M1 INTERNSHIP REPORT

Identify optimal configuration with a machine learning method in multi-criteria decision analysis

HEYUAN LIU

MScT AiViC
École Polytechnique

Supervised by
Pr. François MARÉCHAL
Yi ZHAO

Industrial Processes and Energy Systems Engineering (IPESE) Laboratory
Ecole Polytechnique Fédérale de Lausanne

Autumn 2024

September 4, 2024

Abstract

This report proposes a novel approach for multi-criteria decision analysis (MCDA) using advanced machine learning techniques. A hybrid dimensionality reduction method combining AutoEncoders and Principal Component Analysis (PCA) is developed to reduce high-dimensional data while retaining key features. Geometric analysis techniques, such as Intrinsic Shape Signatures, Harris Corner Detection, and Mesh Saliency, further refine the identification of typical configurations.

Meanwhile, we employs Large Language Models (LLMs) with an iterative self-update mechanism to dynamically adjust weighting strategies, enhancing decision-making capabilities in complex environments. Prompt engineering techniques are integrated to improve the model's reasoning and adaptability.

Additionally, a modified Reinforcement Learning (RL) framework extends the Deep Deterministic Policy Gradient (DDPG) approach by incorporating a multi-head attention mechanism and specialized network layers(individual and shared), enabling real-time optimization of multiple objectives under changing conditions. These methods collectively improve decision-making robustness and adaptability in dynamic industrial contexts.

Key words: Multi-Criteria Decision Analysis, Clustering Algorithm, Shape Representations, Multi-Objective Optimization, Reinforcement Learning, Large Language Model

Acknowledgements

This world was once vast and mysterious, but now it seems to be shrinking and becoming more familiar. We stand at the turning point where legends have become reality with the information world at our fingertips. In this era, the rapid advancement of artificial intelligence techniques excite us profoundly. These breakthroughs significantly boost our productivity, enabling us to manage and utilize information to enhance industrial systems more efficiently and intelligently.

It is an honor to study in École Polytechnique.

I would like to thank various people for their help and contribution to this challenging and interesting project

I would like to thank Professor François Maréchal for his precious academic support and encouragement for deploying the AI techniques in the field of Energy System.

I would like to offer a special thanks to Yi Zhao for his constant guidance and constructive critiques. He always took the time to give his feedback and thoughts towards the advancement of the project, and I am extremely grateful for that.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

In chemical engineering, balancing economic and environmental goals is crucial amidst challenges like climate change. To address conflicting objectives, tools like multi-objective optimization (MOO) and multi-criteria decision analysis (MCDA)[1] are utilized. MOO generates a range of viable solutions, while MCDA helps select the most suitable option considering factors like profitability, environmental impact, safety, and efficiency. These tools aid in making informed decisions amidst complex trade-offs and uncertainties.

Multi-criteria Decision Analysis

A set of solutions (datapoints) is generated by MOO[2] with a set of key performance indicators (KPIs) or features. To select one or several preferred configurations, two methods are generally used:

- The weighting method.
- The data-driven method.

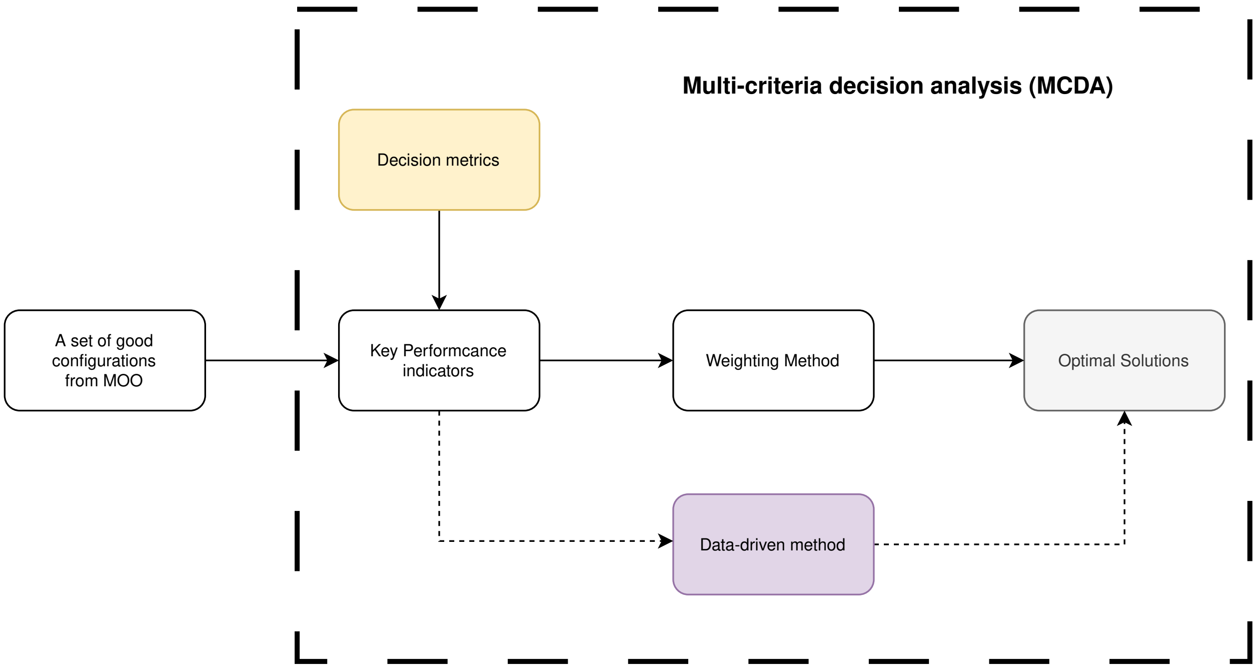

Multi-criteria decision analysis (MCDA)
Decision metrics
A set of good configurations from MOO
Key Performcance indicators
Weighting Method
Optimal Solutions
Data-driven method

Figure 1: Data-driven method in MCDA problems

A number of research have been conducted to apply machine learning methods to MCDA problem[1], ranging from the criteria selection and clustering to weights determination. The main research gaps are to select appropriate ML methods for specific problems and reveal the necessity of applying particular ML technologies . In this internship, the student will develop an ML method to support the decision-making process in the refining decarbonization process. The results will be validated with the industrial partner.

2 Related Work

2.1 Multi-Criteria Decision Analysis

Multi-Criteria Decision Analysis (MCDA) is a common problem encountered across various fields of research, where multiple conflicting objectives must be optimized simultaneously. In such scenarios, improving one objective often leads to the deterioration of another, highlighting the inherent trade-offs between different performance criteria. The core focus of MCDA is on selecting the most suitable configuration from a set of optimal candidates, taking into account the interrelated nature of the objectives and their trade-offs.[3] This process requires a careful balance to ensure that the selected solution aligns with the overall goals and priorities of the decision-making context.

2.1.1 Weighted Sum

To address the challenge of optimizing multiple conflicting objectives, one common approach is the weighted sum method [4]. This classic and straightforward technique remains highly effective due to its intuitive implementation and its ability to clearly illustrate trade-offs among conflicting objectives. By assigning a set of weights to each objective function(shown in equation 1), the method transforms a multi-objective problem into a single-objective optimization problem, combining multiple objectives into a weighted sum.

The weighted sum method can be mathematically represented as:

$$S = \sum_{i=1}^{n} w_i \cdot f_i(x), \tag{1}$$

where S is the aggregated score, w_i represents the weight assigned to the i-th objective, $f_i(x)$ denotes the objective function, and n is the total number of objectives.

A critical aspect of the weighted sum method is the assignment of weights to each objective. However, many existing methods (representative examples in Table 1) lack sufficient objectivity, even when incorporating objectivity-oriented weighting techniques. The choice of weights significantly impacts the outcome, and therefore, it is essential to carefully determine these weights to ensure a balanced and fair representation of all objectives.

Table 1: Weighted Sum Methods in MCDA

Method	**Subjective**	**Objective**	**Related Research**
AHP	✓		[5], [6]
Entropy method		✓	[7]
TOPSIS	✓	✓	[7], [8]

2.1.2 Clustering Analysis

Cluster analysis[9] is an effective tool for addressing Multi-Criteria Decision Analysis (MCDA) problems[10], as it enables the identification of global features within a dataset by analyzing the distribution of data points. By grouping similar data points together, clustering methods can reveal inherent patterns and structures that may not be immediately apparent.

For example, Nolzen's work on ESCAPE 34 [11] demonstrates the use of clustering methods in energy transition scenarios. This study applies K-means clustering[12] and decision trees to analyze 441 selected policies, aiming to identify the most critical policies in energy transition contexts. The identified key policies are then evaluated using a low-regret model to determine their effectiveness. While this approach relies solely on the intrinsic data characteristics, it also presents the risk of overfitting, as it may capture noise or peculiarities specific to the sample rather than generalizable patterns.

Although clustering provides valuable insights into the overall structure of the data, it is essential to carefully consider these potential limitations and complement cluster analysis with additional methods to ensure robust and reliable results.

2.2 Multi-Objective Optimization

In Multi-Objective Optimization(MOO) Problems, there are already several methods to generate the optimal solution candidates, including Genetic Algorithm, Reinforcement Learning

2.2.1 Genetic Algorithm

To tackle the complexity of multi-objective optimization problems, the Genetic Algorithm (GA) and its variants, such as NSGA-II (Non-dominated Sorting Genetic Algorithm II)[13], SPEA2 (Strength Pareto Evolutionary Algorithm 2)[14], and MOEA/D (Multi-Objective Evolutionary Algorithm based on Decomposition)[15], offer powerful and flexible approaches. These algorithms are inspired by natural evolution and employ mechanisms such as selection, crossover, and mutation to efficiently explore the solution space. Their adaptability and ability to maintain a diverse set of potential solutions make them highly effective for solving complex problems involving conflicting objectives.

One significant challenge when applying these evolutionary algorithms is parameter sensitivity. The performance of algorithms like NSGA-II, SPEA2, and MOEA/D heavily depends on key parameters such as population size, crossover rate, mutation rate, and specific settings like NSGA-II's crowding distance or MOEA/D's weight vector decomposition. Improper parameter settings can lead to issues such as premature convergence to local optima, reduced solution diversity, or inefficient search space exploration. Moreover, this sensitivity can cause variability n solution quality across different runs, leading to inconsistent and less reliable outcomes.

