Digital Signal Feature Extraction for Graph-Based Host Classification in VM Placement

[1]Taufik Hidayat, [2]Lukman Medriavin Silalahi, [3]Abdul Hamid
[1]Department of Computer Engineering, Universitas Wiralodra, Indramayu, Indonesia
[2]Department of Electrical Engineering, Universitas Presiden, Cikarang, Indonesia
[3]Faculty of Technical and Vocational Education, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia
[1]thidayat.ft@unwir.ac.id, [2] lukman.silalahi@president.ac.id, [3] abdulhamid@uthm.my

Abstract - This study proposes a host classification methodology based on signal analysis and graph representation as a pre-placement stage for virtual machines (VM) in a cloud computing environment. The Bitbrains dataset is utilized as a source of time-series data representing CPU, memory, disk, and network utilization. Each parameter is modeled as a discrete signal and analyzed in both time and frequency domains. The analysis is conducted using a fixed observation window and frequency-domain transformation to capture workload characteristics across multiple resources. The Fourier transform results indicate the dominance of low-frequency components, suggesting gradual workload variations. Spectral energy is calculated and normalized to identify quantitative differences between host conditions. The results show that the overloaded class contributes 98.9% of the total spectral energy, while the underloaded and balanced classes contribute only 0.7% and 0.4%, respectively. The extracted features are then integrated using a four-node graph model that connects all resource dimensions into a single structure. The aggregated graph score is employed for dynamic percentile-based classification. From a total of 1,239 analyzed hosts, the proposed method classifies 421 hosts as overloaded, 409 as underloaded, and 409 as balanced. These findings demonstrate that spectral characteristics combined with graph integration provide a quantitatively structured and adaptive host segmentation mechanism, where the resulting classification can support VM placement decisions by identifying underloaded, balanced, and overloaded host conditions.

Keywords — cloud computing, graph model, host classification, spectral analysis

I. Introduction

Currently, the development of cloud computing has increased the complexity of resource management in modern data centers. Virtualization allows the consolidation of multiple VMs on a single physical host, thereby improving infrastructure efficiency [1]–[3]. However, the VM Placement (VMP) process becomes a crucial issue because it affects system performance, load stability, and data center energy consumption [1]. Various approaches have been developed to solve the VMP problem. Heuristic methods such as First Fit and Best Fit are widely used due to their simplicity and low computational complexity [4]. Metaheuristic approaches such as Genetic Algorithm and Particle Swarm Optimization have been developed to improve solution quality in large-scale scenarios [5]. In recent years, machine learning and reinforcement learning-based approaches have also been introduced to handle the complex dynamics of cloud workloads [6].

Nevertheless, most VMP methods still use snapshot values of resource utilization such as CPU, memory, storage, and network as the basis for decision-making. However, current VM placement methods mostly use snapshot-based resource utilization or traditional time-series analysis, which don't do a good job of capturing changes in workload in the frequency domain. Also, graph-based models for managing cloud resources often don't use signal-based feature extraction on multi-resource time-series data. Consequently, these methodologies do not provide a structured depiction of dynamic workload behavior across various resource dimensions. So, there is a gap in research when it comes to putting together multi-resource signal processing, time-frequency feature extraction, and graph-based modeling into a single classification framework. Unlike existing approaches that rely on snapshot-based metrics or conventional time-series analysis, the proposed method captures workload dynamics through signal-based modeling in both time and frequency domains. In addition, the integration of signal-derived features into a graph-based representation enables a more structured characterization of multi-resource interactions. To fill this gap, this study suggests a new method that combines digital signal processing with graph representation to allow for adaptive and structured host classification using dynamic percentile-based thresholds.

This approach does not fully represent the temporal dynamics of workloads. Resource utilization in data centers naturally forms time series that contain fluctuation patterns, bursts, and periodic variations [7], [8]. Related research has shown that time-series-based modeling is effective for workload prediction and anomaly detection in cloud systems [9]. In the field of Digital Signal Processing, time-domain and frequency-domain analysis enables the extraction of statistical and spectral characteristics from discrete signals [10], [11]. Techniques such as Fourier transform have been widely used to identify dominant patterns in dynamic systems.

