exploration.

Remarkably, our model excels in intricate pattern recognition, drawing parallels to the sophistication of deep learning algorithms while sidestepping the opacity often associated with ”black-box” models. This transparency and user-centric design empower practitioners to harness the model's potential without contending with undue intricacies. In conclusion, our model masterfully balances performance and resource optimization, offering efficient data processing while judiciously conserving computational assets.

4.5.2 Inherent Limitations

While the preceding section delineates the commendable attributes of the proposed ensemble clustering model, expounding upon its intrinsic limitations is essential. Such a candid exposition grounds the model's applicability and fosters avenues for future enhancement.

At the outset, the most palpable concern is the augmented computational demands. Incorporating many base clustering algorithms, albeit advantageous in capturing nuanced clustering perspectives, naturally escalates computational overhead. It is especially pronounced when the model encounters voluminous datasets, warranting judicious consideration of computational resources.

Corollary to this is the issue of escalated memory utilization, necessitated by the model's iterative clustering approach. The need to persistently store intermediate outcomes, especially in the context of diverse algorithms, intensifies the memory footprint. Furthermore, integrating multiple algorithms augments the hyperparameter domain, potentially introducing complexities in model tuning. Discerning an optimal balance between the diversity of base clusterings and their inherent quality emerges as a nuanced endeavor.

Despite the model's emphasis on transparency and intelligibility, the depth of its multifaceted design may occasionally introduce challenges in direct interpretation. Moreover, the ensemble's efficacy is intrinsically tethered to the performance of its constituent algorithms. A subpar base clustering can attenuate the ensemble's collective output. Ensembles underpinned by redundant or homogenized base clusterings might not outperform standalone clustering techniques.

Finally, as the model burgeons, both in terms of data magnitude and algorithmic plurality, the evaluation paradigm becomes more intricate. A comprehensive, methodologically sound evaluation framework is paramount to distill the model's nuanced performance metrics. While not diminishing the model's merits, these limitations underscore the importance of continual refinement and iterative research.

4.5.3 Future Implications and Extensions

The deliberative exploration of the ensemble clustering model's virtues and constraints furnishes pivotal insights for prospective directions. The following enumerates potential trajectories shaped by the model's analytical acumen and its ramifications in cloud computing.

Central to the future direction of the proposed model is the drive for **Optimization and Efficiency in Model Execution**. The computational intensity of the model necessitates an exploration of algorithmic parallelization or perhaps the deployment of more advanced data structures that promise efficient computation. Complementing this is the need to address the model's robust memory demands, where strategies like dynamic memory allocation or leveraging data compression algorithms might offer viable solutions. Furthermore, our model's ever-expanding landscape of hyperparameters calls for innovative automated tuning strategies. Techniques such as Bayesian optimization offer the potential to refine the tuning process, ensuring its efficiency.

Another pillar of prospective enhancements hinges on **Model Refinement and Interpretability**. A sustained emphasis on model transparency remains indispensable. Introducing tools that provide deeper insights into the ensemble's decision-making processes can elevate its comprehensibility, catering to a user-centric approach. To bolster the ensemble's robustness, continually refining its base algorithms is imperative. Embracing novel clustering methodologies for specific cloud workloads can offer unprecedented advantages. Also, as the intricacies of model evaluation burgeon, the design and implementation of comprehensive evaluation methodologies become paramount. Such frameworks, which meld diverse performance metrics, will serve as a holistic lens to appraise the model's prowess.

Lastly, the transition from the theoretical domain to real-world applications is crystallized in the **Practical Implications for Cloud Environments** and **Bridging Theory and Practice**. Insights drawn from the model's cluster dynamics stand to revolutionize resource allocation paradigms in varied cloud architectures. Such knowledge can also be instrumental in pioneering predictive maintenance schedules, ensuring optimized load distribution for cloud operations. The rich tapestry of cluster diversity beckons refinements in cloud service provisions, aligning them more harmoniously with evolving client demands. However, the theoretical merits of the model need empirical validation. Thus, the emphasis on real-world, cloud-focused case studies and simulations is inevitable, as they hold the key to underscoring the practical significance of our model.