2.2.2 Reinforcement Learning

Reinforcement Learning (RL) is an adaptive approach that is increasingly being applied to solve multi-objective optimization problems. Unlike traditional optimization methods, RL is based on learning through interaction with an environment, where an agent takes actions to maximize cumulative rewards over time. In the context of multi-objective optimization, RL can be employed to learn a policy that balances multiple conflicting objectives by exploring and exploiting different trade-offs within the solution space[16]. And some research like [17] enable agents to learn the a set of Pareto dominating policies to converge to multiple objectives.

A key strength of RL in multi-objective settings is its ability to handle dynamic and complex environments where the objectives may change over time or have dependencies on each other[18]. By utilizing techniques like multi-objective Q-learning, deep reinforcement learning, or Pareto-based reward shaping, RL algorithms can learn to prioritize different objectives based on their relative importance or adapt to new preferences in real-time. This flexibility makes RL particularly suitable for applications where the objectives are not static or clearly defined from the start.

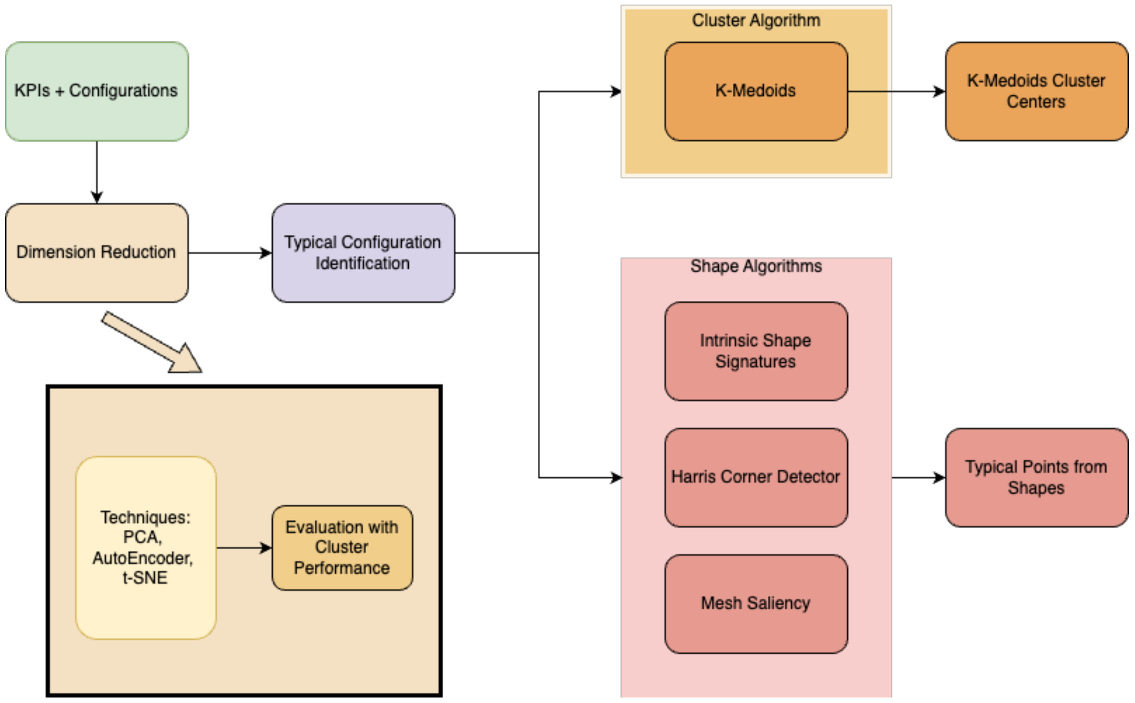
However, applying RL to multi-objective optimization also poses challenges, such as defining a suitable reward function that captures the trade-offs among objectives and ensuring sufficient exploration of the solution space. Moreover, RL algorithms are often sensitive to their own set of parameters, such as learning rates, exploration rates, and reward discount factors, which can significantly impact their performance and convergence speed. Therefore, careful design and tuning of these parameters, along with advanced strategies like meta-learning or transfer learning, are crucial for leveraging RL effectively in multi-objective scenarios.

2.3 Machine Learning for Control in Energy System

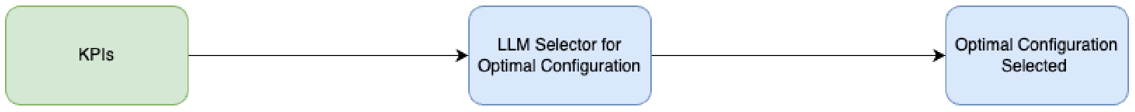
In recent years, considerable research has focused on applying machine learning, particularly reinforcement learning (RL), to control theory. Specifically, within the domain of energy systems, Zhang [19] utilized reinforcement learning to simulate the RC model for heat transfer in scenarios involving multiple chambers. Some related studies have explored the potential of RL agents to override complex principles and physical laws to optimize control strategies. Meanwhile, Lin etc. [20] implements deep reinforcement learning for the control in Organic Rankine Cycle (ORC). However, despite these advances, the broader application of RL in control remains limited due to several challenges. These include high computational and data requirements, convergence and stability issues, and the need for safe and reliable control in critical environments. Additionally, the lack of standardized tools and solutions, the "black-box" nature of many RL algorithms, and the existing dominance of mature traditional control methods further restrict RL's widespread use in the energy sector and beyond. Future research may focus on addressing these challenges to enable more effective integration of RL into energy system control.

3 Methodology

To identify the most representative configurations within our dataset, we initially use clustering analysis as a reference point. While clustering provides insights into the overall distribution and grouping of the data, it comes with inherent limitations, such as sensitivity to the data's shape and distribution, dependence on initial parameter settings, and challenges in determining the optimal number of clusters. Given these limitations, we consider clustering results as a preliminary benchmark rather than a definitive solution.


KPIs + Configurations
Dimension Reduction
Typical Configuration Identification
Cluster Algorithm
K-Medoids
K-Medoids Cluster Centers
Shape Algorithms
Intrinsic Shape Signatures
Harris Corner Detector
Mesh Saliency
Typical Points from Shapes
Techniques: PCA, AutoEncoder, t-SNE
Evaluation with Cluster Performance

(a) Typical Configuration Identification


KPIs
LLM Selector for Optimal Configuration
Optimal Configuration Selected

(b) LLM aided Optimal Configuration Selection

Figure 2: Typical and Optimal Configuration Identification Overlook

To address the high dimensionality of our dataset, we employ a hybrid, multi-step dimensionality reduction approach. By combining multiple techniques, we ensure that the reduced dataset retains as many critical features as possible. This phased reduction process helps manage complexity while preserving essential patterns within the data, providing a solid foundation for further analysis.

After reducing the data to three dimensions, we apply geometric algorithms in 3D space to accurately identify typical configurations.(shown in Figure 2a) Unlike traditional clustering methods, these shape-based techniques capture a broader range of features and patterns, offering a deeper and more nuanced understanding of the data structure. By leveraging 3D geometric analysis, we overcome the limitations of clustering and achieve a more comprehensive representation of the data.

In addition, we propose a method using a Large Language Model (LLM) to perform the Weighted Sum Method automatically shown in Figure 2b, enabling it to handle natural language inputs that express preferences for specific aspects.

By integrating LLM capabilities with geometric analysis, we further enhance our ability to derive meaningful insights on typical and optimal configurations from complex datasets.

3.1 Dimension Reduction

To effectively handle the high dimensionality of the dataset, dimensionality reduction techniques are employed as a preliminary step in the analysis. The goal of this process is to reduce the complexity of the data while preserving its essential structure and features, thereby facilitating the subsequent identification of typical configurations in a lower-dimensional space.

There are several dimensionality reduction methods to consider, including linear approaches such as Principal Component Analysis (PCA)[21] and non-linear techniques like t-Distributed Stochastic Neighbor Embedding (t-SNE)[22] and AutoEncoder(AE)[23]. Each method has its own advantages and is suited to different types of data structures:

3.1.1 PCA

Principal Component Analysis (PCA): PCA is a widely used linear technique that transforms the original high-dimensional data into a lower-dimensional space by identifying the directions (principal components) that maximize the variance in the data. This method is particularly effective for datasets where the underlying structure can be captured through linear combinations of the original features.

3.1.2 AutoEncoder

AutoEncoder: For non-linear dimensionality reduction, an AutoEncoder, a type of artificial neural network, will be utilized. AutoEncoders learn efficient representations of the data through an encoding-decoding process, capturing complex, non-linear relationships that PCA might not fully represent. By training an AutoEncoder to minimize the reconstruction error, we aim to reduce the data to three dimensions, preserving important structural information that is not necessarily captured by linear methods.

3.1.3 Combiner

In this study, we propose a two-stage dimensionality reduction approach combining an AutoEncoder and PCA to enhance clustering performance on high-dimensional datasets. The Autoencoder captures the complex, nonlinear structures of the data by reducing the input dimensionality to a 32-dimensional latent space. Subsequently, PCA is applied to further reduce this latent representation to 3 dimensions, effectively preserving the most significant linear features while eliminating redundant information. Our experimental results(in Section 4) demonstrate that this combined method outperforms traditional dimensionality reduction techniques, such as PCA alone or t-SNE, in terms of Davies-Bouldin Score and Silhouette Score, across various datasets. The proposed approach leverages the strengths of both nonlinear and linear dimensionality reduction methods, providing a robust framework for clustering tasks in complex, high-dimensional environments.

3.2 Clustering

After dimension reduction, the clustering methods are deployed to test the performance of the dimension reduction, the most popular clustering method is the K-means, and we used it for the validation of the proposed dimension reduction technique, Meanwhile, K-Medoids method[24] shares the same computation method with the K-means and the only difference is that the K-Medoids take the real data point as the data center, while the K-Means is a virtual computed one.

Since the center of the cluster is a real data point, K-Medoids is selected in this study.

