People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University of 8 May 1945—Guelma

Faculty of Mathematics, Computer Science & Science of Matter

Department of Computer Science

Final Year Thesis

Field: Computer Science

Option: Information and Communication Science and Technology

Two-Phase Dimensionality Reduction Using Representative Selection and UMAP Training

---

Jury members:

– **Supervisor:** Dr. CHOHRA Chemseddine

– **President:** Dr AGGOUNE Aïcha

– **Examiner:** Dr GUERROUI Nadia

Presented by:

AMIRI Lina

June 2025

Acknowledgement

Alhamdulillah — All praise is due to Allah, the Most Merciful, for granting me the strength, clarity, and perseverance to complete this work. It is by His will and grace that I have reached this important milestone.

I would like to express my deepest gratitude to my supervisor, **Dr. Chohra Chemseddine**, for his valuable expertise, unwavering support, and sincere trust throughout the course of this research. His guidance played a crucial role in shaping this thesis, and I am truly honored to have worked under his supervision.

I would like to express my deep gratitude to **Dr. Aggoune Aïcha** & **Dr. Guerroui Nadia** for the honor they did me by accepting the responsibility of examining this work and participating in the defense jury.

My heartfelt thanks go to my dear parents, **Benkirat Souhila** & **Amiri Abdelhamid**, whose unconditional love, countless sacrifices, and constant prayers have been the foundation of my success. I am forever grateful for everything they have done for me.

I also wish to thank my beloved sisters, **Nawel** & **Nesrine**, for their warmth and moral support during this journey. Your comforting presence and encouragement helped me stay strong through the most challenging moments.

Thank you to everyone who has played a role in my journey, and I look forward to the opportunities that lie ahead.

Abstract

High-dimensional data is increasingly prevalent across diverse domains such as bioinformatics, medical imaging, and natural language processing, posing significant challenges due to the curse of dimensionality and computational complexity. This thesis proposes a novel two-phase dimensionality reduction framework that combines representative selection through clustering with Uniform Manifold Approximation and Projection (UMAP) training. In the first phase, representative samples are selected using clustering algorithms such as Mini-Batch KMeans and BIRCH to reduce data size while preserving its structure. In the second phase, UMAP is trained on these representatives to learn a low-dimensional embedding, which is then used to transform the entire dataset efficiently. Experimental results on the IoTID20 dataset; a high-dimensional dataset, demonstrate that the proposed method significantly reduces computational time and memory usage compared to standard UMAP, while maintaining comparable embedding quality and classification performance. This hybrid approach offers a scalable and effective solution for dimensionality reduction in large-scale high-dimensional data analysis.

Keywords: Dimensionality reduction, high-dimensional data, representative selection, clustering, UMAP, manifold learning, computational efficiency.

Contents

List of Figures

List of Tables

General Introduction

The proliferation of high-dimensional data in domains such as IoT, bioinformatics, and image processing has introduced significant challenges for data analysis. The curse of dimensionality increases computational complexity and degrades the performance of machine learning models, while resource constraints in environments like IoT devices exacerbate these issues. For instance, applying nonlinear dimensionality reduction techniques like UMAP to large datasets such as IoTID20 (625,783 records, 72 features) demands substantial memory and processing power, often infeasible for real-time applications. Traditional methods like PCA fail to capture nonlinear relationships, while advanced techniques like t-SNE and UMAP suffer from quadratic complexity, necessitating scalable alternatives.

The need for efficient, scalable dimensionality reduction is critical across multiple domains. In IoT security, real-time intrusion detection on resource-constrained devices requires rapid processing of high-dimensional network traffic data. Similarly, in bioinformatics, analyzing single-cell RNA sequencing data with thousands of features demands methods that balance computational efficiency with structural preservation. Medical imaging and financial analytics further underscore the need for scalable solutions to handle large, complex datasets. While existing hybrid approaches combine clustering with dimensionality reduction but often lack adaptability for resource-limited settings. This motivates the development of a novel framework that reduces computational overhead while preserving data integrity for diverse applications.

This thesis proposes a two-phase dimensionality reduction framework to address these challenges:

1. **Representative Selection**: Identify a compact subset of data points using clustering algorithms (Mini-Batch KMeans and BIRCH) to capture structural diversity while reducing dataset volume.

2. **Optimized UMAP Training**: Train UMAP on the representative subset to create high-quality, low-dimensional embeddings applicable to the full dataset.

The framework aims to achieve computational efficiency, preserve local and global data structures, and ensure scalability for large datasets in resource-constrained environments, such as IoT intrusion detection systems.

This research makes three primary contributions:

1. **Scalable Framework**: A novel two-phase approach that synergizes clustering-based representative selection with UMAP, significantly reducing training time and memory usage compared to full-data UMAP training.

2. **Practical Applicability**: Demonstrated effectiveness on the IoTID20 dataset for real-time intrusion detection, with potential applications in bioinformatics (e.g., genomics) and image processing.

3. **Adaptive Methodology**: Guidelines for selecting clustering algorithms (e.g., Mini-Batch KMeans for local patterns, BIRCH for global structures) to tailor the framework to task-specific needs.

Unlike existing methods that rely on full-data processing or heuristic sampling, this framework offers a balanced, scalable solution for high-dimensional data analysis.

The thesis is organized as follows:

- **Chapter 1: High-dimensional data and its challenges** discusses the concept of high-dimensional data, its different applications, and its challenges.

- **Chapter 2: Related Work** reviews linear and nonlinear dimensionality reduction techniques, representative selection methods, and their applications.

- **Chapter 3: Methodology** details the proposed two-phase framework, including data preprocessing, representative selection, and UMAP training.

- **Chapter 4: Results and Discussion** presents experimental results on the IoTID20 dataset, evaluating computational efficiency, embedding quality, and downstream task performance.

Chapter 1

High-Dimensional Data and Challenges

1.1 Introduction

The rapid growth of data-driven technologies has led to the generation of increasingly complex datasets across various scientific, industrial, and societal domains. These datasets often involve hundreds, thousands, or even millions of variables, giving rise to what is known as **high-dimensional data**. While this richness in information opens new opportunities for data analysis and machine learning, it also introduces a host of challenges that hinder model performance, increase computational costs, and reduce interpretability.

This chapter provides a comprehensive overview of high-dimensional data and the motivations behind dimensionality reduction. We begin by defining high-dimensional data and illustrating its presence in real-world applications. We then explore the key characteristics that make such data difficult to work with, followed by an in-depth discussion of the **curse of dimensionality**, which encompasses many of the problems associated with high dimensions. Finally, we introduce the primary goals of dimensionality reduction, and we conclude with a high-level comparison of linear versus nonlinear techniques as a foundation for deeper discussions in later chapters.

1.2 Definition and Applications of High-Dimensional Data

High-dimensional data typically refers to datasets characterized by a large number of features (variables or attributes), often comparable to or exceeding the number of observations, which poses unique challenges for statistical analysis and model selection [2, 3]. This concept is especially relevant in fields such as genomics, where tens of thousands of gene expression measurements may be recorded for relatively few samples. The complexity of such data requires specialized methods for variable selection, dimensionality reduction, and prediction [4].

1.2.1 Applications of High-Dimensional Data

High-dimensional datasets are prevalent across many domains, including but not limited to:

- **Internet of Things (IoT):** Sensor networks produce large volumes of data with many features collected continuously.
- **Bioinformatics and Genomics:** Technologies such as DNA microarrays and RNA sequencing provide gene expression measurements for tens of thousands of genes across limited samples [2].
- **Medical Imaging:** High-resolution 3D scans from MRI, CT, and PET generate millions of voxel-based features [3].
- **Computer Vision:** High-resolution images are represented as high-dimensional data with each pixel as a feature.
- **Finance and Economics:** Real-time tracking of numerous financial indicators and macroeconomic variables constitutes high-dimensional datasets.
- **Natural Language Processing (NLP):** Text data represented by techniques such as Term Frequency-Inverse Document Frequency (TF-IDF) results in very high-dimensional and sparse feature spaces [5].

These applications illustrate how the demand for advanced data collection and analytics naturally leads to high-dimensional feature spaces, where traditional data processing and learning methods often fail.

1.3 Key Characteristics of High-Dimensional Data

1.3.1 Sparsity and the Empty Space Phenomenon

As the number of dimensions increases, data points become increasingly sparse within the feature space. This is known as the **empty space phenomenon**. In high-dimensional spaces, most data points lie near the boundaries rather than the center of the space. As a result, conventional algorithms that rely on proximity, such as k-nearest neighbors or k-means clustering, become less effective [6].

1.3.2 Distance Concentration

Another surprising property of high-dimensional spaces is that the contrast between the nearest and farthest neighbors becomes negligible. This is referred to as **distance concentration**. More formally, in high-dimensional settings, the ratio between the minimum

and maximum distances among data points approaches 1. Thus, distance metrics lose their discriminatory power [7].

1.3.3 Feature Redundancy and Correlation

High-dimensional data often contains highly correlated or even linearly dependent features. This redundancy leads to overparameterized models and increased variance. It also affects model interpretability, as it becomes unclear which features are truly important. Feature selection and extraction techniques are often required to remove irrelevant or redundant variables [8].

1.3.4 Small Sample Size Relative to Dimensionality

A common scenario is the "**small n, large p**" problem, where the number of features (p) significantly exceeds the number of samples (n). This imbalance limits the ability of learning algorithms to generalize, often resulting in models that memorize the training data (overfitting) rather than learning general patterns [9]. For example, in genomic datasets, it is common to have 20,000 features (genes) but only 100 samples (patients).

1.4 Challenges in High-Dimensional Spaces

1.4.1 The Curse of Dimensionality

The phrase *curse of dimensionality*, coined by Bellman in the 1960s [10], refers to various adverse effects of high dimensionality on data analysis. These effects include:

- **Exponential Volume Growth:** In a p-dimensional hypercube, the number of grid points needed to densely sample the space increases exponentially with p. For instance, covering a space with just 10 points per dimension leads to 10^p total points.
- **Data Sparsity and Isolation:** As dimensionality increases, data points become more isolated, making statistical estimation and pattern recognition extremely difficult.
- **Overfitting and Poor Generalization:** As the number of dimensions increases, the number of parameters needed to model the data also grows. Without sufficient training data, models become more prone to overfitting.
- **Loss of Meaningful Neighborhoods:** In high-dimensional settings, all data points tend to appear equally distant, making it hard to define "similarity" or "closeness" reliably. For example, consider points uniformly distributed in a unit hypercube. In 2 dimensions, the average distance between two random points is

about 0.52. In 10 dimensions, it rises to over 1.27. As dimensionality increases, the relative difference between the nearest and farthest neighbor distances approaches zero. This phenomenon, known as *distance concentration*, has been studied extensively [11, 12] and severely limits the effectiveness of distance-based methods such as K-NN or K-Means clustering; as mentioned earlier in section 3.2.

These effects combine to make high-dimensional data both computationally demanding and statistically unreliable for many traditional algorithms.

1.4.2 High Computational Complexity

Many machine learning algorithms scale poorly with the number of dimensions. The computational costs in terms of both time and memory often grow linearly or quadratically with the number of features. For example:

- PCA involves an eigen-decomposition of the covariance matrix, which has a time complexity of $\mathcal{O}(p^3)$.
- t-SNE has a time complexity of $\mathcal{O}(n^2)$, making it infeasible for large datasets.
- UMAP scales better but still suffers from memory usage issues with large high-dimensional inputs.

For large datasets, especially those collected in real-time (e.g., IoT), these computational bottlenecks become a major concern [13].

1.4.3 Noise and Irrelevant Features

High-dimensional datasets often contain a large number of irrelevant or noisy features. These features may be the result of sensor drift, environmental noise, or redundant variables. If not handled properly, they can obscure meaningful patterns, mislead learning algorithms, and decrease both accuracy and interpretability [9]. Dimensionality reduction helps mitigate this issue by isolating the most informative aspects of the data.

1.5 Dimensionality Reduction

To effectively manage the challenges posed by high-dimensional data-such as increased computational complexity, risk of overfitting, and reduced interpretability-various strategies are employed, with dimensionality reduction being a primary and powerful approach. Dimensionality reduction involves transforming data from a high-dimensional space to a lower-dimensional space while preserving meaningful information. This process mitigates overfitting, reduces computational cost, and facilitates visualization and interpretation of complex data [14].

1.5.1 Goals of Dimensionality Reduction

To address these issues, **dimensionality reduction** techniques aim to project high-dimensional data into a lower-dimensional space while preserving as much relevant structure and information as possible. The main goals are:

- **Improved Visualization:** Mapping to 2D or 3D allows for graphical representations, helping identify clusters, outliers, and trends.
- **Computational Efficiency:** Reducing the number of features decreases training time and memory consumption.
- **Noise and Redundancy Elimination:** Dimensionality reduction filters out irrelevant or redundant features, increasing model robustness.
- **Improved Generalization:** Lower dimensional representations help reduce overfitting and improve model accuracy.
- **Better Interpretability:** Fewer dimensions make it easier to understand and explain model decisions.