These delineated future directions underscore the ensemble clustering model's expansive potential for innovation and enhancement. They echo the symbiosis of academic ingenuity with pressing cloud-centric challenges, spotlighting the model's multi-dimensional significance.

5 Conclusion

In this paper, we have introduced a novel ensemble clustering technique for categorizing cloud workloads. The proposed approach, coupling multiple data preprocessing pipelines with diverse base clustering learners, has demonstrated remarkable potential for uncovering and capturing complex categorization perspectives. Through rigorous testing using real-world trace data from Microsoft Azure, we have provided empirical evidence of our approach's effectiveness. A unique combined scoring method is proposed to select the most influential models and preprocessing setups, providing valuable insights into the performance of various combinations of clustering algorithms and data preprocessing techniques. Our findings pave the way for a new perspective in managing cloud resources, whereby an advanced ensemble clustering approach can effectively navigate the multifaceted nature of cloud workloads.

Future research should explore additional ways to enhance this approach, incorporating additional machine learning techniques or fine-tuning the preprocessing pipelines to accommodate the evolving nature of cloud workloads. Additionally, investigating the applicability of our approach in different cloud environments beyond Microsoft Azure may provide further insight into its universal applicability and robustness. As cloud computing grows in complexity and scale, it becomes crucial to discover the latent categorization perspectives inherent in cloud data center workloads. By doing so, organizations can gain deeper insight, facilitating improved decision-making processes in resource allocation, performance optimization, and workload balancing. Consequently, this can significantly increase the overall operational efficiency of the cloud data center, translating into improved business results and customer experiences. Therefore, the methodology presented in this paper provides a robust framework for optimizing future cloud resources.

References

[1] Abdelsamea A, Hemayed EE, Eldeeb H, et al (2014) Virtual machine consolidation challenges: A review. International Journal of Innovation and Applied Studies 8(4):1504

[2] Ali M (2020) PyCaret: An open source, low-code machine learning library in Python. URL https://www.pycaret.org, pyCaret version 1.0

[3] Askarizade Haghighi M, Maeen M, Haghparast M (2019) An energy-efficient dynamic resource management approach based on clustering and meta-heuristic algorithms in cloud computing iaas platforms: Energy efficient dynamic cloud resource management. Wireless Personal Communications 104:1367–1391

[4] Bengfort B, Bilbro R (2019) Yellowbrick: Visualizing the Scikit-Learn Model Selection Process. The Journal of Open Source Software 4(35). https://doi.org/10.21105/joss.01075

[5] Bharany S, Badotra S, Sharma S, et al (2022) Energy efficient fault tolerance techniques in green cloud computing: A systematic survey and taxonomy. Sustainable Energy Technologies and Assessments 53:102613

[6] Bhattacharjee P, Mitra P (2021) A survey of density based clustering algorithms. Frontiers of Computer Science 15:1–27

[7] Caliński T, Harabasz J (1974) A dendrite method for cluster analysis. Communications in Statistics-theory and Methods 3(1):1–27

[8] Calzarossa MC, Massari L, Tessera D (2016) Workload characterization: A survey revisited. ACM Computing Surveys (CSUR) 48(3):1–43

[9] Caruana R, Elhawary M, Nguyen N, et al (2006) Meta clustering. In: Sixth International Conference on Data Mining (ICDM'06), IEEE, pp 107–118

[10] Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5):603–619. https://doi.org/10.1109/34.1000236

[11] Cortez E, Bonde A, Muzio A, et al (2017) Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In: Proceedings of the 26th Symposium on Operating Systems Principles. ACM, pp 153–167, https://doi.org/10.1145/3132747.3132772

[12] Dezhabad N, Ganti S, Shoja G (2019) Cloud workload characterization and profiling for resource allocation. In: 2019 IEEE 8th international conference on cloud networking (CloudNet), IEEE, pp 1–4