On the other hand, graph-based modeling has been applied to represent the relationships among components in distributed computing systems [12], [13]. Graph representations allow for the abstraction of host structures through nodes and edges that depict the interconnections among resources. However, most graph approaches in the context of VMP still use direct utilization values without going through a feature extraction

stage from the temporal dynamics of resources [14]. Based on these conditions, this study proposes a signal-based feature extraction approach for multi-resource utilization to build graph-based host representations in the classification process prior to virtual machine placement. CPU, memory, storage, and network utilization are modeled as multi-dimensional discrete signals. Time and frequency domain features are extracted to obtain a more structured representation and then used in the construction of resource-oriented graphs with consistently light computational complexity. The main contribution of this research is:

1. Modeling the utilization of multi-host resources as a discrete time series signal to support the analysis of load characteristics over time.
2. Integrating time-domain and frequency-domain feature extraction in the formation of host representation before graph construction.
3. Developing a graph-based host classification model that maintains lightweight properties to support the VM Placement process.

This research is structured into four main sections. Section I explains the background of the VM Placement problem, the development of related research, as well as the objectives and contributions proposed. Section II presents the system model and methods used, including modeling resource utilization as a time series, feature extraction processes, formation of resource-oriented graphs, as well as experimental scenarios and evaluation metrics. Section III presents the test results along with a comparative analysis against conventional approaches, including discussions on classification performance and computational complexity. Section IV summarizes the main findings of the research, their implications, and directions for future development.

II. Research Method

This section explains the stages of the research method which include system modeling, feature extraction, resource graph construction, and host classification as a pre-VM placement stage [15], [16]. The research framework is shown in Figure 1. The dataset used is the Bitbrains dataset which provides time series data of resource parameters such as CPU, memory, disk, and network from real data center workloads [17]–[19]. The dataset consists of multiple virtual machine traces collected from real data center environments. Each trace represents the temporal behavior of resource utilization over a certain observation period, including CPU, memory, disk, and network usage. The data are recorded at regular sampling intervals, forming consistent time-series signals for each resource. In this study, each virtual machine trace is treated as an individual host instance for analysis. Only segments with sufficient length are considered to ensure consistency in the feature extraction process. In general, the research method consists of signal modeling and segmentation, time and frequency domain feature extraction, four-node graph formation, and dynamic percentile-based host classification.

Figure 1 outlines the process starting with the collection of resource utilization data, including CPU, memory, storage, and network. This data is then modeled as a time series and processed through feature extraction stages in both the time domain and frequency domain. The obtained features are subsequently used in the formation of a resource-based graph to represent the relationships between resources. This graph representation serves as the basis for the host classification process to determine its operational condition before VM placement is carried out.

A. *Model System*

This study considers a cloud infrastructure consisting of several physical hosts running a number of virtual machines. The set of hosts is denoted as $H = \{h_1, h_2, \dots, h_m\}$, while the virtual machines to be allocated are denoted as $V = \{v_1, v_2, \dots, v_n\}$. Each host provides computing resources with limited capacity which serves as an upper limit in the allocation process. On each host h_i, there are four main components available: CPU, memory, storage, and network bandwidth. The total capacity of each of these components is expressed by equation (1) as follows.

$$C_i = \left(C_i^{cpu}, C_i^{ram}, C_i^{sto}, C_i^{cpu}\right) \quad (1)$$

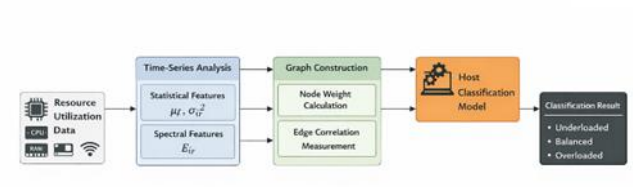

Figure 1. Host Classification Framework

The utilization values of each resource change over time along with the activities of the virtual machine. To capture these dynamics, observations are made periodically at fixed time intervals, forming a discrete time series. The resource utilization on host h_i for resource type r within an observation window is expressed by equation (2) as follows.

$$x_i^r[n], r \in \{cpu, ram, sto, net\} \quad (2)$$

With n representing the time index in the observation period. This approach allows each host to be represented based on utilization behavior over a certain time range, rather than just a single instantaneous value. This information is used in the pre-placement stage to identify the operational conditions of the host before virtual machines are allocated. The formulated system model is general and does not depend on a specific virtualization platform, so it can be applied to both simulation data and real data from data centers.