3.3 Typical Solution Identification

To identify the most typical solutions within our dataset, we initially consider using the cluster centers derived from the K-Medoids method. However, this approach has its limitations. The cluster centers can be heavily influenced by the internal distribution of data points within each cluster, which may not accurately reflect the most representative features of the overall dataset. Therefore, a more precise and effective approach is needed to identify typical configurations.

Since we have transformed the dataset into a three-dimensional space, we can leverage graphical and geometric methods to uncover features that are not immediately apparent in the original data. These methods allow us to capture and analyze more subtle patterns and characteristics.

To achieve this, we employ three advanced techniques from graphics and computer vision: Intrinsic Shape Signatures[25], Harris Corner Detection[26], and Mesh Saliency[27]. Each of these methods offers a unique perspective on identifying typical configurations, providing a clearer and more comprehensive understanding of the dataset's structure to identify the typical configurations.

3.3.1 Intrinsic Shape Signatures

Intrinsic Shape Signatures (ISS) is an algorithm designed to identify similarities between shapes by detecting key points that can represent the overall shape. It is particularly effective in identifying stable and distinctive points by analyzing local geometric structures within the data.

To identify typical configurations within the reduced-dimensional data, we employed the ISS algorithm. This keypoint detection method focuses on locating points that exhibit significant variations in local shape properties. Mathematically, the ISS algorithm computes the covariance matrix $\mathbf{C}$ for a point p and its neighborhood $N(p)$ as follows:

$$\mathbf{C} = \frac{1}{|N(p)|} \sum_{q \in N(p)} (q - \bar{p})(q - \bar{p})^T, \tag{2}$$

where $\bar{p}$ is the centroid of the neighborhood, calculated as:

$$\bar{p} = \frac{1}{|N(p)|} \sum_{q \in N(p)} q. \tag{3}$$

The covariance matrix $\mathbf{C}$ is then subjected to eigenvalue decomposition, yielding eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$ and their corresponding eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, which form the Local Reference Frame (LRF) axes. By examining the eigenvalues, the ISS algorithm detects regions that represent typical configurations with high geometric distinctiveness. Originally, ISS was developed to identify shapes from different viewpoints by extracting key points that characterize the overall shape. In our application, we utilize only the initial phase of the ISS process to extract these key points. The rationale is that if the key points can be used to identify the same shape from multiple perspectives, they can also effectively represent the entire shape — in our case, the set of optimal configurations. This approach enables us to capture the essential structural characteristics of the data, leading to a more accurate representation of the most informative patterns in the lower-dimensional space.

3.3.2 Harris Corner Detection

Complementing the ISS method, we applied an extended 3D Harris Corner Detector to further identify typical configurations within the reduced-dimensional data. The 3D Harris Corner Detector is an adaptation of the classic 2D corner detection algorithm, specifically designed to detect points of interest in 3D data by locating regions with significant curvature variations.

Mathematically, the Harris Corner Detector identifies key points by analyzing the eigenvalues of the local covariance matrix $\mathbf{C}$, which is defined similarly to Equation 2. The detector evaluates the response function R as:

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2, \tag{4}$$

where λ_1 and λ_2 are the eigenvalues of the covariance matrix $\mathbf{C}$, and k is an empirical constant typically set between 0.04 and 0.06. A large positive value of R indicates a corner or an area of high curvature, which signifies a point of interest.

The 3D Harris Corner Detector is particularly effective at finding stable and prominent configurations that capture critical geometric features of the data after dimension reduction. By robustly detecting these points, the method provides deeper insights into the underlying patterns and structural characteristics of the dataset.

3.3.3 Mesh Saliency

To further enhance our ability to identify typical configurations, we employed the Mesh Saliency method, which is specifically suited for data represented as 3D meshes. Mesh Saliency assesses the significance of different regions on a 3D surface by calculating the saliency of each vertex based on local geometric properties, such as curvature and surface variation. Unlike point-based methods, Mesh Saliency leverages the connectivity information within the mesh to detect regions that stand out as both visually and geometrically significant.

Mathematically, the saliency $S(v)$ of a vertex v is computed by evaluating the local surface variations over multiple scales:

$$S(v) = \sum_{i=1}^{n} w_i \left| \kappa_i(v) - \bar{\kappa}(v) \right|, \tag{5}$$

where $\kappa_i(v)$ represents the curvature at vertex v at scale i, $\bar{\kappa}(v)$ is the average curvature over all scales, and w_i is the weight assigned to each scale. This approach enables the detection of regions that are both visually prominent and geometrically significant.

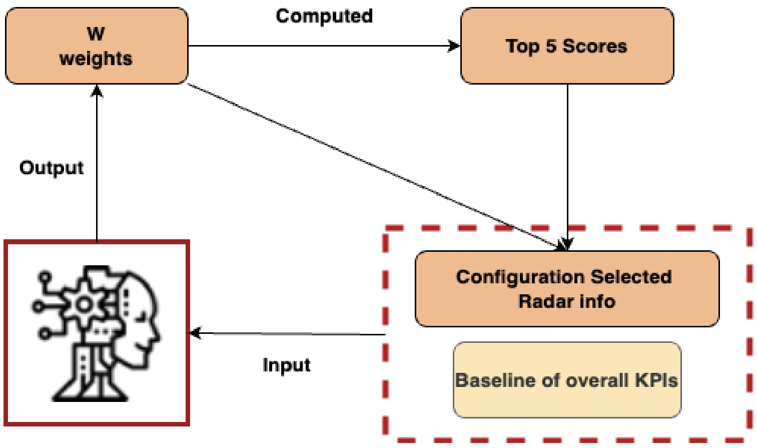
This technique allows us to identify typical configurations by focusing on the most salient regions of the mesh data, offering a comprehensive understanding of the structural and topological features of the reduced-dimensional data. Historically, this algorithm has been utilized for selecting informative viewpoints[28], making it particularly well-suited for tasks where identifying key regions is crucial.

Given the presence of numerous repeated or closely positioned configurations, the methods mentioned above often select different configurations in each experiment. However, despite these variations, the selected configurations remain valid and insightful. To achieve a more reliable identification of typical solutions, we conducted multiple experiments and analyzed the frequency patterns of the selected configurations, as shown in Section 4. From this analysis, we identified the top five most frequently occurring configurations as the most representative, ensuring a robust and comprehensive understanding of the data's key structural features.

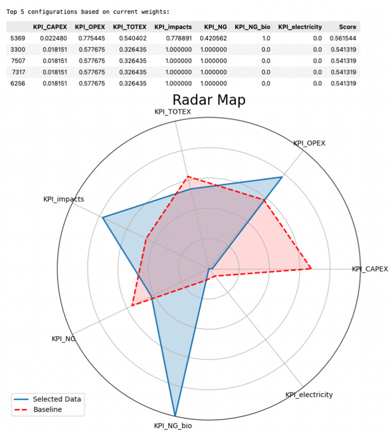
3.4 LLM Aided Decision Making

Large Language Models (LLMs) have revolutionized various fields, particularly following the introduction of the GPT series [29][30]. These models show significant potential in Multi-Criteria Decision Analysis (MCDA) problems, especially when there are preferences for certain aspects and the goal is to find optimal configurations. For instance, an LLM can understand text-specific preferences like "I prefer low environmental impacts." However, due to limitations in the length of information that LLMs can process, we only take Key Performance Indicators (KPIs) into consideration. Despite this focus, when dealing with the extensive array of KPIs typically found in real-world scenarios, LLMs often struggle with effective reasoning due to the overwhelming volume and complexity of the data. This challenge is particularly evident when the LLM attempts to analyze multiple criteria simultaneously.

The recent integration of code interpreter capabilities into LLMs [31], such as those seen in GPT-4 [32], has enabled these models to execute code in Python and automate certain analytical tasks. However, they often default to the Weighted Sum Method, which assigns arbitrary weights to Key Performance Indicators (KPIs) to compute scores and identify the top ones. This approach lacks the nuance and intelligence required for more sophisticated decision-making.



(a) Iterative interaction architecture



Top 5 configurations based on current weights:

	KPI_CAPEX	KPI_OPEX	KPI_TOTEX	KPI_impacts	KPI_NG	KPI_NG_bio	KPI_electricity	Score
5369	0.022480	0.775445	0.540402	0.778891	0.420562	1.0	0.0	0.561544
3300	0.018151	0.577675	0.326435	1.000000	1.000000	0.0	0.0	0.541319
7507	0.018151	0.577675	0.326435	1.000000	1.000000	0.0	0.0	0.541319
7317	0.018151	0.577675	0.326435	1.000000	1.000000	0.0	0.0	0.541319
6256	0.018151	0.577675	0.326435	1.000000	1.000000	0.0	0.0	0.541319

(b) Radar Map for iteration

Figure 3: LLM aided Decision Making Progress

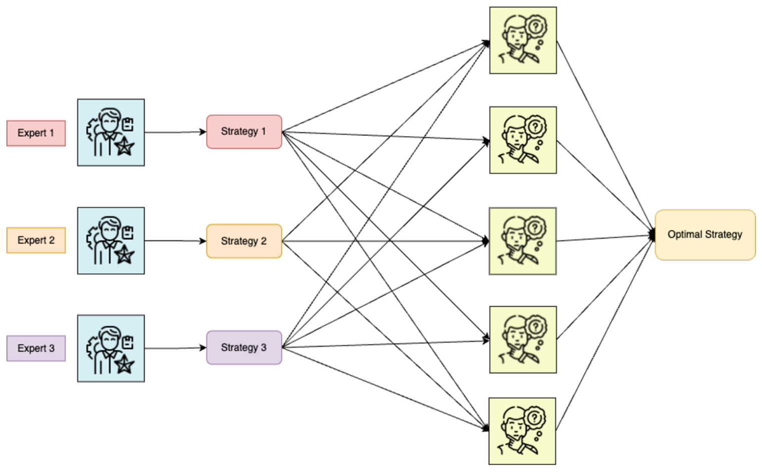


Figure 4: Prompt Architecture with Self-Evaluation

To overcome these limitations, we have developed an innovative iterative self-update mechanism that allows the LLM to dynamically refine its weighting strategy based on feedback from previous iterations. Our approach, illustrated in Figure 3a, leverages a radar map that visually summarizes the means or medians of the entire dataset alongside those of the top five configurations identified by the LLM.