[13] Dong X, Yu Z, Cao W, et al (2020) A survey on ensemble learning. Frontiers of Computer Science 14:241–258

[14] Estrada R, Valeriano I, Aizaga X (2023) Cpu usage prediction model: A simplified vm clustering approach. In: Conference on Complex, Intelligent, and Software Intensive Systems, Springer, pp 210–221

[15] Gao J, Wang H, Shen H (2020) Machine learning based workload prediction in cloud computing. In: 2020 29th international conference on computer communications and networks (ICCCN), IEEE, pp 1–9

[16] Ghobaei-Arani M, Shahidinejad A (2021) An efficient resource provisioning approach for analyzing cloud workloads: a metaheuristic-based clustering approach. The Journal of Supercomputing 77(1):711–750

[17] Gill SS, Tuli S, Toosi AN, et al (2020) Thermosim: Deep learning based framework for modeling and simulation of thermal-aware resource management for cloud computing environments. Journal of Systems and Software 166:110596

[18] Gu Z, Tang S, Jiang B, et al (2021) Characterizing job-task dependency in cloud workloads using graph learning. In: 2021 IEEE international parallel and distributed processing symposium workshops (IPDPSW), IEEE, pp 288–297

[19] Gupta S, Muthiyan N, Kumar S, et al (2017) A supervised deep learning framework for proactive anomaly detection in cloud workloads. In: 2017 14th IEEE India Council International Conference (INDICON), IEEE, pp 1–6

[20] Hameed A, Khoshkbarforoushha A, Ranjan R, et al (2016) A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. Computing 98:751–774

[21] Ikhlasse H, Benjamin D, Vincent C, et al (2022) Multimodal cloud resources utilization forecasting using a bidirectional gated recurrent unit predictor based on a power efficient stacked denoising autoencoders. Alexandria Engineering Journal 61(12):11565–11577

[22] Ismaeel S, Al-Khazraji A, Miri A (2019) An efficient workload clustering framework for large-scale data centers. In: 2019 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO), IEEE, pp 1–5

[23] Jayaprakash S, Nagarajan MD, Prado RPd, et al (2021) A systematic review of energy management strategies for resource allocation in the cloud: Clustering, optimization and machine learning. Energies 14(17):5322

[24] Jivrajani A, Raghu D, Apoorva K, et al (2016) Workload characterization and green scheduling on heterogeneous clusters. In: 2016 22nd Annual International Conference on Advanced Computing and Communication (ADCOM), IEEE, pp 3–8

[25] Karo IMK, MaulanaAdhinugraha K, Huda AF (2017) A cluster validity for spatial clustering based on davies bouldin index and polygon dissimilarity function. In: 2017 Second International Conference on Informatics and Computing (ICIC), IEEE, pp 1–6

[26] Katal A, Dahiya S, Choudhury T (2023) Workload characterization and classification: A step towards better resource utilization in a cloud data center. Pertanika Journal of Science & Technology 31(5)

[27] Kotas C, Naughton T, Imam N (2018) A comparison of amazon web services and microsoft azure cloud platforms for high performance computing. In: 2018 IEEE International Conference on Consumer Electronics (ICCE), IEEE, pp 1–4

[28] Kotsiantis SB, Kanellopoulos D, Pintelas PE (2006) Data preprocessing for supervised

leaning. International Journal of Electrical and Computer Engineering 1:111–117

[29] Li K, Cao X, Ge X, et al (2020) Metaheuristic optimization-based two-stage residential load pattern clustering approach considering intra-cluster compactness and inter-cluster separation. IEEE Transactions on Industry Applications 56(4):3375–3384

[30] Liang Y, Chen K, Yi L, et al (2023) Degtec: A deep graph-temporal clustering framework for data-parallel job characterization in data centers. Future Generation Computer Systems 141:81–95

[31] Liu C, Liu C, Shang Y, et al (2017) An adaptive prediction approach based on workload pattern discrimination in the cloud. Journal of Network and Computer Applications 80:35–44