B. *Power Signal Modeling and Feature Extraction*

The resource utilization on each host, which has been expressed as the discrete-time series $x_i^r[n]$ in equation (2), is then processed to obtain a more concise representation through feature extraction. Before feature extraction, each resource

signal is normalized using min–max scaling to ensure comparability across different resource types and to prevent dominance of features with larger numerical ranges. This approach aims to summarize the characteristics of load behavior within an observation window of N samples [20]–[22]. In this study, a rectangular window is applied for segmentation, assuming uniform weighting across the observation window. The window size is set to N = 200 samples. This value is chosen to capture short-term workload variations while maintaining computational efficiency and stability of the extracted features. In the time domain, basic statistical parameters are calculated to describe the level of resource utilization and stability [23], [24]. The average utilization value in the observation is expressed by equation (3) as follows.

$$\mu_i^r = \frac{1}{N}\sum_{n=1}^{N} x_i^r[n] \quad (3)$$

In addition to the average, the level of load variation is calculated through variance, written with the following equation (4).

$$\sigma_i^{r^2}[k] = \frac{1}{N}\sum_{n=1}^{N}(x_i^r[n] - \mu_i^r)^2 \quad (4)$$

This parameter indicates how stable or fluctuating a resource is during the observation period. To complement the analysis in the time domain, the signal is also transformed into the frequency domain using the Discrete Fourier Transform (DFT), The transformation is applied over the defined observation window with a uniform sampling interval, following the original dataset configuration. The formulation is given in equation (5) as follows:

$$X_i^r[k] = \sum_{n=0}^{N-1} x_i^r[n]\, e^{-\frac{j2\pi kn}{N}} \quad (5)$$

From the results of the transformation, the spectral energy is calculated to depict the distribution of signal intensity in the frequency domain. The spectral energy is computed within each observation window and interpreted as a relative indicator of signal distribution, rather than an absolute measure across hosts, written with equation (6) as follows.

$$E_i^r = \sum_{k=0}^{N-1} |X_i^r[k]|^2 \quad (6)$$

The time domain and frequency domain features a combined into a feature vector for each host, expressed by equation (7) as follows.

$$F_i = \{\mu_i^r, \sigma_i^{r^2}, E_i^r\}, r \in \{cpu, ram, sto, net\} \quad (7)$$

This vector represents the comprehensive characteristics of resource utilization behavior within the observation window and is used as the basis for forming graph models and the host classification process in the subsequent stage. This feature extraction approach is designed to remain efficient because calculations are carried out within a limited window and do not require additional optimization iterations, thus maintaining computational complexity.

C. Graph Construction and Host Classification

After the feature vectorF_i for each host is obtained in equation (7), the next step is to build a graph-based representation that depicts the relationships between resources within a host. This representation is used to produce a more organized structure before the classification process is carried out [25]. The graph for each host is defined by the following equation (8).

$$G_i = (V_i, E_i) \quad (8)$$

Where V_i is the set of nodes representing resources $\{cpu, ram, sto, net\}$, and E_i is the set of edges showing the relationships between resources. The weight of each node is determined based on the extracted feature values. In general, the node weight for resource r on host $h_i$can be expressed by equation (9) as follows.

$$w_i^r = f_i^r \quad (9)$$

The graph structure is used to organize inter-resource relationships, while the final classification is based on the aggregated feature representation derived from node weights. With f_i^r being the integrated feature value of the resource. To describe the relationship between resources, the edge weight is determined based on the degree of similarity or correlation between two resources within the observation window. The correlation value between two resource signals can be calculated using equation (10) as follows.

$$p_i^{r,s} = \frac{\sum_{n=1}^{N}(x_i^r[n] - \mu_i^r)(x_i^s[n] - \mu_i^s)}{\sqrt{\sum_{n=1}^{N}(x_i^r[n] - \mu_i^r)^2}\sqrt{\sum_{n=1}^{N}(x_i^s[n] - \mu_i^s)^2}} \quad (10)$$