This radar map, shown in Figure 3b, serves as a feedback loop, providing critical information to guide the LLM's subsequent decision-making process. By using the radar map data from each iteration, the LLM can learn and adjust its parameters more effectively, moving toward a more accurate and insightful analysis with each cycle. This iterative process not only mitigates the problem of information overload but also enhances the LLM's ability to make more intelligent, data-driven decisions over time.

To mathematically formalize this, we refer back to the weighted sum formula 1 and replace $f_i(x)$ with KPI_i:

$$S = \sum_{i=1}^{n} w_i \cdot \text{KPI}_i, \tag{6}$$

where S is the aggregated score, w_i represents the weight assigned to each KPI, and KPI_i denotes the i-th Key Performance Indicator. By refining the weights w_i iteratively using feedback from the radar map, our method enables the LLM to achieve a more balanced and effective decision-making process.

To further enhance the reasoning capabilities of LLMs, we drew inspiration from recent advances in prompt engineering that focus on the structured organization of thoughts, as demonstrated in works such as Chain-of-Thought (CoT) prompting [33] and Tree-of-Thought (ToT) prompting [34]. Building upon these concepts, we proposed a novel architecture, shown in Figure 4, designed to improve the reasoning performance of LLMs.

In our approach, we define three distinct experts, each responsible for proposing a different selection strategy. These strategies are then evaluated through a voting mechanism conducted by five additional experts. After the voting process, the strategy receiving the most votes is selected. This chosen strategy is then used to generate a new set of parameters for the Weighted Sum Method (WSM), which are formatted as inputs for further analysis.

This architecture enables the LLM to dynamically refine its decision-making process by integrating multiple perspectives and leveraging a structured evaluation mechanism, thereby enhancing its reasoning ability in complex scenarios.

3.5 RL for Real-Time Multi-Objective Control

In addition to the typical configuration selection, we propose a method based on Reinforcement Learning (RL) that can achieve real-time control of configurations under uncertain market conditions, rather than relying solely on pre-computed simulations for fixed market scenarios.

Our approach extends the Deep Deterministic Policy Gradient (DDPG) framework by incorporating a shared layer and an individual layer within the neural network architecture(shown in Figure 5). The shared layer is designed to handle variables that contribute to the computation of two or more Key Performance Indicators (KPIs), ensuring that these interdependencies are effectively learned and managed. The individual layer, on the other hand, is responsible for processing variables that influence only a single KPI, thereby optimizing the network's ability to focus on specific objectives without interference from other variables.

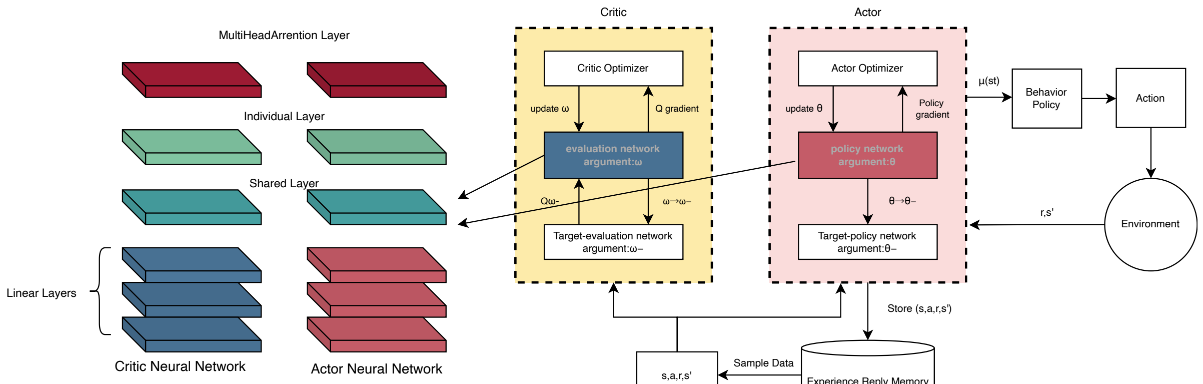


Figure 5: Modified DDPG Reinforcement Learning Framework

To further enhance performance in a multi-objective setting, we introduce a Multi-head Attention Mechanism[35] into the neural network. This mechanism enables the model to effectively attend to different aspects of the input data that are relevant to multiple objectives, thereby improving the convergence rate toward a solution that balances the competing KPIs.

The relationship between KPIs and configurations is modeled as:

$$\text{KPI} = A \times \text{configurations}, \tag{7}$$

where A represents dynamically changing market conditions based on a normal distribution.

By utilizing this reinforcement learning approach, the model dynamically adjusts configurations in response to real-time changes in market conditions, achieving a more adaptive and resilient control strategy. The use of shared and individual layers, combined with the Multi-head Attention Mechanism, facilitates a more nuanced understanding of how different variables affect multiple KPIs, ultimately leading to more effective decision-making in a complex, multi-objective environment.

This method offers several advantages over traditional simulation-based approaches. Firstly, it reduces the reliance on pre-defined market conditions, allowing for more flexible and responsive strategies. Secondly, it ensures that the control system is continuously learning and adapting, which is critical in environments where conditions can change rapidly and unpredictably. Finally, the ability to manage multiple objectives simultaneously with greater efficiency makes this approach particularly valuable for applications in dynamic markets.

By implementing these innovations in the DDPG framework, we demonstrate how reinforcement learning can be effectively employed for real-time, multi-objective control, providing a powerful tool for managing complex systems in uncertain and evolving conditions.

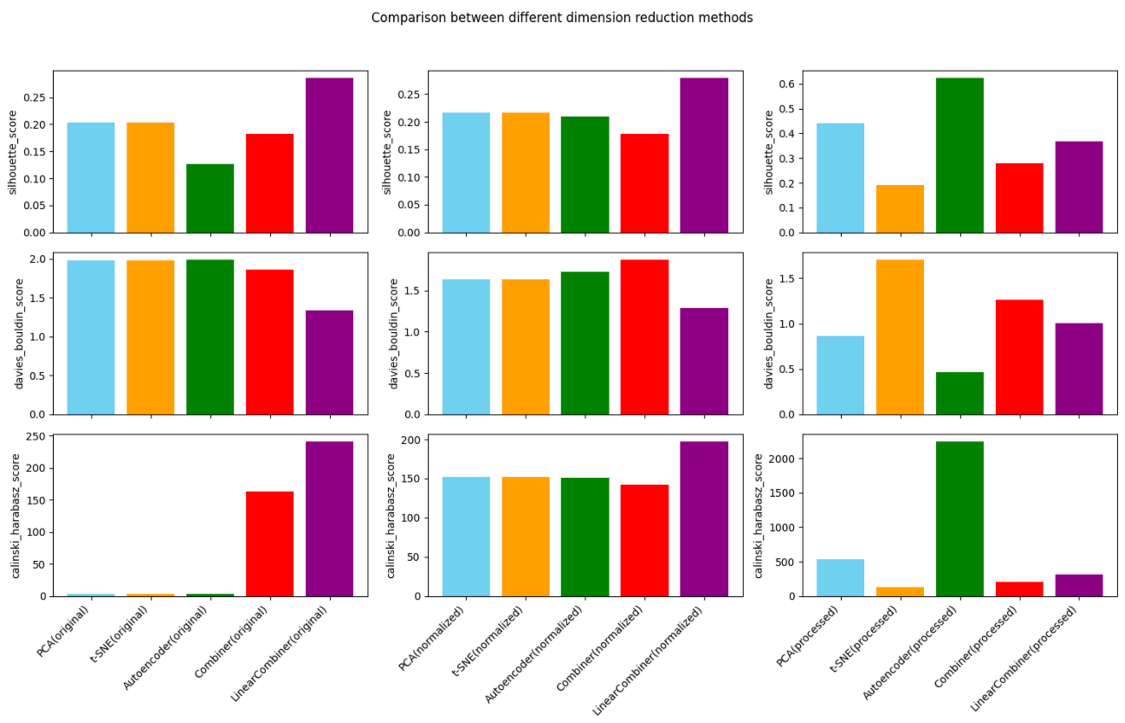
4 Experimental Results and Validation

4.1 Dimension Reduction and Clustering

There are three experimental cases with different sizes of configuration candidates: 500, 8000, and 16,000 configurations. The proposed methodology for dimensionality reduction is validated using clustering results, as directly assessing feature retention after dimensionality reduction is challenging. By examining the clustering outcomes in the reduced-dimensional space and mapping them back to the high-dimensional data, we can evaluate whether the essential features have been effectively preserved. This evaluation is performed using three metrics described in Appendix A.5: the Davies-Bouldin Index (DB) [36], Silhouette Coefficient [37], and Calinski-Harabasz Index (CH) [38].

The clustering performance is assessed using these metrics, each offering a different perspective on cluster quality. A lower DB score indicates better clustering, reflecting higher intra-cluster similarity and greater inter-cluster separation. The Silhouette Coefficient, ranging from -1 to 1, measures how well each data point is matched to its own cluster versus other clusters; values closer to 1 suggest superior clustering quality. Similarly, a higher CH index signifies better clustering performance, indicating that the clusters are more compact and well-separated.

Table 2: Experiments in 500 configuration datasets



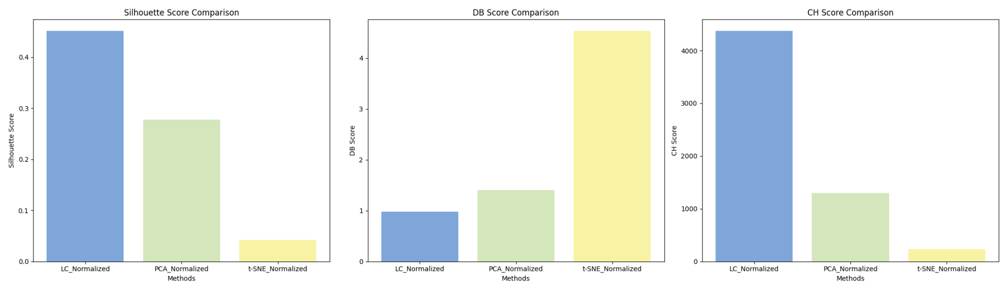
Normalized data is the data normalized from the original dataset before dimension reduction, processed data are the dataset after dimension reduction

If the performance is good in the processed data but poor in the normalized data, could be called as over-fitting.