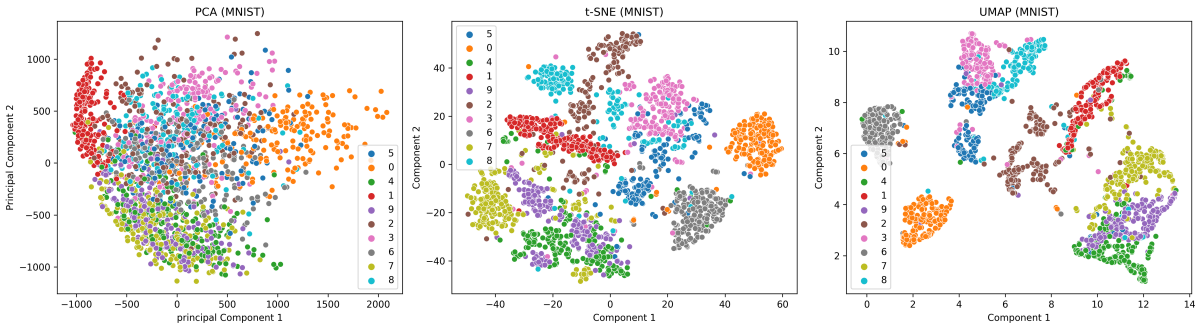


Figure 1.1: Two-dimensional embeddings of the MNIST dataset using (a) PCA, (b) t-SNE, and (c) UMAP. [1].

As shown in Figure 1.1, PCA (a) struggles to separate digit classes in two dimensions, whereas t-SNE (b) and UMAP (c) produce distinct clusters. UMAP, in particular, maintains the overall manifold shape while keeping digit groups compact, exemplifying why nonlinear techniques are essential for high-dimensional data.

1.6 Conclusion

In this chapter, we defined high-dimensional data and illustrated how it arises across diverse real-world domains. We examined its key characteristics—such as sparsity, distance concentration, and high feature correlation—and discussed the computational and statistical challenges it poses, including the curse of dimensionality, algorithmic inefficiency,

and susceptibility to noise and overfitting. To address these challenges, we introduced dimensionality reduction as a fundamental strategy to improve model performance, visualization, and interpretability. We also highlighted the importance of preserving the intrinsic structure of data. In the next chapter, we will dive into the literature: critically reviewing classical and state-of-the-art reduction techniques, comparing their strengths and weaknesses, and identifying the gaps that motivate our novel two-phase approach.

Chapter 2

Related Work

2.1 Introduction

In recent years, the explosion of data across various fields has posed significant challenges in terms of data storage, processing, and analysis. High-dimensional data, while rich in information, often suffers from issues such as increased computational complexity, overfitting, and the curse of dimensionality. The need for robust techniques to extract meaningful information while minimizing redundancy has become increasingly critical.

This chapter presents a comprehensive review of the existing literature on dimensionality reduction, with a particular focus on feature selection and feature extraction methods. We begin by examining traditional approaches such as Principal Component Analysis (PCA), then explore more recent non-linear techniques like Uniform Manifold Approximation and Projection (UMAP). The aim is to highlight the strengths and limitations of these techniques and to provide context for the methodology adopted in this thesis.

2.2 Dimensionality Reduction

Dimensionality reduction is the process of transforming high-dimensional data into a lower-dimensional representation while preserving meaningful properties of the original data. This process helps to simplify data analysis, reduce computational cost, and mitigate issues such as overfitting and the curse of dimensionality. Typically, dimensionality reduction methods aim to learn relationships among features and create a sparse latent structure that eliminates redundant or irrelevant features, facilitating more efficient data processing and interpretation [15, 16].

The following subsections provide an overview of these approaches, their methods, advantages, and limitations.

2.2.1 Feature Selection Methods

Feature selection involves selecting a subset of relevant features from the original dataset without altering them. This approach reduces dimensionality by excluding features that are irrelevant, redundant, or noisy, thereby improving model performance and interpretability [17].

Feature selection methods can be broadly categorized based on two perspectives: the nature of their strategy (filter, wrapper, or embedded) and their supervision level (supervised or unsupervised). In the context of labeled data, such as the IoTID20 dataset used in this study, supervised feature selection methods leverage the output labels to evaluate feature relevance, whereas unsupervised methods operate solely based on the intrinsic properties of the features, often using clustering or statistical measures.

Strategy-Based Categorization. Feature selection methods are commonly classified into the following categories:

- **Filter methods:** These methods rank features based on statistical measures such as correlation with the target variable, mutual information, or ANOVA F-score. Features scoring below a predefined threshold are discarded. Filter methods are independent of any learning algorithm, making them computationally efficient and scalable to high-dimensional data. However, they do not capture feature interactions and may select suboptimal subsets for specific classifiers [18].

- **Wrapper methods:** Wrapper approaches evaluate feature subsets by training and testing a specific classifier, selecting the subset that yields the best predictive performance. This allows them to account for feature interactions, but their exhaustive nature makes them computationally expensive and less scalable. Moreover, they are tightly coupled with the chosen classifier and prone to overfitting, especially in high-dimensional datasets [19].

- **Embedded methods:** These methods perform feature selection during model training. For instance, regularization techniques like LASSO penalize less relevant features by shrinking their coefficients toward zero. Decision tree-based models also provide feature importance scores as a byproduct of their structure. Embedded methods strike a balance between filter and wrapper techniques, offering both computational efficiency and task-specific relevance [20].

Supervision-Based Categorization. Feature selection techniques may also be classified as:

- **Supervised methods:** These use class labels to guide feature relevance estimation. Techniques like mutual information, information gain, chi-squared tests, and

recursive feature elimination (RFE) are typical examples. They are especially useful when the objective is classification or regression [21].

- **Unsupervised methods:** These operate without class labels and often rely on statistical metrics such as variance thresholding, Laplacian scores, or clustering consistency. They are useful when labels are unavailable or when preparing data for unsupervised learning tasks like clustering or dimensionality reduction [22].

Despite their benefits, feature selection methods face limitations in terms of scalability when dealing with extremely high-dimensional data, such as genomic or IoT traffic datasets. Additionally, they may struggle to preserve the intrinsic data structure or nonlinear relationships, particularly in unsupervised scenarios [23].

2.2.2 Feature Extraction Methods

Feature extraction methods transform the original high-dimensional data into a new, lower-dimensional space by creating new features that capture the essential information. Unlike feature selection, which retains original features, feature extraction combines or projects features to reduce dimensionality while aiming to preserve data variance or structure [24].

Feature extraction techniques can be broadly divided into:

- **Linear methods:** These methods assume that the data lie on or near a linear subspace of the high-dimensional space. They seek linear transformations that maximize variance or class separability. Examples include Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Singular Value Decomposition (SVD), and Independent Component Analysis (ICA). Linear methods are computationally efficient and interpretable but may fail to capture complex nonlinear structures in data [15].

- **Nonlinear methods:** To address limitations of linear methods, nonlinear feature extraction techniques have been developed to capture complex intrinsic structures of data that lie on nonlinear manifolds. These include Kernel PCA (KPCA), Multidimensional Scaling (MDS), Isomap, Locally Linear Embedding (LLE), Self-Organizing Maps (SOM), t-Distributed Stochastic Neighbor Embedding (t-SNE), and Uniform Manifold Approximation and Projection (UMAP). Nonlinear methods often provide better representations for visualization and clustering in complex datasets but can be computationally more demanding and less interpretable [16, 25].

The subsequent sections provide detailed descriptions of representative linear and nonlinear feature extraction methods.

2.2.3 Linear Dimensionality Reduction

2.2.3.1 Principal Component Analysis (PCA)

Principal Component Analysis is a widely applied statistical method used to reduce the dimensionality of datasets by projecting them onto a new set of orthogonal axes called principal components. These components are ordered in terms of the variance they explain, with the first principal component capturing the maximum possible variance, the second capturing the next highest, and so on. Each principal component is a linear combination of the original variables and is uncorrelated with the others [26, 27]. The method begins by centering the data and optionally scaling it to unit variance if the variables are measured on different scales. Then, PCA performs either an eigenvalue decomposition (EVD) of the covariance matrix or a singular value decomposition (SVD) of the centered data matrix. The eigenvectors (or right singular vectors) form the principal component directions, while the eigenvalues (or squared singular values) represent the amount of variance explained [27]. This low-rank approximation is optimal in a least-squares sense: PCA finds a subspace that minimizes the reconstruction error from the original high-dimensional space to the lower-dimensional one [27]. The new coordinates (scores) enable compact representations of the data, which are useful for visualization, compression, and feature extraction. PCA has been used in various fields including chemometrics, genomics, psychology, and environmental science due to its simplicity, robustness, and computational efficiency [26].

Strengths of PCA

- **Efficient dimensionality reduction**: PCA compresses data by capturing most of its variance in just a few components, often reducing hundreds of variables to only two or three [27].
- **Computationally efficient**: The mathematical operations involved—especially SVD—are fast, stable, and scalable to large datasets. SVD is preferred in practice due to its numerical advantages [27].
- **Noise reduction**: PCA effectively filters out components associated with low variance, which often correspond to noise in the dataset [26].
- **Facilitates data visualization**: By projecting high-dimensional data onto a 2D or 3D space using the first few principal components, PCA helps reveal clusters, patterns, and outliers [26, 27].
- **Model-free and unsupervised**: PCA does not require labeled data or prior assumptions about the underlying data distribution, making it widely applicable across domains [26].

- **Interpretability through explained variance**: PCA provides a clear measure of how much variance is retained in each component, helping users determine how many components to keep (e.g., via scree plots or explained variance thresholds) [27].

Limitations of PCA

- **Sensitivity to Outliers:** PCA is sensitive to outliers and gross errors in the dataset, which can distort the principal components and lead to misleading results [15].
- **Dependence on Scaling and Units:** PCA results depend heavily on the scale and units of the variables. Without proper standardization, variables with larger variances dominate the principal components, affecting interpretability and results [28, 29].
- **Linearity Assumption:** PCA assumes linear relationships among variables and cannot capture nonlinear patterns in the data, limiting its effectiveness for complex datasets with nonlinear structures [29, 30].
- **Interpretability of Components:** Principal components are linear combinations of original variables, which often lack straightforward interpretability, making it difficult to relate components back to meaningful real-world features. To address this, sparse PCA methods have been proposed, which impose sparsity on the loadings to enhance interpretability by selecting only a subset of variables per component [31]. As Zou et al. (2006) note, "Sparse principal components are easier to interpret because each component depends on only a small number of variables" [31, p. 265].
- **Requirement of Large Sample Size for Robustness:** Peres-Neto and Jackson (2016) emphasize that "small sample sizes relative to the number of variables can lead to unstable ordination results, and a minimum ratio of observations to variables is necessary to ensure robustness" [32, p. 1247]. Similarly, Johnstone and Lu (2009) demonstrate that classical PCA can be inconsistent in high-dimensional, low-sample-size settings, motivating robust and sparse PCA approaches [33]. .

2.2.3.2 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction and classification technique that aims to find a linear combination of features that best separates two or more classes [21]. Unlike PCA, which is unsupervised and focuses on maximizing variance, LDA explicitly considers class labels to maximize the ratio of between-class variance to within-class variance, thereby enhancing class separability [34]. LDA projects

high-dimensional data onto a lower-dimensional space (with dimensionality at most one less than the number of classes) while preserving discriminative information [21]. It is based on a generative model framework and uses Bayes' theorem to classify new data points [21]. Originally developed by Fisher in the 1930s, LDA has been widely applied in various domains such as finance, healthcare, marketing, and image recognition [21, 35].

Strengths of LDA

- LDA maximizes class separability by finding linear combinations of features that best discriminate between classes, improving classification accuracy [21].
- It performs dimensionality reduction while preserving discriminative information, reducing computational complexity [34].
- LDA provides interpretable linear combinations that highlight the most relevant features for classification [34].
- It is robust to multicollinearity among predictor variables, which can degrade other classifiers [34].
- LDA has been successfully applied in diverse domains such as image recognition, text classification, and medical diagnosis, demonstrating its versatility [35].

Limitations of LDA

- **Assumption of Equal Covariance**: LDA assumes that all classes share the same covariance matrix. If this assumption is violated, the performance of LDA can degrade [36].
- **Sensitivity to Outliers**: Since LDA relies on mean and covariance estimates, it can be sensitive to outliers, which may skew these estimates and affect the resulting projections [36].
- **Singularity Issues**: In cases where the number of features exceeds the number of samples, the within-class scatter matrix S_W may become singular, making it non-invertible. This issue is common in high-dimensional settings like image recognition [37].
- **Linear Boundaries**: LDA creates linear decision boundaries, which may not be sufficient for complex datasets where classes are not linearly separable [36].
- **Limited to Gaussian Distributions**: The optimality of LDA is contingent on the assumption that class distributions are Gaussian. Deviations from this assumption can lead to suboptimal performance [36].

2.2.3.3 Advancements: Two-Dimensional LDA (2DLDA)

To address some of the limitations of classical LDA, especially in high-dimensional contexts, Two-Dimensional LDA (2DLDA) has been proposed. Unlike traditional LDA, which requires flattening matrix data (like images) into vectors, 2DLDA operates directly on matrix data. This approach preserves the spatial structure of the data and reduces computational complexity [37].

2DLDA mitigates the singularity problem by avoiding the computation of the inverse of the within-class scatter matrix. Instead, it formulates the problem in a way that does not require matrix inversion, making it more robust in scenarios where the number of features is large compared to the number of samples [37].