With$r \neq s$. This value reflects the level of inter-resource dynamic correlation. Based on the graph structure, each host is then classified into certain operational condition categories. The classification process is carried out by utilizing node weights and graph characteristics as input parameters. In general, the classification function can be expressed as

$$y_i = C(G_i) \quad (11)$$

Where y_i denotes the host condition label, for example underloaded, balanced, or overloaded, and $C(.)$ is the classification mechanism used. This classification stage serves

as a pre-placement process before the VM placement algorithm is executed. Thus, only hosts that meet certain criteria are considered as candidates for virtual machine placement. This approach allows for more structured decision-making without significantly increasing computational complexity. To provide a more explicit mathematical formulation, the classification rule based on the aggregated graph score is defined in Equation (12).

$$y_i = \begin{cases} \text{underloaded}, & S_i < P_{25} \\ \text{balanced}, & P_{25} \le S_i < P_{75} \\ \text{overloaded}, & S_i \ge P_{75} \end{cases} \quad (12)$$

Where P25 and P75 are the 25th and 75th percentiles of the score distribution, and Si is the final score of host i. According to this rule, hosts are put into three groups based on how their scores are spread out. Hosts with lower scores are called "underloaded," those with scores in the middle range are called "balanced," and those with higher scores are called "overloaded." This method uses the data's distribution, so the classification can change depending on how busy things are. To further clarify the implementation of the proposed method, the overall procedure is presented in Algorithm 1.

Algorithm 1: Proposed Host Classification Model

Input: Time-series data of CPU, memory, disk, and network
Output: Host class (underloaded, balanced, overloaded)

1: for each host i do
2: collect resource signals (CPU, memory, disk, network)
3: model each resource as a discrete time-series signal
4:
5: compute time-domain features:
6: mean and variance
7:
8: apply FFT to obtain frequency-domain features
9: compute spectral energy
10:
11: construct feature vector Fi
12:
13: build graph Gi with four nodes (CPU, memory, disk, network)
14: compute aggregated graph score Si
15:
16: end for
17:
18: determine percentile thresholds P25 and P75
19:
20: for each host i do
21: if Si < P25 then
22: yi = underloaded
23: else if P25 ≤ Si < P75 then
24: yi = balanced
25: else
26: yi = overloaded
27: end if
28: end for

The above procedure summarizes the main steps of the proposed method, starting from signal modeling and feature extraction to graph construction and host classification.

III. Results and Discusion

This section presents the results of the implementation of digital signal feature extraction methods and the four-node graph model on the Bitbrains dataset. Evaluation was carried out through time-domain signal analysis, frequency-domain analysis, spectral energy distribution among classes, as well as host classification results as a pre-VM placement stage.

A. Time Domain Analysis Results

As an initial stage of analysis, CPU signals are observed in the time domain to directly see the pattern of load changes against the time index. This visualization provides an overview of VM activity dynamics before transforming to the frequency domain. Based on Figure 2, at the beginning of the observation interval, the CPU value was at a very high level, approaching 90%, then experienced a sharp decline in a relatively short time until it approached zero. After this decline phase, the signal tended to stabilize at a very low level throughout the rest of the observation period. This pattern indicates a significant change in load conditions during the initial phase, which can be associated with the completion of computational processes or changes in VM status. The dominance of a low steady state condition after the initial transition indicates that most of the time interval is spent under minimal load conditions. This characteristic shows that the signal has a strong transient component at the beginning of the period, while in the subsequent part, the signal is relatively constant. The uneven amplitude variation becomes an important factor in forming statistical and spectral features used in the classification stage.