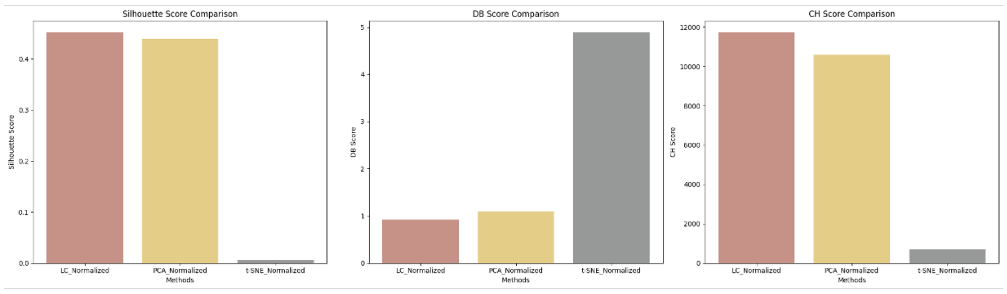
We propose two combiners for dimensionality reduction: the Linear Combiner and the General

Table 3: Experiments in different number of configuration datasets

(a) Experiments in 8000 configuration datasets



(b) Experiments in 16000 configuration datasets



Combiner. The Linear Combiner involves two stages—first using an AutoEncoder, followed by PCA—while the General Combiner adds a third stage with t-SNE.

In our experiment with 500 configurations (Table 2), we observe that dimensionality reduction using the AutoEncoder results in the best clustering performance. However, when the cluster labels are mapped back to the normalized data, they fail to align, indicating overfitting. Due to this overfitting, the AutoEncoder approach will not be utilized in the following experiments.

Conversely, the Linear Combiner consistently performs well across all data types, effectively preserving essential features during the reduction from high-dimensional space to three dimensions. This performance suggests that the Linear Combiner is better suited for our data, retaining key structural characteristics while avoiding overfitting.

Given that datasets in the energy system domain primarily consist of linear features, the General Combiner, which includes t-SNE, will not be considered for further experiments. Instead, PCA and t-SNE will be used as baseline methods in subsequent analyses.

In the experiments of 8000 (see Table 3a)and 16000 (see Table 3b), the Linear Combiner continued to demonstrate the best performance when tracing back to the high-dimensional space. This result confirms that the Linear Combiner retains the most features during dimension reduction.

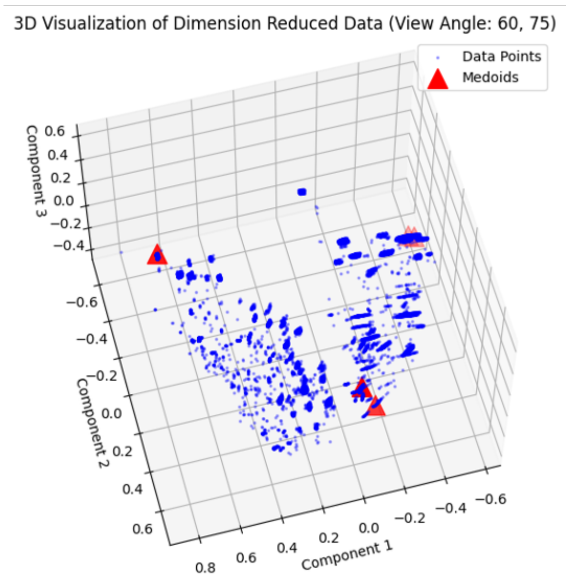
4.2 Typical configuration Identification

In typical configuration identification, the K-Medoids Method could be considered as a baseline, since it is the easiest and direct answer from the cluster distribution of the Datasets.

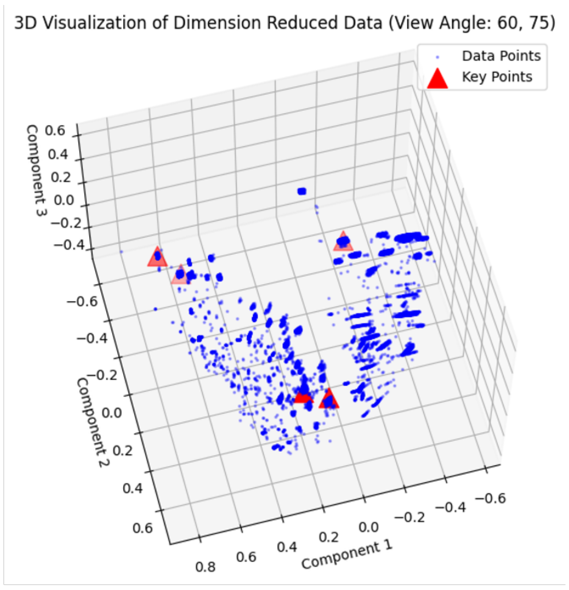
To have a better visualisation of the typical configuration distribution, we visualize them in 3 dimensional space in 3 dimensional datasets which is also the datasets after feature reserved dimension reduction techniques.

4.2.1 K-Medoids Cluster Center

In the K-Medoids method, as shown in Figure 6a, the cluster centers can be influenced by the internal distribution of each cluster, capturing global features while potentially overlooking local features.



(a) Typical Configuration Identified by K-Medoids



(b) Typical Configuration Identified by Shape Algorithms

Figure 6: Comparison of Typical Configurations Identified by Different Methods

4.2.2 Shapes Algorithms

In shape analysis, There are 3 methods, ISS, Harris Corner Detector and Mesh Saliency. The former 2 methods are based one the data point cloud, and the last one is based one the mesh, we performed the Poisson Reconstruction to construct the mesh of the dataset.

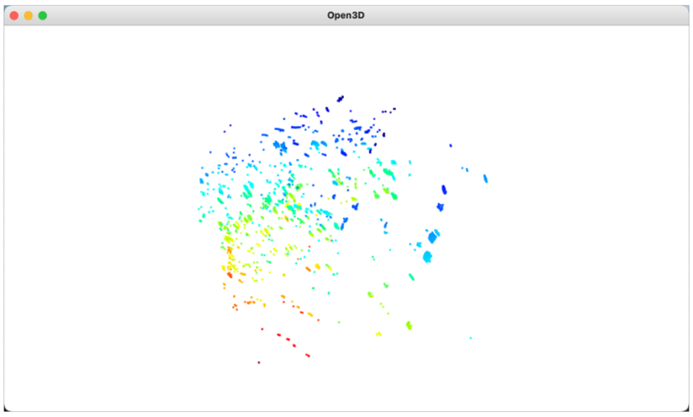
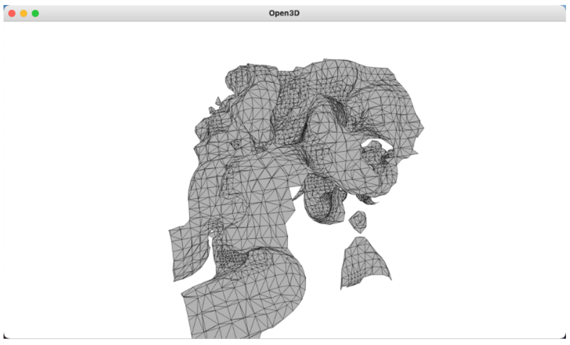
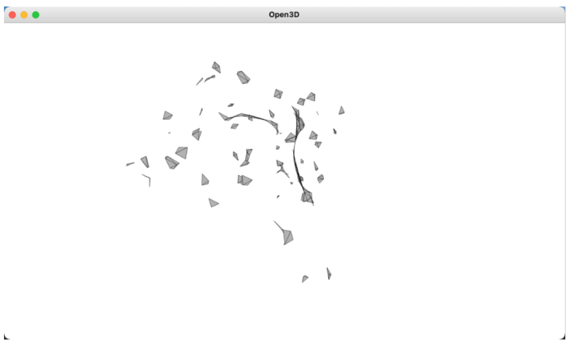


Figure 7: Data in 3D space as point cloud

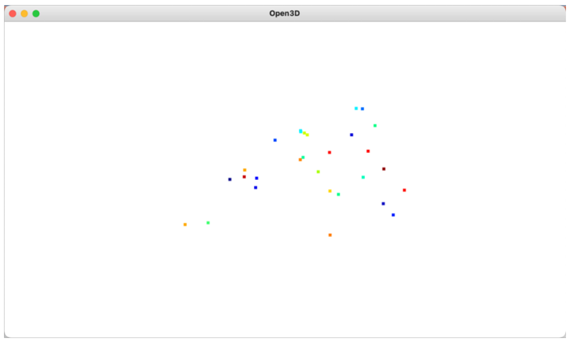
All of the three methods need to compute the normal vectors of the datasets.



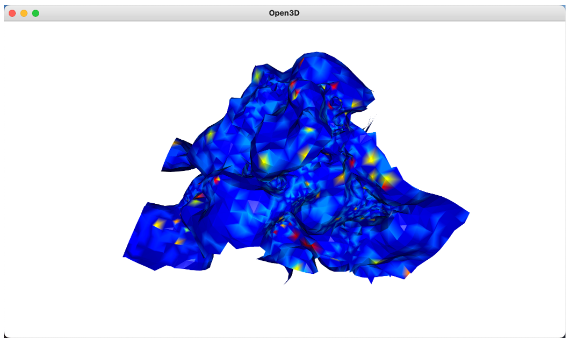
(a) Mesh with Poisson reconstruction



(b) Typical Configuration by ISS



(c) Typical Configuration by Harris



(d) Typical Configuration by Mesh Saliency

Figure 8: Comparison of Different Techniques for Identifying Typical Configurations

In the results, we first visualize the data as a point cloud in 3D space (see Figure 7). To convert this data representation into a mesh, we apply the Poisson Surface Reconstruction technique [39], as shown in Figure 8a.

Using the ISS method, we identify typical configurations within the dataset (Figure 8b). Similarly, the Harris Corner Detector is employed to detect key configurations, with the results depicted in Figure 8c.