[32] McGinnis WD, Siu C, Andre S, et al (2018) Category encoders: a scikit-learn-contrib package of transformers for encoding categorical data. Journal of Open Source Software 3(21):501

[33] Murtagh F, Contreras P (2017) Algorithms for hierarchical clustering: an overview, ii. WIREs Data Mining and Knowledge Discovery 7(6):e1219. https://doi.org/https://doi.org/10.1002/widm.1219

[34] Neamatollahi P, Abrishami S, Naghibzadeh M, et al (2017) Hierarchical clustering-task scheduling policy in cluster-based wireless sensor networks. IEEE Transactions on Industrial Informatics 14(5):1876–1886

[35] Orzechowski P, Proficz J, Krawczyk H, et al (2017) Categorization of cloud workload types with clustering. In: Proceedings of the International Conference on Signal, Networks, Computing, and Systems: ICSNCS 2016, Volume 1, Springer, pp 303–313

[36] Pedregosa F, Varoquaux G, Gramfort A, et al (2011) Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12:2825–2830

[37] Rousseeuw PJ (1987) Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20:53–65. https://doi.org/10.1016/0377-0427(87)90125-7

[38] Satopaa V, Albrecht J, Irwin D, et al (2011) Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In: 2011 31st International Conference on Distributed Computing Systems Workshops, pp 166–171, https://doi.org/10.1109/ICDCSW.2011.20

[39] Sculley D (2010) Web-scale k-means clustering. In: Proceedings of the 19th international conference on World wide web, pp 1177–1178

[40] Shahidinejad A, Ghobaei-Arani M, Masdari M (2021) Resource provisioning using workload clustering in cloud computing environment: a hybrid approach. Cluster Computing 24(1):319–342

[41] Strehl A, Ghosh J (2002) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. Journal of machine learning research 3(Dec):583–617

[42] Tabak J (2014) Geometry: The Language of Space and Form. Facts on File math library, Infobase Publishing, URL https://books.google.ca/books?id=r0HuPiexnYwC

[43] Tareq M, Sundararajan EA, Harwood A, et al (2021) A systematic review of density grid-based clustering for data streams. Ieee Access 10:579–596

[44] Thakur N, Singh A, Sangal A (2022) Cloud services selection: A systematic review and future research directions. Computer Science Review 46:100514

[45] Topchy A, Jain AK, Punch W (2004) A mixture model for clustering ensembles. In: Proceedings of the 2004 SIAM international conference on data mining, SIAM, pp 379–390

[46] Vega-Pons S, Ruiz-Shulcloper J (2011) A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and

Artificial Intelligence 25(03):337–372

[47] Xia Q, Lan Y, Zhao L, et al (2014) Energy-saving analysis of cloud workload based on k-means clustering. In: 2014 IEEE Computers, Communications and IT Applications Conference, IEEE, pp 305–309

[48] Yang Q, Zhou Y, Yu Y, et al (2015) Multi-step-ahead host load prediction using autoencoder and echo state networks in cloud computing. The Journal of Supercomputing 71:3037–3053

[49] Yousif SA, Al-Dulaimy A (2017) Clustering cloud workload traces to improve the performance of cloud data centers. In: Proceedings of the World Congress on Engineering, pp 7–10

[50] Zaman K, Hussain A, Imran M, et al (2022) Cost-effective data replication mechanism modelling for cloud storage. International Journal of Grid and Utility Computing 13(6):652–669

[51] Zhang Q, Yang LT, Yan Z, et al (2018) An efficient deep learning model to predict cloud workload for industry informatics. IEEE transactions on industrial informatics 14(7):3170–3178

[52] Zhou B, Lu B, Saeidlou S (2022) A hybrid clustering method based on the several diverse basic clustering and meta-clustering aggregation technique. Cybernetics and Systems pp 1–27

[53] Zhu L, Huang K, Fu K, et al (2023) A priority-aware scheduling framework for heterogeneous workloads in container-based cloud. Applied Intelligence 53(12):15222–15245