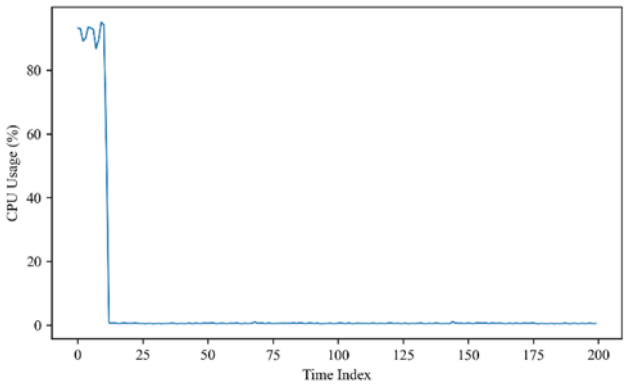

Figure 2. CPU signal in the time domain

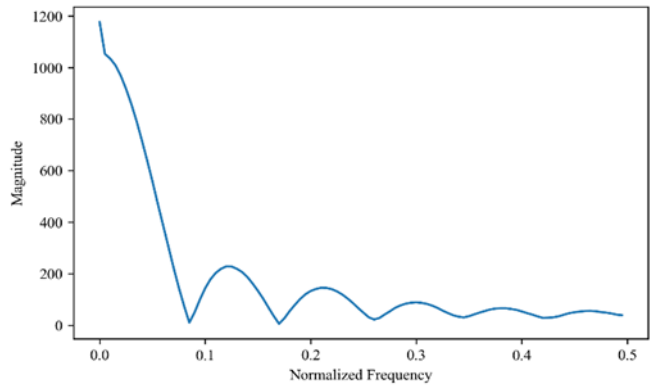

Figure 3. CPU Signal Frequency Spectrum

B. Results of Frequency Domain Analysis

To observe the spread signal energy with respect to frequency, the CPU signal is transformed into the frequency domain using Fast Fourier Transform (FFT).

Figure 3 shows that the largest spectrum magnitude is at low frequencies, especially around components close to zero. This indicates that CPU load changes are more influenced by slow variations than by rapid fluctuations. After a certain frequency, the magnitude decreases quite sharply and forms a waveform pattern with progressively smaller amplitudes. This pattern illustrates that the contribution of high frequencies to the overall signal energy is relatively small. To obtain a more stable picture of the power distribution, Power Spectral Density estimation was performed using the Welch method.

Figure 4 shows that the highest power density remains at low frequencies and then gradually decreases as the frequency increases. The resulting curve is relatively smooth and does not show significant random spikes. This indicates that the signal has a fairly regular spectral structure. The dominance of low frequencies reinforces the findings in the time-domain analysis, namely the presence of transient components at the beginning of the period followed by more stable load conditions. This power distribution is then utilized in the calculation of spectral energy as one of the determinants in the host classification process.

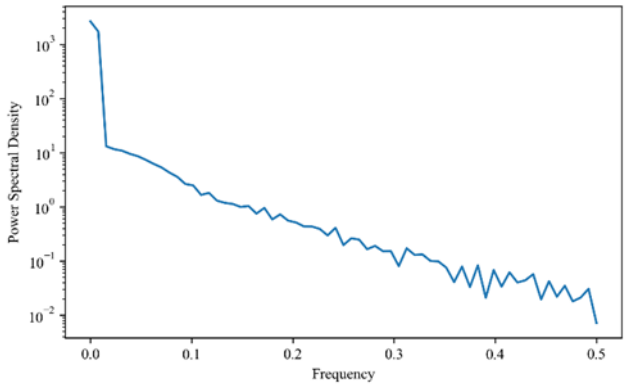

Figure 4. Power Spectral Density

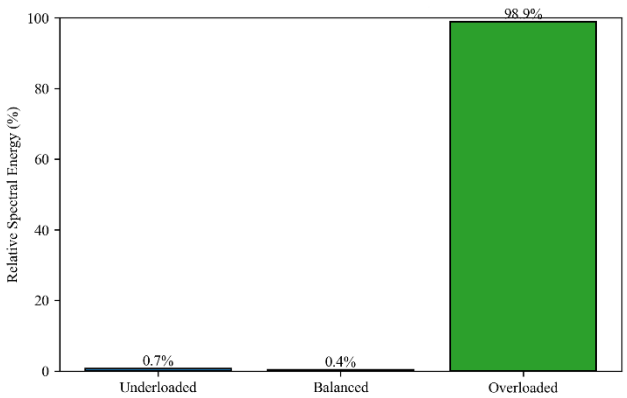

Figure 5. Relative Spectral Energy per Class

C. Comparison Results of Spectral Energy Between Classes

To observe the spectral characteristics capable of distinguishing host conditions, a comparison of average spectral energy between classes was conducted based on the classification results. Spectral energy was calculated from the Fourier transform of the CPU signal and then normalized as a percentage relative to the total energy of all classes.