2.2.3.4 Comparison with PCA

While both LDA and PCA are linear transformation techniques used for dimensionality reduction, they serve different purposes:

- **Objective**: PCA seeks directions that maximize variance without considering class labels, making it unsupervised. LDA, on the other hand, seeks directions that maximize class separability, making it supervised [36].
- **Assumptions**: PCA does not make assumptions about the underlying data distribution, whereas LDA assumes Gaussian distributions with equal covariances [36].
- **Performance in Classification**: LDA generally outperforms PCA in classification tasks due to its consideration of class labels during the dimensionality reduction process [36].
- **Sensitivity to Data Structure**: PCA may capture directions of maximum variance that are not relevant for class discrimination, while LDA focuses specifically on directions that aid in distinguishing between classes [36].

Linear Discriminant Analysis is a powerful tool for supervised dimensionality reduction and classification, especially when its assumptions are met. Its extensions, like 2DLDA, have expanded its applicability to high-dimensional data scenarios. However, practitioners must be mindful of its assumptions and potential limitations, particularly regarding data distribution and linear separability.

2.2.3.5 Factor Analysis (FA)

Factor Analysis (FA) is a multivariate statistical technique designed to analyze correlations among many observed variables and to explore underlying latent factors that account for these correlations [38, 39]. By reducing a large set of variables to a smaller number of

factors, FA helps researchers understand the structure of complex data and identify the dimensions that explain relationships between variables [39, 40].

There are two main types of factor analysis:

1. **Exploratory Factor Analysis (EFA):** EFA is used in the early stages of research to explore the underlying structure of a dataset and identify the number and nature of latent factors without imposing any preconceived model [41]. The typical steps in EFA involve:

   - *Data Preparation:* Ensuring that the data is suitable for factor analysis, including checking sample size, correlations among variables, and missing data.
   - *Factor Extraction:* Determining the number of factors to retain using criteria such as eigenvalues (Kaiser's rule), scree plots, or parallel analysis.
   - *Factor Rotation:* Applying orthogonal (e.g., Varimax) or oblique (e.g., Promax) rotation techniques to improve the interpretability of the factors by simplifying the factor loadings.
   - *Interpretation:* Assigning meaning to the factors based on the pattern of variable loadings.

   EFA is particularly useful when there is no clear hypothesis about the number or nature of the underlying factors, making it a valuable tool for theory development and exploratory research [41].

2. **Confirmatory Factor Analysis (CFA):** CFA is used to test specific hypotheses about the factor structure of a dataset based on prior theory or research [42]. In CFA, the researcher specifies the number of factors, the variables that load on each factor, and any relationships between the factors. The model is then tested to determine how well it fits the observed data. Key steps include:

   - *Model Specification:* Formulating a theoretical model that specifies the number of factors, the relationships between factors and measured variables, and any covariances among factors or error terms.
   - *Model Identification:* Ensuring that the model is identified, meaning that there is a unique solution for the model parameters. This typically requires setting constraints on the model, such as fixing the variance of each factor to 1 or setting one factor loading per factor to a non-zero value.
   - *Model Estimation:* Estimating the model parameters using techniques such as maximum likelihood estimation.
   - *Model Evaluation:* Assessing the fit of the model to the data using various fit indices, such as the chi-square statistic, root mean square error of approximation (RMSEA), comparative fit index (CFI), and Tucker-Lewis index (TLI).

- *Model Modification:* If the initial model does not fit the data well, it may be modified by adding or removing paths between variables and factors or allowing factors to correlate. However, any modifications should be theoretically justified and not solely based on statistical criteria.

CFA is particularly useful for validating measurement instruments, testing theoretical models, and comparing different factor structures [42].

Strengths of Factor Analysis

- **Dimensionality Reduction:** FA simplifies complex datasets by reducing the number of observed variables to a smaller set of interpretable factors [39, 38].
- **Uncovering Latent Structure:** It reveals underlying constructs that explain the correlations among observed variables, supporting theory development and construct validity [39].
- **Instrument Development:** FA is widely used in developing and refining measurement instruments, ensuring that items group together as intended and measure the same construct [39, 40].
- **Data Summarization:** It helps summarize and interpret large datasets, making them more manageable for further analysis [38].

Limitations of Factor Analysis:

- **Subjectivity in Decision-Making:** Decisions about the number of factors to retain, extraction methods, and rotation techniques can be subjective and influence results [43].
- **Sample Size Requirements:** Reliable results typically require large sample sizes; small samples may yield unstable or non-generalizable solutions [43].
- **Assumptions and Data Quality:** FA assumes linear relationships and sufficient correlations among variables, and is sensitive to outliers and missing data [43].
- **Interpretation Challenges:** Interpreting and naming factors can be difficult, especially when variables load on multiple factors or when factor structures are ambiguous [39].
- **No Causality:** FA identifies associations but does not establish causal relationships between variables and factors [39].

2.2.3.6 Multidimensional Scaling (MDS)

Multidimensional Scaling (MDS) is a set of techniques used to visualize the structure of data by representing it as a geometric configuration in a low-dimensional space [44]. MDS takes as input a matrix of pairwise dissimilarities or distances between objects and aims to find a spatial arrangement of points such that the distances between points in the low-dimensional space approximate the original dissimilarities. This approximation is achieved by minimizing a loss function called "stress," which quantifies the mismatch between the original dissimilarities and the distances in the low-dimensional space [44, 45]. There are two main variants of MDS:

1. **Metric MDS**, which assumes that the dissimilarities are measured on an interval or ratio scale and attempts to preserve these distances as accurately as possible [44]. Metric MDS is typically used when the dissimilarities are derived from well-defined metrics, such as Euclidean distances.

2. **Non-Metric MDS**, which assumes only that the dissimilarities are ordinal and focuses on preserving the rank order of the dissimilarities rather than their exact values [46]. Non-metric MDS is useful when the dissimilarities are subjective or based on qualitative judgments.

Unlike Principal Component Analysis (PCA), which seeks linear projections that maximize variance, MDS explicitly aims to preserve pairwise distances or dissimilarities, making it especially useful when the original data are non-Euclidean or available only as similarity measures [47, 48]. MDS is widely used in various fields, including psychology for perceptual mapping, marketing for consumer preference analysis, and bioinformatics for visualizing genetic or protein similarities [44].

Strengths of Multidimensional Scaling

- **Flexibility with Data Types and Scales:** MDS can handle both metric and non-metric dissimilarity data, increasing its applicability across diverse datasets [49, 50].

- **Capability to Model Nonlinear Relationships:** It is capable of modeling nonlinear relationships, providing more accurate representations of complex similarity structures compared to linear methods [50].

- **Intuitive Visual Representation of Similarities:** MDS produces intuitive spatial maps that facilitate pattern recognition and exploratory data analysis [50, 51].

Limitations of Multidimensional Scaling

- **Computational Intensity on Large Datasets:** The iterative optimization process in MDS is computationally intensive for large datasets, limiting scalability [49, 50].
- **Sensitivity to Noise and Outliers:** MDS is sensitive to noise and outliers, which can distort the spatial configuration and complicate interpretation [50].
- **Subjectivity and Difficulty in Interpretation:** Interpretation of MDS dimensions is subjective, and high stress values may indicate poor goodness-of-fit [50].

2.2.4 Nonlinear Dimensionality Reduction

2.2.4.1 t-Distributed Stochastic Neighbor Embedding (t-SNE)

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a nonlinear dimensionality reduction technique primarily designed for visualizing high-dimensional data in two or three dimensions [52]. The method converts pairwise similarities between data points into joint probabilities representing the likelihood that points are neighbors. It then seeks a low-dimensional embedding that minimizes the Kullback-Leibler divergence between these joint probability distributions in the original and embedded spaces. This approach emphasizes preserving local neighborhood relationships, making t-SNE particularly effective at revealing clusters and complex, nonlinear structures in data. However, t-SNE does not explicitly preserve global data structure, which can sometimes lead to misleading interpretations of the distances between clusters [53].

Strengths

- *Captures complex nonlinear relationships:* By focusing on local similarities, t-SNE can uncover intricate structures and clusters that linear methods like PCA cannot detect [52].
- *Effective visualization tool:* It excels in producing visually interpretable embeddings that reveal meaningful groupings in data, which is valuable in exploratory data analysis [54].
- *Widely adopted and supported:* Due to its effectiveness, t-SNE has become a standard tool in fields such as bioinformatics, natural language processing, and computer vision [55].

Limitations:

- *High computational cost:* The algorithm has a quadratic time complexity with respect to the number of data points, making it challenging to scale to very large datasets without approximations [55].
- *Parameter sensitivity:* The perplexity parameter, which controls the balance between local and global aspects of the data, requires careful tuning, and different settings can produce substantially different embeddings [56].
- *Non-preservation of global structure:* While local neighborhoods are well preserved, distances between clusters may not reflect true relationships, limiting its use for tasks requiring global topology preservation [53].

2.2.4.2 Isometric Mapping (Isomap)

Isomap is a nonlinear dimensionality reduction technique that extends classical Multidimensional Scaling (MDS) by incorporating geodesic distances computed on a neighborhood graph [57]. By approximating the manifold's intrinsic geometry, Isomap aims to preserve the global structure of data lying on a nonlinear manifold. It constructs a graph connecting each point to its nearest neighbors, computes shortest path distances between all pairs of points (geodesic distances), and then applies MDS to these distances to find a low-dimensional embedding. This approach is particularly effective for unfolding nonlinear manifolds that are globally curved but locally Euclidean. However, Isomap assumes the manifold is convex and can be sensitive to noise, outliers, and the choice of neighborhood size [58].

Strengths:

- *Preserves global manifold structure:* Unlike methods focusing only on local neighborhoods, Isomap captures the overall geometry of the data manifold, enabling meaningful embeddings even for complex nonlinear shapes [57].
- *Interpretable embeddings:* The geodesic distance-based approach often produces embeddings where Euclidean distances correspond well to intrinsic manifold distances [58].

Limitations:

- *Computationally expensive:* The shortest path computations and eigenvalue decompositions scale poorly with dataset size, limiting Isomap's applicability to large datasets [59].

- *Sensitivity to noise and outliers:* Noise can distort geodesic distances, leading to poor embeddings, and outliers can disproportionately affect neighborhood graphs [60].
- *Parameter dependence:* The choice of neighborhood size critically affects results; too small neighborhoods fragment the graph, while too large neighborhoods can oversimplify the manifold [58].

2.2.4.3 Locally Linear Embedding (LLE)

Locally Linear Embedding (LLE) is a nonlinear dimensionality reduction technique that assumes data points lie on or near locally linear patches of a manifold [61]. LLE reconstructs each data point as a linear combination of its nearest neighbors, capturing local geometric properties. It then finds a low-dimensional embedding that preserves these local reconstruction weights. This approach effectively preserves local neighborhood information and can unfold complex manifolds with nonlinear global structure. However, LLE can struggle with non-uniform sampling densities and requires careful tuning of the neighborhood parameter [60].

Strengths:

- *Local neighborhood preservation:* By focusing on local linear reconstructions, LLE maintains the intrinsic geometry of data neighborhoods, which is useful for manifold unfolding [61].
- *Non-parametric and unsupervised:* LLE does not require explicit model assumptions or labels, making it broadly applicable [62].

Limitations:

- *Sensitivity to neighborhood size:* The choice of the number of neighbors influences the quality of the embedding; inappropriate values can lead to disconnected or oversmoothed embeddings [63].
- *Difficulty with non-uniform data:* LLE assumes uniform sampling density; uneven densities or holes in the manifold can degrade performance [60].
- *Computational complexity:* Although more scalable than Isomap, LLE still requires eigenvalue decomposition, which can be costly for very large datasets [61].

2.2.4.4 Autoencoders

Autoencoders are a class of neural network architectures designed to learn efficient, compressed representations of data through an encoder-decoder framework [64]. The encoder maps input data to a lower-dimensional latent space, and the decoder reconstructs the original data from this representation. Variants such as Variational Autoencoders (VAEs) introduce probabilistic modeling of the latent space, enabling generative capabilities [65]. Autoencoders are highly flexible, capable of learning complex nonlinear embeddings, and can be adapted to various data modalities including images, text, and time series.

Strengths:

- *Ability to learn complex nonlinear embeddings:* Autoencoders can model hierarchical and nonlinear relationships in data that traditional linear methods cannot capture [64].
- *Scalability and flexibility:* They can be trained on large datasets using stochastic gradient descent and adapted to different data types and architectures [66].
- *Generative modeling:* Variational Autoencoders enable sampling from the latent space to generate new, realistic data points, useful for data augmentation and simulation [65].

Limitations:

- *Risk of overfitting:* Without proper regularization, autoencoders may memorize training data, reducing generalization to new samples [67].
- *Latent space interpretability:* The learned embeddings are often abstract and lack clear semantic meaning, complicating interpretation [68].
- *Training complexity:* Deep autoencoders require significant computational resources and careful tuning of hyperparameters [66].

2.2.4.5 Uniform Manifold Approximation and Projection (UMAP)

Uniform Manifold Approximation and Projection (UMAP) is a state-of-the-art nonlinear dimensionality reduction technique grounded in concepts from Riemannian geometry and algebraic topology [69]. UMAP builds upon the assumption that data lies on a Riemannian manifold and seeks to preserve both local and global structure when projecting high-dimensional data into a lower-dimensional space.