In the Mesh Saliency analysis, we compute the saliency values of the data points and visualize the mesh using a color gradient from blue to red (see Figure 8d). Regions closer to red indicate higher degrees of prominence or significance, while regions closer to blue represent areas of lesser importance or more trivial features.

Taking all three methods into account from various perspectives, after 800 iterations of the experiments, we identified the five most frequently occurring configuration numbers as the most typical ones (as shown in Figure 9). They are mapped back to 3-dimensional spaces with visualizations shown in Figure 6b. In fact, all the numbers listed can be considered typical to some extent.

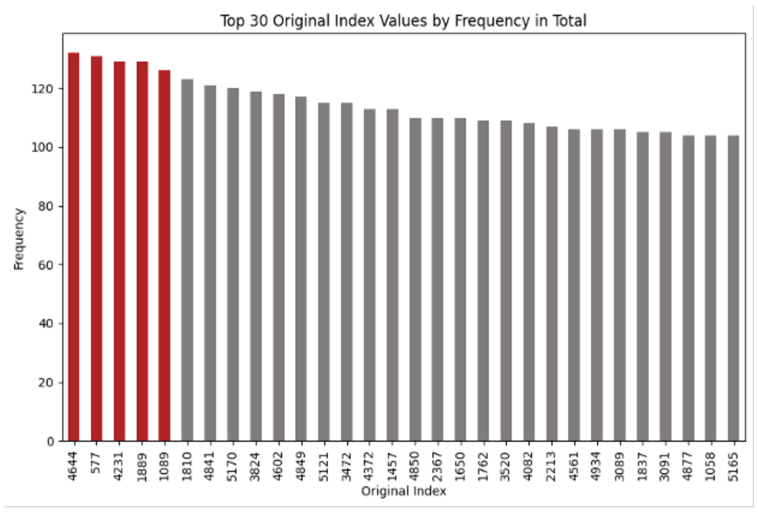


Figure 9: Frequencies for Typical Configurations Identified

4.3 LLM aided Decision Making

Within the LLM component, we initially explored several strategies, including an approach where the LLM automatically defined acceptable percentage intervals iteratively to converge to a final set of five or a few remaining configurations. However, due to inherent limitations in memory capacity and reasoning capabilities, as well as the presence of numerous similar or nearly identical configurations, the LLM frequently became trapped in a repetitive loop and failed to achieve convergence.

Conversely, employing the LLM-embedded Weighted Sum method with a radar map as an intermediary effectively mitigated the constraints imposed by limited context size. Moreover, utilizing a carefully designed prompt structure enhanced the model's reasoning capabilities, ensuring more robust performance.

In the domain of prompt engineering, we systematically experimented with varying the number of experts (see Table 4) assigned to propose strategies and vote, refining our methodology through multiple iterations. Due to the text length limitations in each message, GPT-4 can effectively handle up to three experts in the expected output format. When the number of experts exceeds three, the output often lacks a coherent thought process; instead, the LLM agent in the GPT playground defaults to using the code interpreter, assigning random weights—even when preferences are provided—rendering the decision process non-traceable.

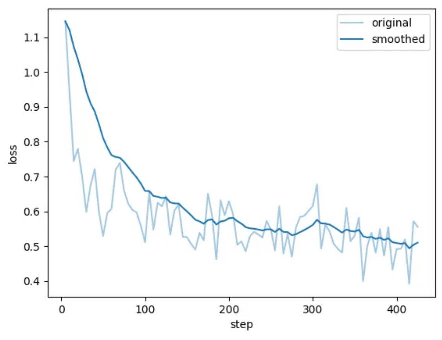
When the number of voting experts is set to three, the architecture resembles a Self-Consistency framework [40]. However, this can result in all three experts consistently voting for a single strategy, effectively endorsing their own suggestions. In cases requiring more complex reasoning—unlike the straightforward decisions typical in games like "24"—this setup can limit diversity in decision-making, as the experts tend to vote for their own strategies.

By increasing the number of experts to five, the possible voting combinations expand to [2, 2, 1] or [3, 1, 1]. In the [3, 1, 1] scenario, the agent executes the strategy with three votes. In the [2, 2, 1] case, the agent combines the two strategies with two votes each. However, if the number of voting experts exceeds five and multiple strategies receive the maximum number of votes, the agent may abandon the reasoning process entirely, defaulting to arbitrary weight assignments and failing to converge in subsequent iterations.

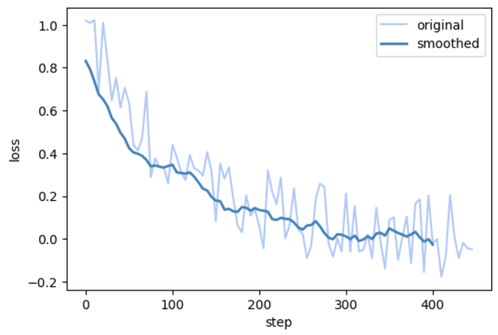
Table 4: Effect of Expert Number on LLM Decision-Making and Convergence

Number of Voting Experts	**Voting Combination**	**Behavior**
3	[1, 1, 1]	Similar to Self-Consistency; no specialized Voting Experts, experts only vote for themselves, resulting in limited reasoning capabilities.
4	[2, 1, 1]	Three experts only vote for themselves and the last one decides everything, leading to a lack of diversity.
5	[3, 1, 1] or [2, 2, 1]	In [3, 1, 1], the strategy with 3 votes is executed. In [2, 2, 1], two strategies with 2 votes each are combined; balances between decision-making diversity and reasoning capabilities.
More than 5	Multiple combinations possible	Increased complexity in decision-making; the agent abandons the reasoning process, defaults to random weight assignments, and fails to converge.

We attempted to fine-tune[41] open-source LLM models, specifically Qwen2-7B [42] and Mistral-7B [43], using the LLaMa-Factory platform [44] to develop a specialized LLM for this task. However, the training process proved to be highly unstable in qwen2 fine tuning progress (shown in Figure 10a), with both experiments (shown in Figure **??**) exhibiting severe oscillations in the loss function. As a result, the fine-tuned models were unable to generate acceptable responses, and the overall output quality was significantly inferior to that of GPT-4. Given these challenges, we decided to utilize the GPT-4 API directly, foregoing further fine-tuning attempts on relatively smaller models.



(a) Qwen2-7B Loss Curve



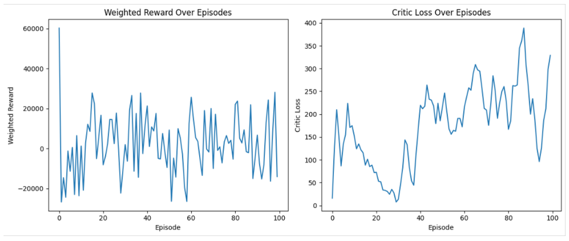
(b) Mistral-7B Loss Curve

Figure 10: Loss Curves for Fine Tuning on Qwen2 and Mistral

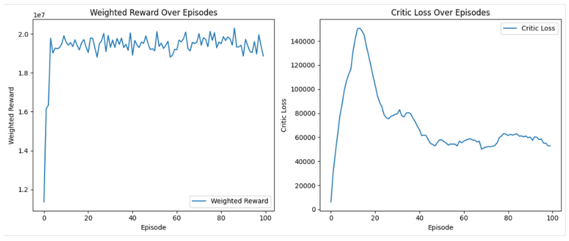
4.4 RL for Real-time Multi-Objective Control

The curves of Weighted Reward and Critic Loss in each experiments shown in Figure11

There are 73 configuration units participate in the computation of 3 KPIs, and 64 of them are components of equal or more than 2 KPIs. We performed two experiments on the Reinforcement learning framework. The first one is only taking the Shared Layer and Individual Layer into consideration. The other one added the Multi-head Attention Mechanism in each neural network to extract more features among the units.



(a) Curves for RL without Multi-head Attention Layer



(b) Curves for RL with Multi-head Attention Layer

Figure 11: Curves for no Multi-head Attention Layer

It is evident that, without the Multi-head Attention Layer, the rewards are highly unstable and predominantly negative. In contrast, when the Multi-head Attention Layer is included, the rewards stabilize after a few episodes, and the Critic Loss converges to a steady state after approximately 30 episodes. This indicates that the agent, trained with the Multi-head Attention Layer, effectively adapts to dynamic market conditions by modifying the configuration units with minimal violations. Furthermore, the Critic Neural Network in the DDPG framework can reliably predict the actions of the Actor Neural Network at a stable and acceptable level.

5 Conclusion

In this paper, we propose a novel method for identifying typical configurations from high-dimensional non-dominated solution sets using dimension reduction techniques, combined with graphics and vision algorithms in three-dimensional space. The stage-wise dimension reduction process preserves as many features as possible while transforming the high-dimensional dataset into three-dimensional space, and the graphics and vision methods are employed to extract location features that are not visible in the raw datasets through a combination of clustering techniques in both high-dimensional and low-dimensional spaces. This approach offers a new perspective in the field of Multi-Criteria Decision Analysis (MCDA) by providing a means to identify the most prominent solutions within the data itself, serving as representative outcomes under uncertain market conditions.

Meanwhile, the integration of a Multi-head Attention Layer within the DDPG framework enhances the adaptability to dynamically changing, multi-objective environments. This suggests that the Attention mechanism can play a unique role in refining the process of multi-objective optimization, complementing our proposed method by focusing on diverse objectives in real-time. Together, these methods provide a comprehensive toolkit for effectively navigating and interpreting complex, high-dimensional datasets, enabling more robust decision-making in dynamic, uncertain environments.

References

[1] J.-J. Wang, Y.-Y. Jing, C.-F. Zhang, and J.-H. Zhao, "Review on multi-criteria decision analysis aid in sustainable energy decision-making," *Renewable and sustainable energy reviews*, vol. 13, no. 9, pp. 2263–2278, 2009.

[2] Y. Zhao, H. Hagi, B. Delahaye, and F. Maréchal, "A holistic approach to refinery decarbonization based on atomic, energy and exergy flow analysis," *Energy*, vol. 296, p. 131 117, 2024.