Figure 5 shows that the overloaded class has a very dominant contribution to spectral energy, which is about 98.9%. In contrast, the underloaded and balanced classes contribute only about 0.7% and 0.4%, respectively. This difference indicates a significant gap in the intensity of signal dynamics between classes. Hosts in the overloaded condition generate much greater signal variations, resulting in a high accumulation of spectral energy. Meanwhile, in underloaded and balanced conditions, the signals tend to be more stable with relatively small fluctuations, so the energy produced is much lower. These findings indicate that spectral energy is not merely a mathematical representation in the frequency domain, but also quantitatively reflects the level of system activity. The larger the spectral energy, the higher the load dynamics occurring on the host. Thus, the difference in energy between classes provides a strong basis for the use of spectral features in the host classification process at the VM pre-placement stage.

D. Results of the Four-Node Graph Representation

After the signal features are extracted from each resource parameter, the next step is to represent the relationships between these resources in the form of a graph. This representation aims to observe the integration of CPU, memory, disk, and network contributions within a unified structure before the classification process is carried out.

Figure 6 shows a four-node graph model consisting of CPU, MEM, DISK, and NET as the main nodes. Each node represents the aggregate value of features resulting from the extraction of digital signal parameters, while the edges connect all pairs of nodes to depict the relationships between resources within a single host system.

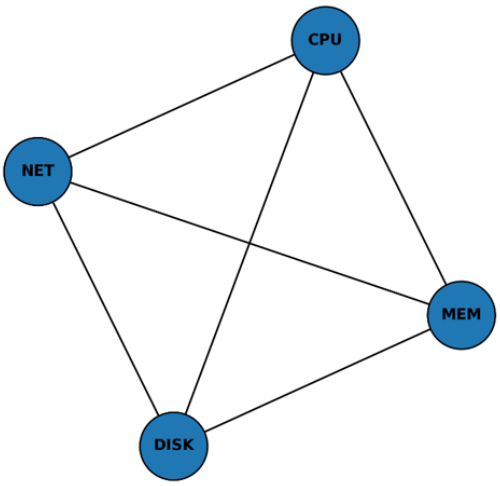

Figure 6. Four-Node Resource Graph Model

The resulting structure is a fully connected graph, so each component has direct or indirect relationships with the other components. This result indicates that system load cannot be viewed separately by parameter. Changes in the CPU, for example, often correlate with memory, disk, or network activity. By using a graph model, information from various signals can be integrated through the aggregation of node scores, producing a composite representation of the host's overall condition.

This approach provides a more comprehensive overview compared to single-parameter-based analysis, because classification decisions do not rely on just one type of resource, but on a combination of the dynamics of all system components. This graph representation serves as the basis for forming the final host score, which is then used in the classification process of underloaded, balanced, and overloaded. Thus, the four-node graph functions as a bridge between the results of digital signal feature extraction and the classification decision-making stage.

E. Host Classification Results

The final stage of the proposed method is grouping hosts into three categories based on the aggregate score of signal feature extraction results and graph representations. The classification process is carried out using a dynamic percentile approach to divide hosts into underloaded, balanced, and overloaded classes. Figure 7 shows the distribution of the number of hosts in each class. Out of the total hosts analyzed, 421 hosts were categorized as overloaded, while 409 hosts were in the underloaded category and 409 hosts in the balanced category. This distribution shows a relatively even division among the classes, without extreme dominance in any category.

The results indicate that the classification method based on spectral energy and graph aggregation can produce structured segmentation based on the distribution of data characteristics. The nearly balanced division also shows that the use of dynamic percentile-based thresholds operates adaptively to the distribution of actual scores, than based on fixed cutoff values.

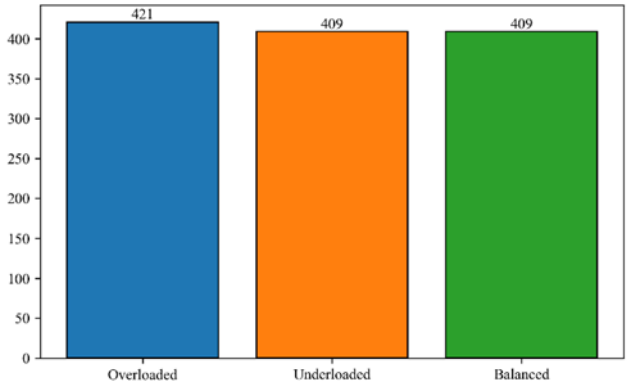

Figure 7. Host Classification Distribution

Thus, the resulting classification reflects the relative conditions among hosts in the system, making it suitable to be used as a basis for decision-making in the VM pre-placement stage.