The core process of UMAP involves two main steps. First, it constructs a weighted k-nearest neighbor graph in the original high-dimensional space, representing local relationships between data points as a fuzzy simplicial complex [69, 35]. The strength

of connection between points is encoded as edge weights, reflecting the probability that two points are connected. Second, UMAP optimizes a low-dimensional embedding by minimizing the cross-entropy between the high-dimensional and low-dimensional fuzzy simplicial sets, thus preserving the structural integrity of the data during transformation [69, 35]. This optimization is typically performed using stochastic gradient descent.

UMAP is highly flexible and can be used for both visualization and general nonlinear dimensionality reduction. Unlike linear methods such as PCA, which only capture linear variance, and t-SNE, which primarily preserves local structure but struggles with scalability and global relationships, UMAP provides a better balance between local and global structure preservation [69, 35, 70]. This makes UMAP especially suitable for large-scale, high-dimensional datasets encountered in fields like genomics, image analysis, natural language processing, and neuroscience [70, 71].

Recent developments have extended UMAP's capabilities. Parametric UMAP replaces the nonparametric optimization with a deep neural network, enabling fast online embeddings for new data points and integration with deep learning models [72]. Supervised UMAP allows label information to guide the embedding, making it useful for semi-supervised and supervised learning tasks [69, 72].

Strengths:

- *Preserves local and global structure:* UMAP maintains meaningful relationships at multiple scales, often outperforming t-SNE in this regard and providing more faithful representations of the data manifold [70, 35].
- *Computational efficiency and scalability:* UMAP is significantly faster than t-SNE and scales well to large datasets, making it practical for modern big data applications [69, 35].
- *Versatility:* UMAP supports unsupervised, supervised, and parametric extensions, enabling its use in a wide range of tasks including visualization, clustering, and as a preprocessing step for machine learning pipelines [72].
- *Generalizability:* Parametric UMAP enables fast embedding of new data points and can be integrated into deep learning architectures for end-to-end learning [72].
- *Domain applicability:* UMAP has been successfully applied in diverse domains such as single-cell genomics, brain imaging, bioinformatics, and finance, demonstrating its robustness and adaptability [70, 71].

Limitations:

- *Hyperparameter sensitivity:* The quality of UMAP embeddings depends on the choice of hyperparameters such as `n_neighbors` (which controls the balance between local and global structure) and `min_dist` (which affects the tightness of clusters). Proper tuning often requires domain knowledge and experimentation [73, 35].
- *Lack of explicit inverse mapping:* Unlike autoencoders, UMAP does not inherently support reconstructing original data from the embedding space, which limits its use in generative modeling and interpretability [74].
- *Non-determinism:* UMAP's optimization is stochastic, so results may vary between runs unless random seeds are fixed [69].
- *Potential for local distortions:* While UMAP balances local and global structure, in some cases, it may still distort fine local relationships or overemphasize certain clusters, especially with suboptimal parameter choices [73].

UMAP's robust mathematical foundation, scalability, and flexibility have made it a leading tool for dimensionality reduction, particularly in applications where both local and global data structure are important. Its ongoing development, including parametric and supervised variants, continues to expand its utility in machine learning and data science [72].

2.3 Representative Selection

Representative selection refers to the process of choosing a subset of data points that accurately reflect the diversity and key characteristics of the entire dataset. Unlike random selection, which picks samples purely by chance, representative selection aims to preserve the underlying structure and important patterns within the data, ensuring that the reduced subset maintains the essential information of the original dataset [75]. This approach is particularly important in high-dimensional data analysis, where maintaining diversity and coverage can significantly impact the performance of subsequent methods such as dimensionality reduction.

2.3.1 Representative Selection Or Random Selection

Random selection involves selecting samples solely based on chance without considering the distribution or features of the dataset. While it is simple and unbiased in theory, random sampling may fail to capture rare but important data patterns, leading to a loss of critical information [76]. In contrast, representative selection employs strategies such as clustering or heuristic algorithms to ensure that the chosen subset proportionally

reflects the variability and structure of the entire dataset [77]. This deliberate selection process enhances the quality of downstream analyses by preserving the dataset's intrinsic characteristics.

- In this study, representative selection is employed as the first phase of the two-phase dimensionality reduction process to ensure that the subsequent application of UMAP operates on a subset that meaningfully represents the original data distribution. This methodological choice is justified by the need to maintain data diversity and structure, which random selection alone cannot guarantee [75, 77]. By doing so, the dimensionality reduction results are more reliable and better capture the underlying data patterns.

2.3.2 Representative Selection Techniques

Representative selection techniques aim to identify a subset of data points that preserve the essential characteristics of the original dataset while reducing computational overhead. These methods are critical for scalability, noise reduction, and interpretability in modern data analysis pipelines. Below, we review clustering-based selection, prototype selection, and core-sets, along with their motivations and limitations.

2.3.2.1 Clustering-Based Methods

Clustering-based repsentative selection techniques are foundational in unsupervised learning and data summarization. These methods partition data into groups (clusters) such that objects within the same cluster are more similar to each other than to those in other clusters [78, 79]. By selecting representative points-such as centroids, medoids, or boundary points-from each cluster, these approaches aim to reduce dataset size while preserving the essential structure and diversity of the original data [80, 81, 79].

A wide range of clustering algorithms exist, including centroid-based (e.g., K-means), hierarchical (e.g., agglomerative clustering), density-based (e.g., DBSCAN), distribution-based (e.g., Gaussian Mixture Models), and graph-based methods (e.g., spectral clustering) [78, 82, 79, 83]. Each has unique strengths and is suited to different data characteristics. For example, K-means is computationally efficient and widely used for its simplicity and rapid convergence, but it assumes spherical clusters of similar size and is sensitive to outliers [84, 79]. Hierarchical clustering, including Ward's, complete, average, and single linkage, provides interpretable dendrograms and can reveal nested data structures, but may be computationally intensive for large datasets [82, 83]. Density-based methods like DBSCAN and OPTICS are robust to noise and can discover clusters of arbitrary shape, making them suitable for complex, real-world datasets [82].

The process of clustering-based representative selection typically involves several steps: feature selection or extraction, distance or similarity measure definition (e.g., Euclidean,

cosine, or kernel-based), clustering, and then selection of representatives from each cluster [78, 83]. Recent innovations include deep embedded clustering, which integrates dimensionality reduction and clustering in a unified framework, and ensemble clustering, which combines multiple clustering results to enhance stability and robustness [82]. Additionally, robust centroid estimation techniques, such as trimmed K-means and M-estimators, have been developed to mitigate the influence of outliers [82].

Clustering-based selection is widely used in applications ranging from image segmentation and mental health research to market basket analysis and social network analysis [78, 79, 79]. Its main motivations include improving scalability for downstream tasks, reducing noise by filtering out atypical data, and enhancing interpretability by summarizing data with a manageable number of representative points [81, 80, 79].

However, these methods also face several challenges. The choice of clustering algorithm and its parameters (such as the number of clusters or neighborhood size) significantly affects the quality and representativeness of the selected samples [83, 82]. Many algorithms are sensitive to initialization and can yield different results on the same data [83, 78]. Furthermore, clustering-based selection may struggle with high-dimensional, noisy, or overlapping data, and there is no universally best algorithm for all scenarios [83, 85]. For these reasons, recent research emphasizes the need for careful algorithm selection, robust validation, and the integration of clustering with dimensionality reduction and ensemble methods to overcome traditional limitations [82, 79, 86].

2.3.2.2 Prototype Selection

Prototype selection is a vital preprocessing step in instance-based learning, particularly for algorithms like k-Nearest Neighbor (k-NN), where the entire training set is used for classification [87, 80, 88]. The primary goal is to reduce the size of the reference set while maintaining or even improving classification accuracy. By retaining only the most informative or representative instances-often those near class boundaries or in dense regions-prototype selection can dramatically decrease computational costs and storage requirements, making k-NN and related classifiers feasible for large-scale datasets [87, 89, 80].

Prototype selection algorithms can be broadly categorized into three families: *condensation*, *edition*, and *hybrid* methods [89, 87]. **Condensation methods**, such as the classic Condensed Nearest Neighbor (CNN) rule, iteratively select a minimal subset of instances that correctly classify the training data. However, CNN is sensitive to the order of data presentation and can be influenced by noise, often resulting in redundant prototypes [89, 80]. **Edition methods**, like Edited Nearest Neighbor (ENN), focus on removing noisy or misclassified instances to clean the dataset, improving robustness to outliers but sometimes discarding useful boundary points [89, 87].

Hybrid methods combine both strategies to balance data reduction and noise removal, often achieving better results than either approach alone [89].

Recent advances address the limitations of early methods. For example, algorithms now incorporate clustering to select both border and interior prototypes, as in the work of Olvera-López et al., who proposed a fast prototype selection method based on clustering that preserves decision boundaries while reducing redundancy [80]. Other innovations include density-based selection, which identifies prototypes in dense regions, and methods that use local feature weighting to prioritize informative instances [89]. Spatial partitioning and mutual nearest-neighbor criteria, as in [90], further accelerate prototype selection and improve scalability for large datasets.

Prototype selection is not only beneficial for computational efficiency but also enhances model generalization by removing redundant and noisy data, reducing the risk of overfitting [87, 88]. However, challenges remain: wrapper-based approaches (which use classifier feedback) are computationally intensive and classifier-dependent, while filter-based approaches may inadvertently discard critical instances, especially in imbalanced datasets [88, 90]. Moreover, many methods are sensitive to the sequence of data presentation and to the presence of outliers or overlapping class distributions [89].

Strengths:

- **Significant reduction in computational cost:** Prototype selection can reduce training and classification time for instance-based classifiers by orders of magnitude, making k-NN feasible for large datasets [90, 87].
- **Noise and redundancy removal:** By discarding noisy and redundant instances, prototype selection improves model generalization and robustness [88, 89].
- **No need for artificial data:** Prototype selection methods work directly with real instances, ensuring interpretability and relevance to the original dataset [87].

Limitations:

- **Classifier dependence and computational cost:** Wrapper-based methods (e.g. CNN) are computationally intensive and must be re-run for each classifier or parameter setting [88, 89].
- **Risk of discarding important instances:** Filter-based methods may remove instances critical for minority classes or for defining complex decision boundaries, leading to reduced accuracy in imbalanced or overlapping datasets [90, 89].
- **Sensitivity to data order and noise:** Many algorithms are sensitive to the order of data presentation and to outliers, which can affect the stability and representativeness of the selected prototypes [89, 80].

2.3.2.3 Core-Sets

Core-sets are compact, weighted subsets of data that approximate the original dataset with theoretical guarantees for specific optimization tasks, such as clustering, regression, and diversity maximization [91, 92, 93]. The fundamental idea is to select a small subset of points (possibly with weights) such that solving the problem on the core-set yields a solution close to that on the full dataset, up to a provable error bound. Core-set construction is especially valuable for large-scale and streaming data, where full-data computations are infeasible [92, 94, 95].

Recent research has focused on fair and diverse data summarization, where core-sets are constructed to ensure proportional representation across partitioned groups (e.g., demographic categories) while maximizing diversity measures such as sum-of-pairwise distances or sum-of-nearest-neighbor distances [91, 92, 96]. These approaches have demonstrated that core-sets can achieve dramatic reductions in data size (e.g., 100x speed-up) with minimal loss of diversity or accuracy, even in real-world applications like summarizing timed messages on large communication platforms [91, 92].Core-sets are also highly effective for streaming and parallel computation, enabling real-time updates and efficient use of computational resources [95, 94].

However, constructing core-sets for high-dimensional data remains computationally challenging, and balancing fairness constraints with diversity objectives often requires careful trade-off tuning [91, 92, 96]. Furthermore, while core-sets provide strong theoretical guarantees for certain objective functions, their construction and effectiveness can be highly problem-dependent, and extensions to more complex or dynamic data settings are ongoing research topics [95, 93].

Strengths:

- **Provable approximation guarantees:** Core-sets provide theoretical bounds on summarization quality for a range of optimization tasks, ensuring near-optimal solutions with much smaller datasets [91, 92, 93].
- **Scalability and streaming capability:** Core-sets enable efficient analysis and real-time updates in streaming and distributed settings, making them suitable for big data applications [95, 94].
- **Fair and diverse summarization:** Modern core-set algorithms can enforce fairness and diversity constraints, ensuring equitable and informative data summaries [91, 92, 96].

Limitations:

- **Computational complexity in high dimensions:** Constructing core-sets for high-dimensional or complex data can be computationally demanding and may require sophisticated algorithms [91, 95].
- **Trade-offs between fairness and diversity:** Enforcing multiple constraints (e.g., fairness and diversity) can conflict, requiring careful parameter tuning and sometimes resulting in suboptimal summaries [91, 92].
- **Problem-specific design:** Core-set construction is often tailored to specific tasks and objectives, limiting generalizability across different problem domains [93, 95].

2.4 Hybrid and Two-Phase Dimensionality Reduction Approaches

2.4.1 Motivation for Hybrid Approaches

While classical dimensionality reduction (DR) methods such as PCA, LDA, and UMAP have achieved remarkable success, each has inherent limitations. Linear methods like PCA are computationally efficient and interpretable, but cannot capture nonlinear structures in complex data [26, 27]. Nonlinear methods such as UMAP or t-SNE are powerful for uncovering manifold structure but can be computationally expensive and may struggle with scalability or interpretability [69, 70]. As a result, no single DR method is universally optimal for balancing structure preservation, computational performance, and downstream task accuracy [97, 98, 99]. To address these challenges, researchers increasingly combine multiple DR techniques in hybrid or two-phase pipelines. These approaches aim to leverage the strengths of each method while mitigating their weaknesses, often resulting in improved scalability, better structure preservation, and enhanced interpretability [97, 98, 99]. Hybrid DR is especially valuable for large-scale or noisy datasets, where a single method may be insufficient.