[3] I. B. Huang, J. Keisler, and I. Linkov, "Multi-criteria decision analysis in environmental sciences: Ten years of applications and trends," *Science of the total environment*, vol. 409, no. 19, pp. 3578–3594, 2011.

[4] R. M. Dawes and B. Corrigan, "Linear models in decision making.," *Psychological bulletin*, vol. 81, no. 2, p. 95, 1974.

[5] K. Sohn, "A systematic decision criterion for the elimination of useless overpasses," *Transportation Research Part A: Policy and Practice*, vol. 42, no. 8, pp. 1043–1055, 2008.

[6] Z. Chourabi, F. Khedher, A. Babay, and M. Cheikhrouhou, "Multi-criteria decision making in workforce choice using ahp, wsm and wpm," *The Journal of The Textile Institute*, vol. 110, no. 7, pp. 1092–1101, 2019.

[7] N. Jain, A. Tomar, and P. K. Jana, "A novel scheme for employee churn problem using multi-attribute decision making approach and machine learning," *Journal of Intelligent Information Systems*, vol. 56, pp. 279–302, 2021.

[8] U. M. Modibbo, M. Hassan, A. Ahmed, and I. Ali, "Multi-criteria decision analysis for pharmaceutical supplier selection problem using fuzzy topsis," *Management Decision*, vol. 60, no. 3, pp. 806–836, 2022.

[9] A. Saxena, M. Prasad, A. Gupta, *et al.*, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, 2017.

[10] P. Meyer and A.-L. Olteanu, "Formalizing and solving the problem of clustering in mcda," *European Journal of Operational Research*, vol. 227, no. 3, pp. 494–502, 2013.

[11] N. Nolzen, A. Lademann, D. Roskosch, *et al.*, "Low-regret decisions for the steam supply in the chemical industry," in *ESCAPE-34 PSE-2024*, 2024.

[12] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Oakland, CA, USA, vol. 1, 1967, pp. 281–297.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[14] E. Zitzler, "Spea2: Improving the strength pareto evolutionary algorithm," 2001.

[15] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[16] C. F. Hayes, R. Rădulescu, E. Bargiacchi, *et al.*, "A practical guide to multi-objective reinforcement learning and planning," *Autonomous Agents and Multi-Agent Systems*, vol. 36, no. 1, p. 26, 2022.

[17] K. Van Moffaert and A. Nowé, "Multi-objective reinforcement learning using sets of pareto dominating policies," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3483–3512, 2014.

[18] C. Liu, X. Xu, and D. Hu, “Multiobjective reinforcement learning: A comprehensive overview,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 3, pp. 385–398, 2015.

[19] C. Zhang, Y. Shi, and Y. Chen, “Bear: Physics-principled building environment for control and reinforcement learning,” in *Proceedings of the 14th ACM International Conference on Future Energy Systems*, 2023, pp. 66–71.

[20] R. Lin, Y. Luo, X. Wu, *et al.*, “Surrogate empowered sim2real transfer of deep reinforcement learning for orc superheat control,” *Applied Energy*, vol. 356, p. 122 310, 2024.

[21] I. T. Jolliffe and J. Cadima, “Principal component analysis: A review and recent developments,” *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20 150 202, 2016.

[22] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.,” *Journal of machine learning research*, vol. 9, no. 11, 2008.

[23] Y. Wang, H. Yao, and S. Zhao, “Auto-encoder based dimensionality reduction,” *Neurocomputing*, vol. 184, pp. 232–242, 2016.

[24] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis.* John Wiley & Sons, 2009.

[25] Y. Zhong, “Intrinsic shape signatures: A shape descriptor for 3d object recognition,” in *2009 IEEE 12th international conference on computer vision workshops, ICCV Workshops*, IEEE, 2009, pp. 689–696.

[26] C. Harris, M. Stephens, *et al.*, “A combined corner and edge detector,” in *Alvey vision conference*, Citeseer, vol. 15, 1988, pp. 10–5244.

[27] C. H. Lee, A. Varshney, and D. W. Jacobs, “Mesh saliency,” in *ACM SIGGRAPH 2005 Papers*, 2005, pp. 659–666.

[28] Y. Kim, A. Varshney, D. W. Jacobs, and F. Guimbretiere, “Mesh saliency and human eye fixations,” *ACM Transactions on Applied Perception (TAP)*, vol. 7, no. 2, pp. 1–13, 2010.

[29] T. B. Brown, “Language models are few-shot learners,” *arXiv preprint arXiv:2005.14165*, 2020.

[30] L. Ouyang, J. Wu, X. Jiang, *et al.*, “Training language models to follow instructions with human feedback,” *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[31] X. Wang, Y. Chen, L. Yuan, *et al.*, “Executable code actions elicit better llm agents,” *arXiv preprint arXiv:2402.01030*, 2024.

[32] J. Achiam, S. Adler, S. Agarwal, *et al.*, “Gpt-4 technical report,” *arXiv preprint arXiv:2303.08774*, 2023.

[33] J. Wei, X. Wang, D. Schuurmans, *et al.*, “Chain-of-thought prompting elicits reasoning in large language models,” *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[34] S. Yao, D. Yu, J. Zhao, *et al.*, “Tree of thoughts: Deliberate problem solving with large language models,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[35] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[36] D. L. Davies and D. W. Bouldin, “A cluster separation measure,” *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.

[37] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.
[38] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.
[39] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, vol. 7, 2006.
[40] X. Wang, J. Wei, D. Schuurmans, *et al.*, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022.
[41] Z. Han, C. Gao, J. Liu, S. Q. Zhang, *et al.*, "Parameter-efficient fine-tuning for large models: A comprehensive survey," *arXiv preprint arXiv:2403.14608*, 2024.
[42] J. Bai, S. Bai, Y. Chu, *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.
[43] A. Q. Jiang, A. Sablayrolles, A. Mensch, *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.
[44] Y. Zheng, R. Zhang, J. Zhang, *et al.*, "Llamafactory: Unified efficient fine-tuning of 100+ language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand: Association for Computational Linguistics, 2024. [Online]. Available: `http://arxiv.org/abs/2403.13372`.
[45] I. Kantor, J.-L. Robineau, H. Bütün, and F. Marechal, "A mixed-integer linear programming formulation for optimizing multi-scale material and energy integration," *Frontiers in Energy Research*, vol. 8, p. 49, 2020.

A Appendix

A.1 Code

The code for this project can be found on GitHub at `https://github.com/MiSFiT5/IPESEinternship`.

A.2 Optimal Configuration Sets Sources

The optimal configuration candidates are generated with the Mixed Integer Linear Programming Method[45], an unlimited solution set generated and all of the solution in the set are components of the Pareto Front in this particular MCDA problem.

In multi-objective optimization, the Pareto Front represents the set of non-dominated solutions, where no single objective can be improved without degrading another. A solution $\mathbf{x}_1$ is said to dominate another solution $\mathbf{x}_2$ if the following conditions are met:

$$\begin{aligned} \forall i \in \{1, \ldots, m\}, &\quad f_i(\mathbf{x}_1) \leq f_i(\mathbf{x}_2) \\ \exists j \in \{1, \ldots, m\}, &\quad f_j(\mathbf{x}_1) < f_j(\mathbf{x}_2) \end{aligned}$$

where m is the number of objectives and $f_i(\mathbf{x})$ denotes the value of the i-th objective function for the solution $\mathbf{x}$.

The Pareto Front P is then defined as the set of all non-dominated solutions:

$$P = \{\mathbf{f}(\mathbf{x}) \in \mathbb{R}^m \mid \nexists \mathbf{x}' \in \mathbb{R}^n, \mathbf{x}' \neq \mathbf{x} \text{ such that } \mathbf{x}' \text{ dominates } \mathbf{x}\}$$

where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ is the vector of objective functions.

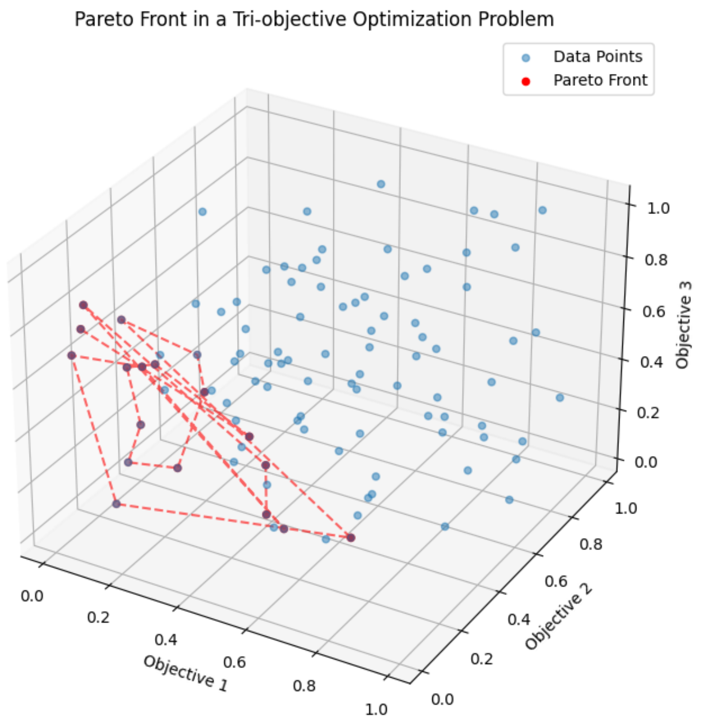


Figure 12: Pareto Front with random data points

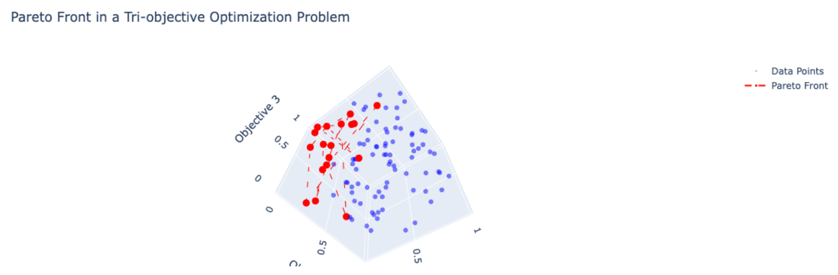


Figure 13: Pareto Front with random data points

A.3 Data Description

The datasets used in the three experiments consist of 500, 8000, and 16,000 configurations, which are transformed into dataframes of sizes 500×1289, 8000×808, and 16000×808, respectively. These dimensions indicate that the dataset for the 500 configurations contains 1289 attributes, while the datasets for the 8000 and 16,000 configurations each have 808 attributes.