F. Classification Performance Evaluation

To evaluate the performance of the proposed method, several standard metrics are considered, including accuracy, precision, recall, and F1-score. The overall results are summarized in Table 1.

Tabel 1. Classification Performance of the Proposed Method

Method	Accuracy	Precision	Recall	F1-score
Proposed	0.35	0.96	0.35	0.47

As a simple reference, a baseline based on average CPU utilization is considered to illustrate the difference between single-resource and multi-resource representations. Table 1 shows that the proposed method gets a high precision value, which means that the class labels it gives are usually correct. But the accuracy and recall values are not very high, which means that some workload conditions may not be fully captured. Figure 8 shows the confusion matrix, which gives a more detailed look at the classification results.

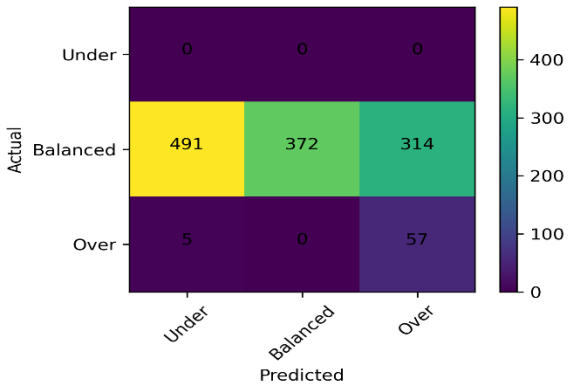

Figure 8. Confusion matrix of the proposed

Figure 8 shows how the predicted and actual class labels are spread out. It is evident that the majority of accurate classifications are found within the predominant class, whereas certain misclassifications arise between neighboring workload conditions. This means that some patterns of work have similar traits, which makes it harder to tell them apart. These results reflect the characteristics of real-world workload distributions, where the imbalance between different host conditions and overlapping behavior can influence classification performance. In addition, the relatively small proportion of hosts classified as balanced indicates that fewer workload instances remain within a stable intermediate range, as most conditions tend to shift toward lower or higher utilization levels.

The proposed approach, based on signal-driven feature representation and structured classification, is able to capture the dominant patterns of resource utilization in a consistent manner. This also reflects the balance between model simplicity and performance, particularly in scenarios that require lightweight and adaptive solutions. This classification can assist in determining appropriate VM placement. Hosts classified as underloaded still have sufficient resources to accommodate additional VMs and are therefore suitable candidates for VM allocation. In contrast, overloaded hosts exhibit high resource utilization, indicating that load redistribution or VM migration should be considered to prevent performance degradation. Meanwhile, hosts categorized as balanced demonstrate relatively stable resource usage, suggesting that no significant adjustments are required. The resulting classification provides a more structured representation of host conditions and supports more informed decision-making for resource management in dynamic environments.

G. Discussion

The results of the time-domain analysis up to the classification stage show a consistent correlation between the signal characteristics and the host load conditions. In the time domain, there is a strong transient component at the beginning of the observation period, followed by relatively stable conditions. This pattern indicates that the load dynamics do not remain constant but undergo significant changes at certain intervals. Frequency-domain analysis shows the dominance of low-frequency components in both the FFT and PSD spectra. The concentration of energy in the low-frequency range indicates that load changes tend to occur gradually, not because of random rapid fluctuations. The gradually decreasing power distribution shows that the signal has an orderly structure and can be represented through spectral parameters. The comparison of spectral energy between classes shows very noticeable differences, especially in the overloaded class, which has much greater energy contribution compared to the other classes.