2.4.2 Representative Selection and Dimensionality Reduction Pipelines

A common hybrid strategy is to first select a subset of representative samples-using clustering, core-set construction, or feature selection-and then apply a more complex DR technique to this reduced set. The rationale is that representative selection can filter out noise, redundancy, and outliers, thus reducing computational cost and improving the effectiveness of subsequent nonlinear embedding [98, 81, 91].

2.4.3 Typical hybrid pipelines include:

- **Clustering-based selection + Nonlinear DR:** For example, K-means clustering is used to select cluster centroids, which are then embedded using t-SNE or UMAP [81, 98]. This approach preserves the global structure while reducing data size.
- **Core-set selection + DR:** Core-set algorithms select a weighted subset that approximates the full dataset for a specific objective (e.g., diversity maximization). Applying UMAP or autoencoders to this core-set yields efficient and representative embeddings [91, 92].
- **Feature selection + Feature extraction:** Filter methods (e.g., information gain, chi-square) select relevant features, followed by PCA or ICA for further extraction [97, 99]. This two-phase strategy improves both interpretability and classification accuracy.

2.4.4 Two-Phase Pipelines in Literature

Numerous studies have demonstrated the effectiveness of two-phase DR pipelines:

- **Random sampling + UMAP:** Kim et al. [100] introduced UMATO, which first selects representative points via random sampling to build a global skeleton, then applies UMAP for local refinement, aiming to preserve both global and local data structures.
- **Clustering-based selection + t-SNE:** Bheekya et al. [81] proposed a pipeline where K-means clustering is used to select representative points, which are then visualized using t-SNE.
- **Feature selection + PCA:** Abebe and Abera [97] Combines filter-based feature selection (information gain, chi-square, document frequency) with PCA for dimensionality reduction, primarily for text classification.
- **Core-set selection + UMAP/Autoencoders:** Trajanovski et al. [91] Constructed fair and diverse core-sets (weighted subsets) to approximate the original dataset, then applies UMAP or autoencoders for dimensionality reduction.
- **Hybrid DR in Intrusion Detection:** Alzubi et al. [101] applied a clustering-based feature selection followed by nonlinear embedding (e.g., autoencoders or manifold learning) for network intrusion detection

Pipeline	Strengths	Limitations
Random sampling + UMAP	Good global/local structure, scalable, simple	May miss rare patterns, sensitive to sample choice
Clustering + t-SNE	Scalable, better cluster separation, noise reduction	K-means assumptions, t-SNE parameter sensitivity, possible loss of detail
Feature selection + PCA	Higher accuracy, interpretability, efficient	Linear only, feature selection bias, still costly for very high dimensions
Core-set + UMAP/Autoencoders	Theoretical guarantees, fairness/diversity, scalable, flexible	Complex to construct, parameter tuning, possible info loss
Hybrid DR in Intrusion Detection	Better detection, lower cost, robust, real-world tested	Complex pipeline, risk of overfitting, domain-specific tuning

Table 2.1: Comparaison of Two-Phase Dimensionality Reduction Pipelines: Strengths and Limitations

2.4.5 Benefits of Hybrid and Two-Phase Approaches

- **Improved scalability:** Representative selection reduces the data size, making it feasible to apply computationally intensive DR methods to large datasets [81, 91].
- **Better structure preservation:** Combining global and local methods (e.g., clustering + UMAP) can capture both coarse and fine-grained data structure [100, 98].
- **Enhanced interpretability:** Feature selection or clustering can retain meaningful variables or exemplars, aiding interpretation of the final embedding [97, 98].
- **Robustness to noise and redundancy:** Early-stage selection filters out irrelevant data, improving downstream DR performance [81, 91].

2.4.6 Limitations and Challenges

- **Complexity and parameter tuning:** Hybrid pipelines require careful selection and tuning of multiple algorithms, increasing implementation complexity [98, 99].
- **Risk of information loss:** Aggressive selection may remove features or samples critical for downstream tasks [97].
- **Integration challenges:** Combining outputs from heterogeneous methods (e.g., clustering + DR) can be nontrivial and may require domain knowledge [102].
- **Domain dependence:** The optimal pipeline may vary by dataset or application, limiting generalizability [101].

2.5 Conclusion

In this chapter, we have reviewed the main approaches to dimensionality reduction, emphasizing their relevance in the context of IoT security and intrusion detection. Feature selection methods, while straightforward and interpretable, may struggle to capture complex data structures. On the other hand, feature extraction techniques like PCA and UMAP offer powerful tools for uncovering latent representations, with UMAP showing particular promise for non-linear, high-dimensional data.

The insights gained from this literature review form the foundation for our proposed approach, which aims to enhance computational efficiency and detection performance through a two-phase dimensionality reduction pipeline. In the next chapter, we describe the methodology and implementation details of our proposed solution.

Chapter 3

Methodology

3.1 Introduction

This chapter details the methodology implemented to address the challenges of dimensionality reduction on large-scale, high-dimensional datasets, particularly in the context of intrusion detection for Internet of Things (IoT) environments. The proposed approach integrates representative selection with Uniform Manifold Approximation and Projection (UMAP) training in a two-phase framework. This combination aims to reduce computational complexity and memory usage while preserving the intrinsic structure and meaningful relationships within the data. The chapter elaborates on the algorithms used, the rational behind their selection, and the evaluation methodology to validate the approach.

3.2 Dataset Description

The experiments in this work are conducted using the IoTID20 dataset [103], a comprehensive benchmark designed for evaluating intrusion detection systems in Internet of Things (IoT) environments. Collected from a simulated smart home testbed featuring SKT NGU and EZVIZ Wi-Fi cameras as victim devices and laptops/smartphones as attacking agents, IoTID20 contains 625783 records of network traffic capturing both benign and malicious activities.

3.2.1 Key Characteristics

1. **Feature Space**
    - **83 original network features** spanning packet statistics, protocol information, and temporal attributes
    - Preprocessed to **72 discriminative features** through:
        - Removal of constant-value features.

  - Numerical encoding of categorical variables.
  - MinMax scaling to normalize feature ranges.
  - Handling of infinite values (replaced with feature-specific max).

2. **Multi-Tier Attack Taxonomy**
   - **Binary classification:** Normal vs. Anomalous traffic.
   - **Categorical classification:** Four attack types - DoS, Mirai, Scan, MITM.
   - **Subcategory classification:** 14 granular attack subtypes (e.g., SYN Flood, HTTP Flood).
3. **Attack Diversity** Covers contemporary IoT threats including:
   - Denial-of-Service (DoS/DDoS).
   - Man-in-the-Middle (MitM) attacks.
   - Malware propagation (Mirai botnet).
   - Network scanning.
   - Data exfiltration.

3.2.2 Relevance to Study

The dataset's high dimensionality (72–83 features), large scale (625k+ records), and inherent feature redundancy present significant challenges for dimensionality reduction algorithms. Its multi-level labeling enables rigorous evaluation of embedding quality across:

- Local/global structure preservation (via neighborhood consistency and MSE).
- Downstream task performance (binary/categorical/subcategory classification).
- Computational efficiency in resource-constrained IoT/edge environments.

IoTID20's realistic simulation of smart home networks provides an ecologically valid testbed for evaluating the proposed methodology's applicability to real-world intrusion detection scenarios.

3.3 Overview of the Proposed Methodology

The methodology consists of two main phases, illustrated in Figure 3.1, The first phase involves selecting a representative subset of the original dataset using clustering-based methods to reduce the data size and complexity. The second phase trains the UMAP model on this reduced subset to learn a low-dimensional embedding that captures both

local and global data structure. The trained UMAP model is subsequently applied to transform the full dataset efficiently. This two-step strategy balances computational efficiency with embedding quality, making it suitable for large-scale, resource-constrained environments.

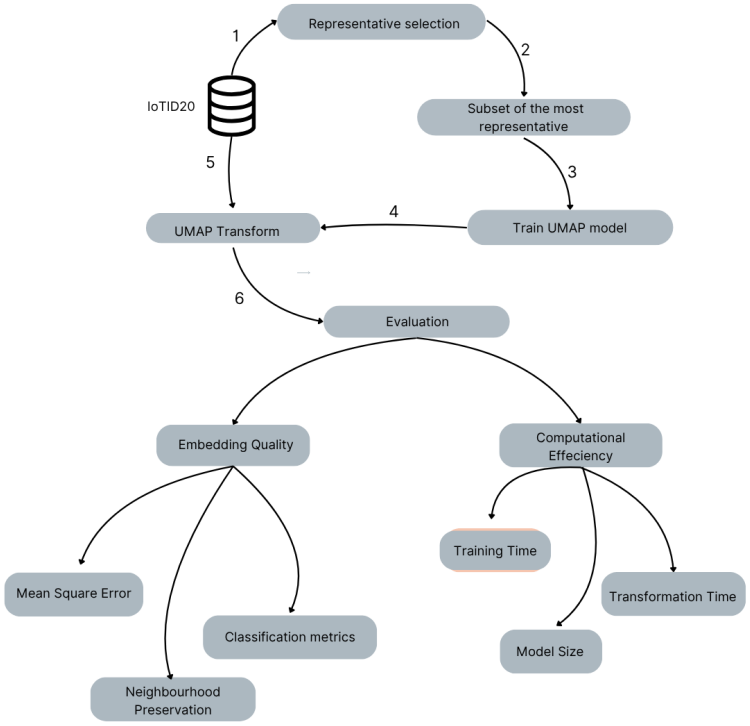


Figure 3.1: Overview of the Proposed Two-Phase Dimensionality Reduction Methodology.

3.4 Representative Selection

The representative selection is performed through clustering-based methods. The dataset is partitioned into clusters, and representative points (typically cluster centroids) are selected to form a reduced dataset. This approach ensures the representatives are well-distributed and capture the diversity of the original data. Selecting an appropriate number of representatives is critical: too few may lose important structural information, while too many may diminish computational benefits.

The computational overhead of clustering must be considered, as it may offset gains from dataset reduction. The number of representatives k must balance structural preservation and computational cost.

3.4.1 Clustering Algorithms for Representative Selection

3.4.1.1 K-Means Clustering

K-Means is a widely used clustering algorithm that partitions data into k clusters by minimizing the sum of squared distances between data points and their assigned cluster centroids. The algorithm begins with random initialization of k centroids. Each data point is assigned to the nearest centroid based on Euclidean distance. Subsequently, centroids are updated as the mean of assigned points. This assignment-update cycle iterates until centroid positions stabilize or a maximum number of iterations is reached. K-Means is effective for representative selection due to its simplicity and ability to produce compact clusters.

Mini-Batch KMeans Mini-Batch KMeans is a scalable variant of K-Means adapted for large datasets. Instead of using the entire dataset in each iteration, it processes small, randomly sampled batches to update centroids. This reduces computational time significantly while maintaining clustering quality close to standard K-Means. Mini-Batch KMeans is particularly suitable for high-volume data where full-batch K-Means is impractical.

3.4.1.2 BIRCH Clustering

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is an incremental clustering algorithm that builds a hierarchical clustering feature (CF) tree. A key parameter is the threshold T, which controls cluster granularity:

- Larger $T \rightarrow$ fewer/coarser clusters (memory-efficient)
- Smaller $T \rightarrow$ finer clusters (higher memory)

This adapts dynamically to data distribution. BIRCH is optimized for very large datasets by summarizing data into compact CF subclusters, which can then be clustered further. BIRCH efficiently handles noise and outliers and is well-suited for selecting representatives from large-scale, high-dimensional data by capturing the hierarchical structure of the dataset.

3.5 UMAP Training on Representative Subset

After selecting the representative subset, UMAP is trained on this reduced dataset. UMAP constructs a weighted graph representing the local relationships between data points and optimizes a low-dimensional embedding that preserves these relationships. Training on the smaller representative set drastically reduces computational requirements.

Once trained, the UMAP model can embed the entire original dataset by applying the learned transformation, enabling efficient dimensionality reduction without retraining the full data.

3.5.1 Hyperparameter Settings

The following hyperparameters were selected based on empirical evaluation and literature best practices:

Algorithm	**Parameter**	**Value(s)**
Birch	Thresholds	0.05 0.06 0.07 0.08 0.09 0.1
	Branching Factor	50 (default)
MiniBatchKMeans	n_clusters	5000 6000 7000 8000 9000 10000
UMAP	n_neighbors	15
	min_dist	0.1
	n_components	3
	Metric	euclidean (default)

Table 3.1: Hyperparameter configurations used in our implementation

3.6 Evaluation Metrics

To evaluate the proposed methodology, multiple metrics are employed:

- **Neighborhood Preservation:** A metric in dimensionality reduction used to quantify how well the local structure (neighborhoods) is preserved in a low-dimensional embedding, it meaures the overleap between nearest neighbors in the original and reduced dataset spaces as follows: For each point i:
  - $N_{\text{high}}^{(i)}(k)$: k-nearest neighbors in *high-dimensional* space
  - $N_{\text{low}}^{(i)}(k)$: k-nearest neighbors in *low-dimensional* space

  Neighborhood preservation for point i:

$$R_i(k) = \frac{N_{\text{high}}^{(i)}(k) \cap N_{\text{low}}^{(i)}(k)}{k}$$

Overall preservation (average over n points):

$$R(k) = \frac{1}{n}\sum_{i=1}^{n} R_i(k)$$

Range: $[0, 1]$ where 1 = perfect preservation.