Originally, the data were provided as separate CSV files: one file for the Key Performance Indicators (KPIs) of each configuration and another file for the configuration settings. During preprocessing, these files were combined to facilitate subsequent data cleaning and further analysis.

A.4 Data Cleaning

In large-scale industrial systems, data cleaning is essential due to the immense number of dimensions in the original dataset. Our dataset consists of three subsets: one with 500 records and 1296 dimensions, and others with 8000 and 16000 configurations with 808 dimensions. During the data cleaning stage, we initially eliminate all columns where the values are uniform across all entries. These columns do not provide any informational value for subsequent analysis. This process results in a significant reduction in the number of dimensions for both subsets.

By removing these non-informative dimensions, we not only simplify the data structure but also enhance the efficiency of subsequent data processing and analysis. Specifically, for the subset with 500 records, the number of dimensions is significantly reduced from 1296. Similarly, for the subset with 8000 and 16000 configurations, the dimensions are greatly decreased from 808. This step ensures that in our following analyses, we can focus more on meaningful and variable features, thereby improving the accuracy and reliability of our results. The number of dimension after data cleaning is 107 for the 8000 and 16000 experiment sets.

A.5 Metrics For Cluster Performance Evaluations

To evaluate the performance of clustering algorithms, several metrics can be used to assess the quality of the resulting clusters. In this section, we describe three commonly used metrics: the Davies-Bouldin (DB) score, the silhouette coefficient, and the Calinski-Harabasz (CH) index. Each of these metrics provides a different perspective on cluster quality and can help determine the most appropriate clustering method for a given dataset.

A.5.1 Davies-Bouldin (DB) Score

The Davies-Bouldin (DB) score [36] is a metric that evaluates the quality of clustering by examining the average similarity ratio between each cluster and its most similar cluster. The DB score corresponding to the Figure 14 is defined as:

$$\mathrm{DB} = \frac{1}{n}\sum_{i=1}^{n} \max_{i \neq j}\left(\frac{\sigma_i + \sigma_j}{d_{ij}}\right), \tag{8}$$

where n is the total number of clusters, σ_i represents the average distance between each point in cluster i and its centroid, and d_{ij} is the distance between the centroids of clusters i and j. A lower DB score indicates better cluster separation and compactness, as it suggests that the clusters are well separated from each other and have low intra-cluster variance.

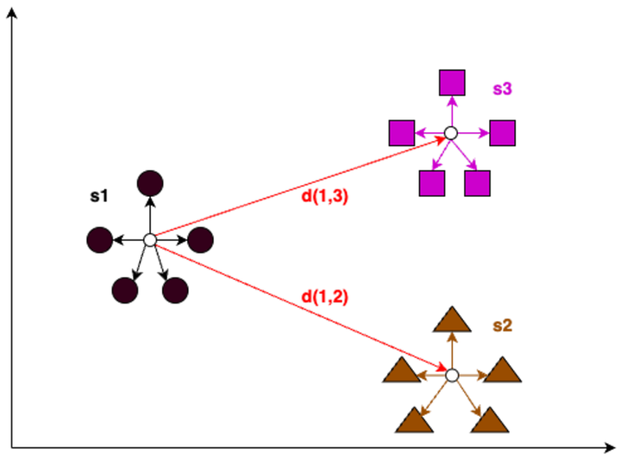


Figure 14: DB Score Computation

A.5.2 Silhouette Coefficient

The silhouette coefficient [37] measures how similar each point in a cluster is to points in its own cluster compared to points in other clusters. The silhouette coefficient for a single point i corresponding to the Figure 15 is calculated as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \tag{9}$$

where $a(i)$ is the average distance between point i and all other points in the same cluster, and $b(i)$ is the minimum average distance between point i and all points in the nearest cluster. The silhouette coefficient ranges from -1 to 1, where a value close to 1 indicates that the point is well matched to its own cluster and poorly matched to neighboring clusters, a value close to 0 indicates that the point is on or very close to the decision boundary between two clusters, and a negative value indicates that the point might have been assigned to the wrong cluster.

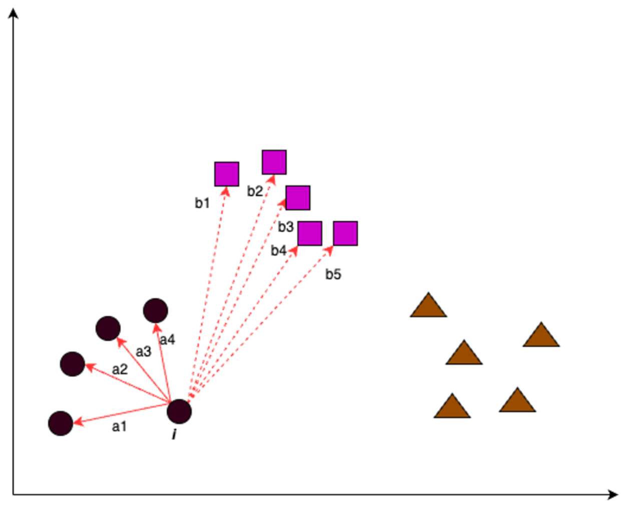


Figure 15: Silhouette Score Computation

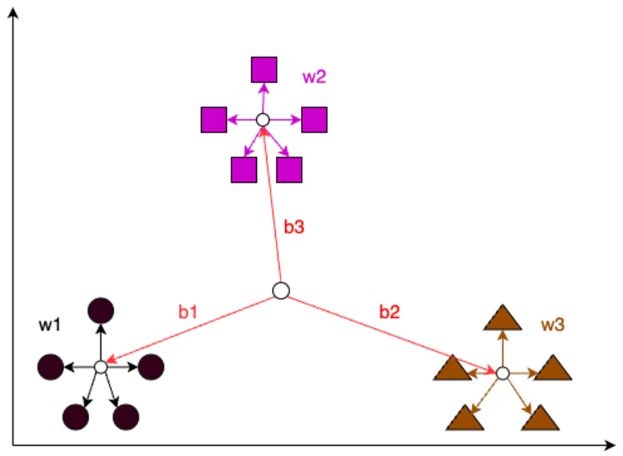


Figure 16: Calinski-Harabasz Index Computation

A.5.3 Calinski-Harabasz (CH) Index

The Calinski-Harabasz (CH) index [38], also known as the variance ratio criterion, measures the ratio of the sum of between-cluster dispersion to the sum of within-cluster dispersion for all clusters. The CH index corresponding to the Figure 16 is defined as:

$$\mathrm{CH} = \frac{\mathrm{Tr}(B_k)}{\mathrm{Tr}(W_k)} \times \frac{n-k}{k-1}, \tag{10}$$

where $\mathrm{Tr}(B_k)$ is the trace of the between-cluster dispersion matrix, $\mathrm{Tr}(W_k)$ is the trace of the within-cluster dispersion matrix, n is the total number of data points, and k is the number of clusters. A higher CH index indicates better-defined clusters with higher between-cluster dispersion and lower within-cluster dispersion, suggesting that the clustering solution has distinct, compact clusters.

Algorithm 1 Davies–Bouldin index Computation

```
 1: procedure ComputeDBIndex($clusters$)
 2:     Initialize $DBIndex \leftarrow 0$
 3:     for each cluster $C_i$ in $clusters$ do
 4:         $max\_similarity \leftarrow -\infty$
 5:         for each other cluster $C_j$ in $clusters$ do
 6:             if $i \neq j$ then
 7:                 Compute intra-cluster distance $d_{in} \leftarrow$ average distance within $C_i$
 8:                 Compute inter-cluster distance $d_{out} \leftarrow$ average distance from $C_i$ to $C_j$
 9:                 Compute similarity $similarity \leftarrow (d_{in} + d_{out})/d_{out}$
10:                 $max\_similarity \leftarrow \max(max\_similarity, similarity)$
11:             end if
12:         end for
13:         $DBIndex \leftarrow DBIndex + max\_similarity$
14:     end for
15:     $DBIndex \leftarrow DBIndex/$ number of clusters
16:     return $DBIndex$
17: end procedure
```

Algorithm 2 Silhouette Score Computation

```
 1: procedure ComputeSilhouetteScore($samples, clusters$)
 2:     Initialize $silhouette \leftarrow 0$
 3:     for each sample $s$ in $samples$ do
 4:         Compute $a(s)$: average distance from $s$ to other points in the same cluster
 5:         Compute $b(s)$: average distance from $s$ to points in the nearest cluster
 6:         Compute silhouette coefficient for $s$: $(b(s) - a(s))/\max(a(s), b(s))$
 7:         $silhouette \leftarrow silhouette +$ silhouette coefficient for $s$
 8:     end for
 9:     $silhouette \leftarrow silhouette/$ number of samples
10:     return $silhouette$
11: end procedure
```

Algorithm 3 Calinski-Harabasz Index Computation

```
1: procedure ComputeCHIndex($clusters$)
2:     Compute within-cluster scatter $W \leftarrow$ sum of squared distances from points to centroids
   over all clusters
3:     Compute between-cluster scatter $B \leftarrow$ sum of squared distances between centroids
4:     $CHIndex \leftarrow (B/(k-1))/(W/(n-k))$   ▷ $k$: number of clusters, $n$: total number of samples
5:     return $CHIndex$
6: end procedure
```

A.6 LLM related Environment

A.6.1 Fine Tuning

We perform the fine tuning on the LLaMa-Factory, which is a powerful tool, and the training is established on Google Colab pro, with GPU L4.

A.6.2 LLM aided Decision Making

The LLM embedded in the Weighted Sum Method utilizes APIs from OpenAI, with the model set to 'gpt-4o'. The API management and development are handled using LangChain in Python. The prompts were tested in both the GPT Assistant and the GPT Playground first and then deployed and tested in local environment.