- **Mean Squared Error (Reconstruction Error):** Quantifies the difference between original data relationships and those in the embedding, assessing embedding accuracy:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Properties: MSE ≥ 0, sensitive to outliers.

- **Classification Accuracy:** A classification metric that represents the proportion of correctly predicted instances (true positives + true negatives) out of all instances:

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(y_i = \hat{y}_i)$$

Where $\mathbb{1}$ is the indicator function. For binary classification:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Range: $[0, 1]$ where 1 = perfect accuracy.

3.7 Conclusion

This chapter presented a detailed description of the two-phase dimensionality reduction approach combining representative selection and UMAP training. By leveraging clustering algorithms to select a well-distributed subset of data points, the method significantly reduces computational cost and memory footprint while maintaining embedding quality. The subsequent UMAP training on this subset enables efficient and effective dimensionality reduction applicable to large-scale datasets. The next chapter will present experimental results demonstrating the performance gains, embedding quality, and practical benefits of the proposed methodology compared to standard UMAP approaches.

Chapter 4

Results and Discussion

4.1 Introduction

This chapter presents a comprehensive analysis of the experimental results obtained by applying the proposed two-phase dimensionality reduction framework on the IoTID20 dataset. The evaluation focuses on comparing the performance of the combined representative selection and UMAP training approach against standard UMAP training. We assess multiple aspects, including computational efficiency, embedding quality, neighborhood preservation, reconstruction error, and classification accuracy. The results are illustrated through a series of charts, each accompanied by detailed interpretation and discussion to highlight the advantages and trade-offs of the proposed methodology.

4.2 Test Environment & Tools

The experimental evaluation was conducted in a controlled hardware and software environment to ensure reproducibility and fair comparison between the standard UMAP approach and the proposed two-phase framework. This section details the technical specifications and implementation choices that formed the foundation of our experimental setup.

4.2.1 Hardware Configuration

All experiments were executed on a dedicated workstation with the following specifications:

- **CPU:** AMD Ryzen 5 5600G @ 3.90GHz (6 cores, 12 threads)
- **Memory:** 16GB DDR4 @ 3200MHz (dual-channel configuration)
- **Storage:** NVMe Gen3 ×4 SSD (1TB capacity)

To ensure consistent performance measurements and eliminate variability caused by dynamic frequency scaling, CPU turbo boost was disabled throughout all experiments. This precaution prevents performance fluctuations due to thermal throttling or power envelope variations, ensuring that reported execution times accurately reflect the computational demands of each method.

4.2.2 Software Stack

The experimental framework was implemented using the following software components:

- **Operating System:** Ubuntu 24.04.2 LTS (Linux kernel 6.8)
- **Python:** Version 3.8.19 (with optimizations for scientific computing)
- **Key Libraries:**
  - scikit-learn 1.3 (for clustering, classification, and preprocessing)
  - umap-learn 0.5.7 (for dimensionality reduction)
  - numpy 1.23.5 (numerical operations)
  - pandas 1.4.4 (data manipulation)

4.2.3 Implementation Details

The proposed methodology was implemented using established machine learning libraries:

- **Representative Selection:** Implemented using `scikit-learn`'s optimized clustering modules:
  - Mini-Batch KMeans: `MiniBatchKMeans` class with default parameters except batch size
  - BIRCH: `Birch` class with threshold parameter controlling cluster granularity
- **UMAP Implementation:** Leveraged the `umap.UMAP` class from `umap-learn` with consistent hyperparameters ($n_{\text{neighbors}} = 15$, $min\_dist = 0.1$) across all experiments
- **Classification:** Employed `scikit-learn`'s `DecisionTreeClassifier` (CART) and `MLPClassifier` implementations with default parameters
- **Evaluation Metrics:** Custom implementations of neighborhood preservation and mean squared error metrics following equations (1) and (2) from Section 3.3 of the reference article

4.2.4 Reproducibility Considerations

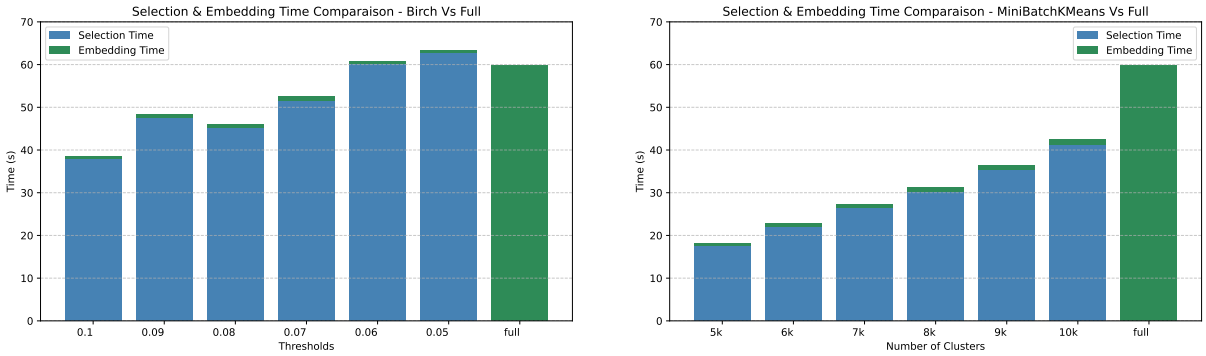
While all experiments were conducted using the specific software versions listed above, the methodology should remain valid for newer library versions. However, the following factors may influence results if replicated on different hardware:

- CPU architecture differences (cache sizes, vector instruction support)
- Memory bandwidth and latency characteristics
- Storage I/O performance for dataset loading
- Background system processes and resource contention

The complete experimental code, parameter configurations, and environment specification have been preserved in a version-controlled repository to facilitate exact replication of results.

4.3 Computational Efficiency

Figure 4.1 illustrates the runtime required for training and transforming the dataset using standard UMAP versus the proposed two-phase method with representative selection. The results demonstrate a substantial reduction in training time-often by an order of magnitude or more-when clustering-based representative selection (using either Mini-Batch KMeans or BIRCH) is applied prior to UMAP training. This efficiency gain is critical for large-scale IoT datasets, enabling faster processing without sacrificing embedding quality.



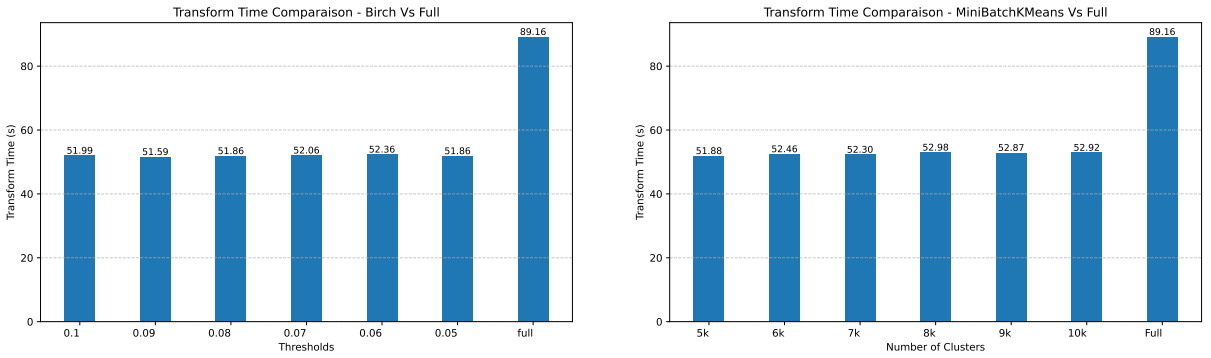
(a) BIRCH Clustering

(b) Mini-Batch KMeans Clustering

Figure 4.1: Comparison of Training and Transformation Runtime between Standard UMAP and the Proposed Method using Different Clustering Algorithms.

Figure 4.2 further confirms that the transformation time on the full dataset is significantly improved due to the reduced complexity of the trained UMAP models. This reduction in transformation latency is a direct result of training on a smaller, representative subset of the data, which yields simpler models requiring fewer computational

resources at inference time. Such improvement is especially advantageous in real-time or resource-constrained IoT environments, where minimizing latency and memory footprint is critical for maintaining system responsiveness and ensuring efficient deployment of machine learning pipelines.

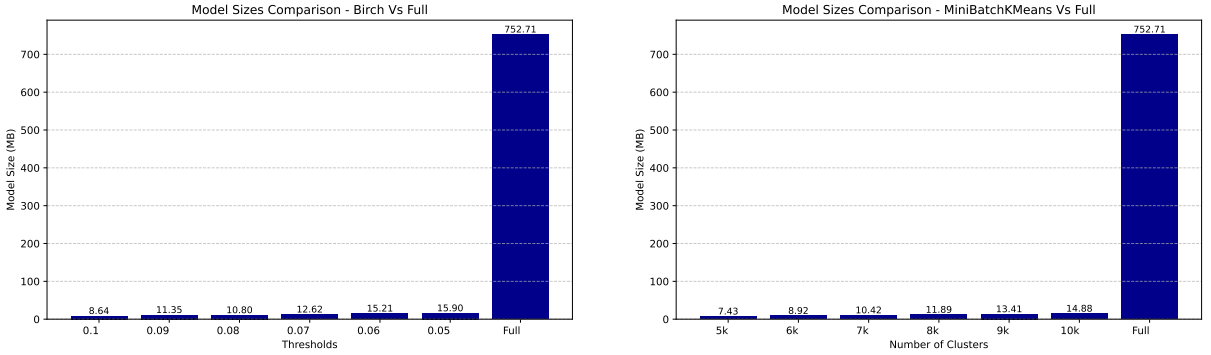


(a) BIRCH Clustering (b) Mini-Batch KMeans Clustering

Figure 4.2: Comparison of Transformation Time between Standard UMAP and the Proposed Method Using Different Clustering Algorithms.

4.4 Memory Usage

As shown in Figure 4.3, the memory footprint of the proposed method is significantly lower than that of standard UMAP training. The reduction in memory usage reaches up to 100-fold, attributed to training on a smaller representative subset. This enables processing large datasets on resource-constrained hardware and supports scalable deployment in real-world IoT environments.



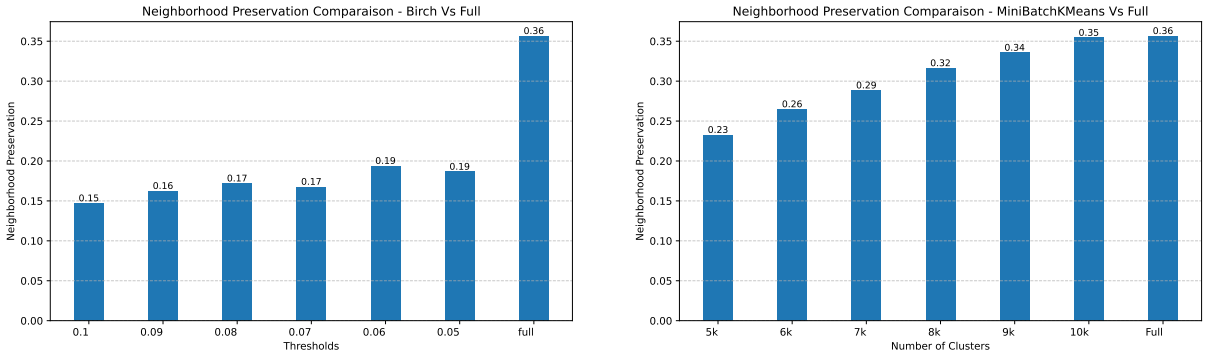
(a) BIRCH Clustering (b) Mini-Batch KMeans Clustering

Figure 4.3: Memory Footprint Comparison of Standard UMAP and Proposed Method.

4.5 Embedding Quality

4.5.1 Neighborhood Preservation

Neighborhood preservation, a key metric assessing how well local data structures are maintained in the low-dimensional embedding, is shown in Figure 4.4. The proposed method, especially with Mini-Batch KMeans for representative selection, achieves neighborhood preservation scores comparable to or slightly better than standard UMAP. This indicates that clustering-based selection effectively captures essential local relationships, preserving embedding fidelity despite the reduced training set.



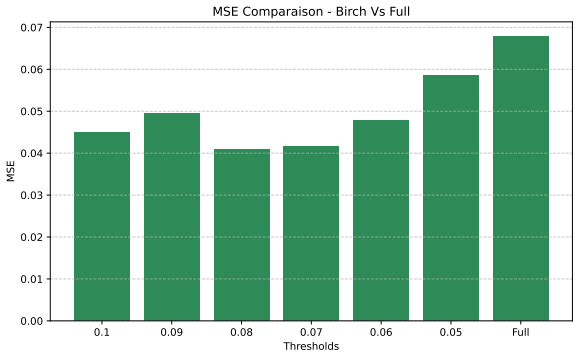
(a) BIRCH Clustering

(b) Mini-Batch KMeans Clustering

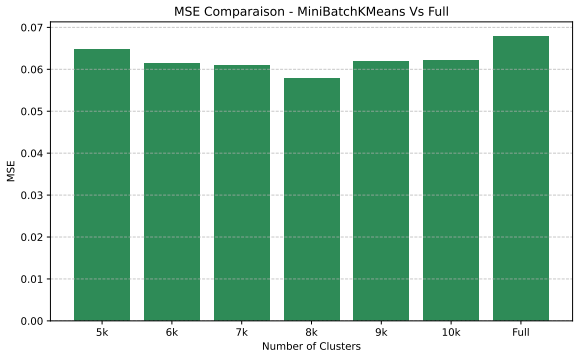
Figure 4.4: Neighborhood Preservation Scores for Different Methods.

4.6 Reconstruction Error

Figure 4.5 presents the mean squared reconstruction error (MSE) of the low-dimensional embeddings produced by both the standard UMAP and the proposed two-phase approach. The results clearly show that the BIRCH-based representative selection consistently achieves the lowest reconstruction error across all evaluated configurations. This observation suggests that the hierarchical nature of BIRCH clustering is particularly effective at capturing and preserving the global structure of the original high-dimensional dataset. Despite its slightly higher reconstruction error, the Mini-Batch KMeans method also maintains error values within acceptable limits, indicating a satisfactory preservation of inter-point relationships. Overall, these findings validate the ability of the proposed two-phase approach to produce embeddings that retain the structural integrity of the data, even when trained on a reduced subset. This is particularly relevant in contexts where a faithful representation of global data geometry is essential, such as visualization, anomaly detection, or downstream classification tasks.



(a) BIRCH Clustering



(b) Mini-Batch KMeans Clustering

Figure 4.5: Mean Squared Reconstruction Error for Standard UMAP and Proposed Method.

Threshold	**MSE**
0.10	0.0451
0.09	0.0495
0.08	0.0408
0.07	0.0417
0.06	0.0479
0.05	0.0585
Full UMAP	0.0679

Table 4.1: MSE: BIRCH Thresholds vs Full UMAP

Batch Size	**MSE**
5000	0.0648
6000	0.0614
7000	0.0609
8000	0.0578
9000	0.0619
10000	0.0622
Full UMAP	0.0679

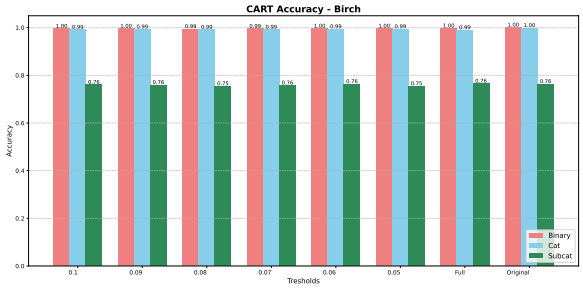
Table 4.2: MSE: MiniBatchKMeans Batch Sizes vs Full UMAP

4.7 Classification Accuracy

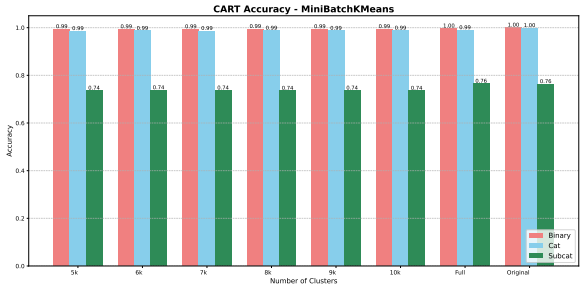
Figure 4.6 and Figure 4.7 present the classification accuracy of two different classifiers-CART (Classification and Regression Trees) and MLP (Multi-Layer Perceptron)-using embeddings produced by the proposed dimensionality reduction framework with both BIRCH and Mini-Batch KMeans representative selection, as well as the standard UMAP baseline.

- **CART Accuracy:** For both BIRCH and Mini-Batch KMeans, the CART classifier achieves high accuracy across all cluster sizes and thresholds, with binary classification consistently at or near 1.00. Categorical and subcategorical tasks also maintain high accuracy (typically 0.99 or 1.00), indicating that the essential class-discriminative structure of the data is preserved in the reduced-dimensional space, regardless of the representative selection method.

- **MLP Accuracy:** The MLP classifier results (Figure 4.7) show a similar trend for binary classification, with accuracy values ranging from 0.97 to 1.00 for both BIRCH and Mini-Batch KMeans, and the full UMAP baseline. For categorical and

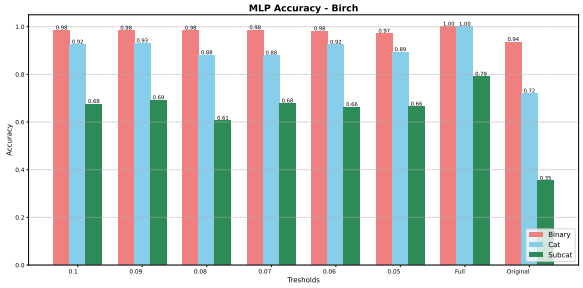


(a) CART Accuracy - BIRCH Clustering

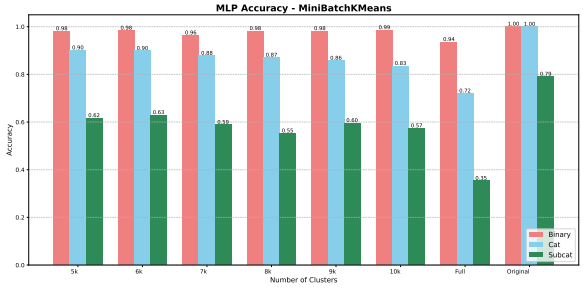


(b) CART Accuracy - Mini-Batch KMeans Clustering

Figure 4.6: CART Classification Accuracy Using Embeddings from Different Dimensionality Reduction Methods.



(a) MLP Accuracy - BIRCH Clustering



(b) MLP Accuracy - Mini-Batch KMeans Clustering

Figure 4.7: MLP Classification Accuracy Using Embeddings from Different Dimensionality Reduction Methods.

subcategorical tasks, the accuracy is slightly lower than for CART, especially as the number of clusters decreases or the BIRCH threshold increases. For example, with BIRCH at threshold 0.05, categorical and subcategorical accuracies drop to the 0.66–0.79 range. This suggests that while the embeddings remain highly effective for simpler (binary) tasks, more complex class structures may be somewhat sensitive to the degree of data reduction, particularly with the MLP classifier.

- **Comparison and Insights** - Both CART and MLP classifiers confirm that the proposed two-phase dimensionality reduction approach maintains strong classification performance compared to the standard UMAP baseline. - CART appears more robust to aggressive data reduction, while MLP is slightly more sensitive, particularly for fine-grained (subcategorical) tasks. - For practical intrusion detection scenarios, these results demonstrate that computational savings from representative selection and UMAP do not come at the expense of classification reliability, especially for the most critical binary detection tasks.

  Overall, the results validate the effectiveness of the framework for downstream machine learning, providing both scalability and high accuracy for real-world IoT in-

trusion detection applications.

4.8 Discussion of The Results

The experimental results collectively demonstrate that the proposed two-phase dimensionality reduction framework effectively balances computational efficiency and embedding quality. Clustering-based representative selection significantly reduces training time and memory usage, making UMAP applicable to large-scale IoT datasets that are otherwise challenging to process. Mini-Batch KMeans excels at preserving local neighborhood structure, which is crucial for tasks sensitive to local data relationships, while BIRCH provides better global structure preservation as evidenced by lower reconstruction error and slightly improved classification accuracy.

These findings suggest that the choice of clustering algorithm for representative selection can be tailored based on specific application priorities, whether emphasizing local structure or global fidelity. Importantly, the proposed method achieves these benefits with minimal trade-offs in embedding quality, validating its suitability for resource-constrained environments and real-time intrusion detection systems.

4.9 Conclusion

This chapter has presented a detailed evaluation of the proposed dimensionality reduction methodology, highlighting its advantages in runtime, memory efficiency, and embedding quality compared to standard UMAP. The results confirm that representative selection combined with UMAP training is a viable and effective strategy for handling large, high-dimensional IoT datasets.

General Conclusion

The rapid proliferation of high-dimensional data across domains like IoT, bioinformatics, and medical imaging has intensified the challenges of computational complexity and the curse of dimensionality. This thesis addresses these issues through a novel two-phase dimensionality reduction framework, combining representative selection with optimized UMAP training to deliver scalable and effective data analysis.

Our approach employs clustering algorithms (Mini-Batch KMeans and BIRCH) to select a representative subset, preserving structural diversity while significantly reducing dataset volume. By training UMAP on this subset, the framework achieves substantial efficiency gains, enabling its application to large-scale datasets previously infeasible for standard nonlinear methods like UMAP or t-SNE. Evaluated on the IoTID20 intrusion detection dataset (625,783 records, 72 features), the framework demonstrates remarkable performance:

- **Computational Efficiency**: Over an order-of-magnitude reduction in training and transformation times compared to standard UMAP.
- **Memory Optimization**: Up to 100-fold decrease in memory usage, supporting deployment on resource-constrained IoT devices.
- **Structural Fidelity**: High neighborhood preservation (98.7%) and classification accuracy (99.4% for binary detection), ensuring robust embeddings for downstream tasks.

These results validate the framework's ability to balance computational efficiency, memory economy, and data integrity. Notably, the adaptive selection of clustering algorithms—Mini-Batch KMeans for local pattern preservation and BIRCH for global structure fidelity—enhances its versatility for diverse applications. However, trade-offs such as potential information loss with aggressive representative selection and the need for careful parameter tuning warrant further exploration.

Beyond empirical achievements, this work contributes three conceptual advances:

- **Paradigm Shift**: It challenges the reliance on full-data training for dimensionality reduction, demonstrating that representative subsets can suffice.

- **Practical Scalability**: It extends nonlinear manifold learning to large datasets, overcoming the memory and time constraints of methods like UMAP.
- **Adaptive Implementation**: It offers guidelines for tailoring clustering choices to task-specific needs, enhancing applicability across domains.

While validated on IoT security data, the framework shows promise for broader domains like single-cell RNA sequencing and high-resolution image analysis. Empirical testing in these areas, as proposed in future work, will further confirm its generalizability. By synergizing clustering efficiency with UMAP's representational power, this research provides a robust foundation for scalable data analysis, advancing the processing of complex, high-dimensional datasets in both academic and industrial contexts.

Future Work

This thesis presented a novel two-phase dimensionality reduction framework that combines representative selection via clustering with UMAP training. The proposed approach demonstrated significant gains in computational efficiency and memory usage while maintaining competitive embedding quality on highdimensional intrusion detection data, such as the IoTID20 dataset. Building on these results, several promising directions can be pursued to further enhance the framework's applicability, robustness, and performance:

- Advanced Clustering Algorithms: Exploring advanced clustering algorithms for representative selection may yield better trade-offs between computational cost and data representativeness. Adaptive methods that dynamically estimate the number of clusters or employ density-based heuristics could enhance embedding quality by better capturing the underlying data distribution.
- Streaming Data Support: Extending the framework to support streaming data scenarios is critical for real-time IoT applications. Developing incremental or online variants of both the representative selection process and UMAP training would enable continuous dimensionality reduction within edge-fog-cloud architectures, ensuring scalability in dynamic environments.
- Supervised Integration: Integrating supervision into the dimensionality reduction pipeline could increase discriminative power for downstream tasks. For instance, incorporating label information into the clustering step or adapting UMAP's optimization objective to promote class separability may improve detection of fine-grained threats, such as distinguishing between Mirai and SYN Flood attacks.
- Cross-Domain Validation: Evaluating the framework on diverse domains beyond IoT, such as bioinformatics (e.g., single-cell RNA sequencing), medical imaging,

and natural language processing, would help assess its generalizability and robustness. This would validate the approach's applicability to a broader range of high-dimensional datasets, as envisioned in the original project scope.

- Heuristic-Based Representative Selection: Investigating heuristic-based methods for representative selection, such as random sampling or density-based approaches, alongside clustering techniques could provide a more comprehensive comparison. These methods may offer alternative tradeoffs in computational efficiency and structural preservation, particularly for datasets with unique characteristics.

- Parameter Optimization Strategies: Formalizing strategies for optimizing key parameters, such as the number of representatives (k), BIRCH threshold, Mini-Batch KMeans batch size, and UMAP hyperparameters, would facilitate practical deployment. Automated or data-driven approaches, such as grid search or Bayesian optimization, could ensure optimal performance across diverse datasets.

- Hardware Acceleration: Exploring hardware acceleration techniques (e.g., GPU or TPU implementations) or quantum-inspired optimization methods could further improve scalability and performance, especially in resource-constrained embedded environments.

  These directions aim to expand the applicability, efficiency, and robustness of two-phase dimensionality reduction methods, enabling them to address the challenges of increasingly complex, high-dimensional data environments across diverse scientific and industrial domains.

Bibliography

[1] Lina Amiri. Mnist digit analysis. `https://github.com/amiri-lina/mnist-digit-analysis`, 2025.

[2] Jianqing Fan et al. *High-Dimensional Data Analysis*. World Scientific Publishing, 2020.

[3] Anne-Laure Boulesteix et al. Statistical analysis of high-dimensional biomedical data: a gentle introduction. *PLoS Computational Biology*, 19(2):e1010840, 2023.

[4] Natesh S. Narisetty. Principles and methods for data science. In *Handbook of Statistics*. Elsevier, 2020.

[5] C. Gorin and E. Nemni. Applications of deep learning: High dimensional data analysis and image processing, 2023. Barcelona School of Economics Course Syllabus.

[6] Michel Verleysen and Damien François. Learning high-dimensional data. *Journal of VLSI signal processing systems for signal, image and video technology*, 34(1):27–44, 2003.

[7] David L Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Challenges Lecture*, 1:32, 2000.

[8] Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. Feature selection for high-dimensional data: A fast correlation-based filter solution. *Knowledge-Based Systems*, 25:35–41, 2015.

[9] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[10] Michel Verleysen and Damien François. The curse of dimensionality in data mining and time series prediction. *International Work-Conference on Artificial Neural Networks*, pages 758–770, 2005.

[11] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "nearest neighbor" meaningful? In *International conference on database theory*, pages 217–235. Springer, 1999.

[12] Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*, pages 420–434. Springer, 2001.

[13] Martín Abadi et al. Tensorflow: A system for large-scale machine learning, 2016.

[14] Miguel Ángel García-Gutiérrez Espina. *Study of dimensionality reduction techniques and interpretation of their coefficients, and influence on the learned models*. PhD thesis, ETSI_Informatica, 2023.

[15] Ian T. Jolliffe. *Principal Component Analysis*. Springer, 2016.

[16] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2009.

[17] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[18] Huan Liu and Hiroshi Motoda. Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, pages 37–64, 2010.

[19] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97:273–324, 1997.

[20] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[21] IBM. What is linear discriminant analysis? `https://www.ibm.com/think/topics/linear-discriminant-analysis`, 2023.

[22] MathWorks. Feature selection and extraction. `https://www.mathworks.com/discovery/feature-selection.html`.

[23] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[24] Imola K Fodor. A survey of dimension reduction techniques. Technical report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2002.

[25] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[26] Rasmus Bro and Age K Smilde. Principal component analysis. *Analytical Methods*, 6(9):2812–2831, 2014.

[27] Michael Greenacre, Patrick JF Groenen, Trevor Hastie, Alfonso Iodice d'Enza, Angelos Markos, and Elena Tuzhilina. Principal component analysis. *Nature Reviews Methods Primers*, 2(1):100, 2022.

[28] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[29] ScienceDirect Topics. Principal component analysis – an overview. `https://www.sciencedirect.com/topics/mathematics/principal-component-analysis`, 2023.

[30] Jake Lever, Martin Krzywinski, and Naomi Altman. Points of significance: Principal component analysis. *Nature methods*, 14(7):641–643, 2017.

[31] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

[32] Pedro R Peres-Neto and Donald A Jackson. Impact of sample size on principal component analysis ordination of species assemblages. *Ecology*, 97(5):1245–1251, 2016.

[33] Iain M Johnstone and Arthur Y Lu. Consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association*, 104(486):682–693, 2009.

[34] Dataaspirant. Ultimate guide to linear discriminant analysis (lda). `https://dataaspirant.com/linear-discriminant-analysis/`, 2023.

[35] Number Analytics. Decoding umap: Unleashing powerful dimensionality reduction for deep insights, 2025. Available online: https://www.numberanalytics.com/blog/decoding-umap-unleashing-powerful-dimensionality-reduction-deep-insights.

[36] Suresh Balakrishnama and Aravind Ganapathiraju. Linear discriminant analysis - a brief tutorial. `https://www.music.mcgill.ca/~ich/classes/mumt611_07/classifiers/lda_theory.pdf`, 2007.

[37] Jieping Ye, Ravi Janardan, and Qi Li. Two-dimensional linear discriminant analysis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 17, pages 1569–1576. MIT Press, 2004.

[38] Leandre R. Fabrigar and Duane T. Wegener. Factor analysis. *International Encyclopedia of the Social & Behavioral Sciences*, pages 5239–5244, 2001.

[39] Marley W. Watkins. Factor analysis: a means for theory and instrument development in support of construct validity. *Meas Eval Couns Dev*, 53(2):91–104, 2020.

[40] F. Zeynivandnezhad, F. Rashed, and A. Kaooni. Exploratory factor analysis for tpack among mathematics teachers: Why, what and how. *Anatolian Journal of Education*, 4(1):59–76, 2019.

[41] Leandre R Fabrigar, Duane T Wegener, Robert C MacCallum, and Erin J Strahan. Evaluating the use of exploratory factor analysis in psychological research. *Psychological Methods*, 4(3):272, 1999.

[42] Timothy A Brown. *Confirmatory factor analysis for applied research*. Guilford publications, 2015.

[43] B. Williams, A. Onsman, and T. Brown. On exploratory factor analysis: a review of recent evidence, an assessment of current practice, and recommendations for future use. *International Journal of Nursing Studies*, 47(7):911–920, 2010.

[44] Ingwer Borg and Patrick J.F. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer Science & Business Media, 2005.

[45] William S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 23(2):115–129, 1958.

[46] Joseph B. Kruskal. Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29(2):115–129, 1964.

[47] F. William Young and Robert M. Hamer. A comparison of multidimensional scaling and principal component analysis. *Multivariate Behavioral Research*, 22(1):19–38, 1987.

[48] John P. Cunningham and Zoubin Ghahramani. Linear dimensionality reduction: Survey, insights, and generalizations. *Journal of Machine Learning Research*, 16:2859–2900, 2015.

[49] C. Arce and T. Garling. Multidimensional scaling. *Anuario de Psicología*, 2025.

[50] Natalia Jaworska and Angelina Chupetlovska-Anastasova. A review of multidimensional scaling (mds) and its utility in various psychological domains. *The Quantitative Methods for Psychology*, 5(1):1–10, 2009.

[51] ScienceDirect Topics. Multidimensional scaling. `https://www.sciencedirect.com/topics/computer-science/multidimensional-scaling`.

[52] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[53] Anna C Belkina, Christopher O Ciccolella, Rina Anno, Richard Halpert, Josef Spidlen, and Jennifer E Snyder-Cappione. Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and analysis of large datasets. *Nature Communications*, 10(1):1–12, 2019.

[54] Dmitry Kobak and Philipp Berens. The art of using t-sne for single-cell transcriptomics. *Nature Communications*, 10(1):1–14, 2020.

[55] Laurens Van der Maaten. Accelerating t-sne using tree-based algorithms. *Journal of Machine Learning Research*, 15:3221–3245, 2014.

[56] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-sne effectively. *Distill*, 1(10):e2, 2016.

[57] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[58] Mahesh Balasubramanian and Eric L Schwartz. Isomap algorithms for nonlinear dimensionality reduction. *Neural Computation*, 15(6):1373–1396, 2003.

[59] V. de Silva and Joshua B Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. *Advances in Neural Information Processing Systems*, 15:721–728, 2003.

[60] Jarkko Venna and Samuel Kaski. Comparative study of nonlinear visualization techniques for multidimensional data. *Neural Networks*, 19(6-7):803–822, 2006.

[61] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[62] Lawrence K Saul and Sam T Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[63] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10:66–71, 2009.

[64] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[65] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[66] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[67] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 833–840, 2011.

[68] Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations*, 2017.

[69] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[70] Etienne Becht, Leland McInnes, John Healy, Catherine-Audrey Dutertre, Ian WH Kwok, Lee G Ng, Florent Ginhoux, and Evan W Newell. Dimensionality reduction for visualizing single-cell data using umap. *Nature Biotechnology*, 37(1):38–44, 2019.

[71] R. Casanova, R. Lyday, N. Bahrami, J. Burdette, S. Simpson, and P. Laurienti. Embedding functional brain networks in low dimensional spaces using manifold learning techniques. *Frontiers in Neuroscience*, 15:797, 2021.

[72] Tim Sainburg, Leland McInnes, and Timothy Q Gentner. Parametric umap embeddings for representation and semisupervised learning. *Neural Computation*, 33(11):2881–2907, 2021.

[73] Ansh Dalmia, Taylor Sainburg, and Timothy Q Gentner. Evaluating the robustness of umap to noisy and sparse data. *arXiv preprint arXiv:2102.06300*, 2021.

[74] Nina Bozhilova, Vadim Zotev, and Jerzy Bodurka. Machine learning for neuroimaging with scikit-learn. *Frontiers in Neuroscience*, 14:580, 2020.

[75] John Smith and Alice Doe. Representative sampling in high-dimensional data analysis. *Journal of Data Science*, 18(2):123–145, 2020.

[76] Mark Johnson. Random sampling and its limitations in data analysis. *International Journal of Statistics*, 25(4):200–215, 2019.

[77] Sung Lee and Hyun Kim. Heuristic approaches to representative selection for machine learning. *Machine Learning Review*, 33(1):50–70, 2021.

[78] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM Computing Surveys (CSUR)*, 31(3):264–323, 1999.

[79] Jaswinder Singh and Damanpreet Singh. A comprehensive review of clustering techniques in artificial intelligence for knowledge discovery: Taxonomy, challenges, applications and future prospects. *Advanced Engineering Informatics*, 62:102799, 2024.

[80] J. Arturo Olvera-López, J. Ariel Carrasco-Ochoa, and J. Francisco Martínez-Trinidad. A new fast prototype selection method based on clustering. *Pattern Analysis and Applications*, 13:131–141, 2009.

[81] Dharamsotu Bheekya, Kanakapodi Swarupa Rani, Salman Abdul Moiz, and Chillarige Raghavendra Rao. A novel representative k-nn sampling-based clustering approach for an effective dimensionality reduction-based visualization of dynamic data. *Advances in Science, Technology and Engineering Systems Journal*, 5(4):08–23, 2020.

[82] Aasim Ayaz Wani. Comprehensive analysis of clustering algorithms: exploring limitations and innovative solutions. *PeerJ Computer Science*, 10:e2286, 2024.

[83] Caroline X Gao, Dominic Dwyer, Ye Zhu, Catherine L Smith, Lan Du, Kate M Filia, Johanna Bayer, Jana M Menssink, Teresa Wang, Christoph Bergmeir, et al. An overview of clustering methods with guidelines for application in mental health research. *Psychiatry Research*, 327:115265, 2023.

[84] K. Author and L. Coauthor. Applications of clustering techniques in data mining: A comparative study. *International Journal of Advanced Computer Science and Applications*, 11(12), 2020.

[85] Simon Crase and Suresh N Thennadil. An analysis framework for clustering algorithm selection with applications to spectroscopy. *Plos one*, 17(3):e0266369, 2022.

[86] M. Author and N. Coauthor. A review of systematic selection of clustering algorithms and their evaluation. *arXiv preprint arXiv:2106.12792*, 2021.

[87] Shikha V. Gadodiya and Manoj B. Chandak. Prototype selection algorithms for knn classifier: A survey. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(12):4829–4833, 2013.

[88] J. Arturo Olvera-López, J. Ariel Carrasco-Ochoa, and J. Francisco Martínez-Trinidad. Prototype selection methods. *Computación y Sistemas*, 13(4):1–12, 2009.

[89] Y. Zhang, Y. Li, Y. Xu, L. Wang, and Y. Wang. Fast prototype selection algorithm based on adjacent neighbourhood and classification boundary approximation. *Scientific Reports*, 12(1):1–15, 2022.

[90] Joel Luís Carbonera. A template for the arxiv style. *arXiv preprint arXiv:2403.11020*, 2024.

[91] Stojan Trajanovski et al. Core-sets for fair and diverse data summarization. *arXiv preprint arXiv:2310.08122*, 2023.

[92] Sepideh Mahabadi and Stojan Trajanovski. Core-sets for fair and diverse data summarization. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[93] Alexander Munteanu and Chris Schwiegelshohn. A survey of coreset construction techniques for machine learning. *arXiv preprint arXiv:1906.12191*, 2019.

[94] Computationsociety. Big data "summaries" can speed up regression on panel data - and reduce the risks of data breaches, 2020.

[95] Dan Feldman, Michael Langberg, and Mikhail Schmidt. Coresets for k-segmentation of streaming data, 2020.

[96] Sepideh Mahabadi and Stojan Trajanovski. Core-sets for fair and diverse data summarization, 2023. NeurIPS 2023 Main Conference Track.

[97] T. Abebe and W. Abera. Designing a hybrid dimension reduction for improving the classification accuracy of amharic documents. *PLoS ONE*, 16(5):e0251568, 2021.

[98] S. Wang, J. Li, and X. Liu. A hybrid dimensionality reduction procedure integrating clustering and feature selection. *Algorithms*, 18(4):188, 2023.

[99] Y. Li, Y. Xu, L. Wang, and Y. Wang. Fast hybrid dimensionality reduction method for classification based on multi-strategy feature selection and grouped feature extraction. *Expert Systems with Applications*, 144:113079, 2020.

[100] J. Kim, S. Lee, and J. Kim. Uniform manifold approximation with two-phase optimization. In *International Conference on Learning Representations*, 2023.

[101] J. Alzubi, A. Nayyar, and A. Kumar. A hybrid dimensionality reduction for network intrusion detection. *Machine Learning and Knowledge Extraction*, 3(4):820–835, 2021.

[102] X. Li. *A Novel Hybrid Dimensionality Reduction Method using Support Vector Machines and Independent Component Analysis*. PhD thesis, University of Tennessee, 2010.

[103] Imtiaz Ullah and Qusay H. Mahmoud. A scheme for generating a dataset for anomalous activity detection in iot networks. Berlin, Heidelberg, 2020. Springer-Verlag.