This high energy reflects a greater intensity of signal variation, while the underloaded and balanced classes show more stable characteristics with relatively small energy. These quantitative differences support the use of spectral energy as an indicator of host condition. The four-node graph model plays a role in integrating all resource parameters into a single connected representation. With this approach, classification decisions are not determined by a single parameter, but rather by a combination of CPU, memory, disk, and network dynamics. The integration produced host segmentation that is more representative of the actual conditions of the system. These findings indicate that the approach used is able to map host conditions with a pattern consistent with system load dynamics. The resulting spectral characteristics align with the classification results obtained and provide a quantitative basis in the decision-making process before virtual machine deployment is carried out.

The results indicate that some workload conditions are difficult to separate, especially between underloaded and balanced states. Their patterns look quite similar in several cases, which explains why misclassification still occurs. Even so, when the model assigns a label, it is usually consistent with the observed resource behavior, as reflected by the high precision value. This situation is quite common in real cloud environments, where resource usage is uneven and constantly changing. In this context, the proposed method is still able to capture the main behavior of the workload and provide a simple way to classify hosts without relying on a training process.

There are a few things to keep in mind about this study. There hasn't been any supervised validation of the classification method, so we can't measure how well it works. Also, using a fixed observation window might make it harder to see changes in workload patterns that happen quickly. Also, the proposed method has not been directly compared to machine learning-based methods, so it is still not clear how well it works compared to other methods.

IV. Conslusion

The conclusions from the research results are explained in this section:

1. Time and frequency domain analysis shows that the host load signal has a structured pattern with a dominance of low-frequency components.
2. Relative spectral energy can clearly distinguish host classes, especially under overloaded conditions which have the highest energy values.
3. A four-node graph representation allows the integration of CPU, memory, disk, and network into a single connected model to produce a comprehensive classification score.
4. From a practical perspective, the proposed approach can be utilized as a pre-placement mechanism to support VM placement decisions by identifying suitable hosts prior to allocation.
5. This study has several limitations, including the absence of supervised validation, the use of a fixed observation window that may limit the capture of rapid

workload changes, and the lack of comparison with machine learning-based approaches.

V. References

[1] Z. Ding, Y.-C. Tian, Y.-G. Wang, W. Zhang, and Z.-G. Yu, "Progressive-fidelity computation of the genetic algorithm for energy-efficient virtual machine placement in cloud data centers," *Appl. Soft Comput.*, vol. 146, p. 110681, 2023, doi: https://doi.org/10.1016/j.asoc.2023.110681.

[2] K. Sindhu and K. Seshadri, "A dynamic probabilistic graphical model for mapping tasks to virtual machines in data centers," *Eng. Appl. Artif. Intell.*, vol. 155, p. 110963, 2025, doi: https://doi.org/10.1016/j.engappai.2025.110963.

[3] L. M. Silalahi and D. Gunawan, "Routing Architecture: Pioneering Traffic Engineering for Multiprotocol Label Switching and Beyond," in *2025 IEEE 15th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, IEEE, May 2025, pp. 373–378. doi: 10.1109/ISCAIE64985.2025.11080920.

[4] S. M. Rozehkhani, F. Mahan, and W. Pedrycz, "VM consolidation steps in cloud computing: A perspective review," *Simul. Model. Pract. Theory*, vol. 138, p. 103034, 2025, doi: https://doi.org/10.1016/j.simpat.2024.103034.

[5] D. Wang *et al.*, "Towards dynamic virtual machine placement based on safety parameters and resource utilization fluctuation for energy savings and QoS improvement in cloud computing," *Futur. Gener. Comput. Syst.*, vol. 171, p. 107853, 2025, doi: https://doi.org/10.1016/j.future.2025.107853.

[6] S. O. Ogundoyin and I. A. Kamil, "Optimal fog node selection based on hybrid particle swarm optimization and firefly algorithm in dynamic fog computing services," *Eng. Appl. Artif. Intell.*, vol. 121, p. 105998, 2023, doi: https://doi.org/10.1016/j.engappai.2023.105998.

[7] Y. Chen, Z. Liu, Y. Zhang, Y. Wu, X. Chen, and L. Zhao, "Deep Reinforcement Learning-Based Dynamic Resource Management for Mobile Edge Computing in Industrial Internet of Things," *IEEE Trans. Ind. Informatics*, vol. 17, no. 7, pp. 4925–4934, Jul. 2021, doi: 10.1109/TII.2020.3028963.

[8] P. Tamilarasu and G. Singaravel, "Quality of service aware improved coati optimization algorithm for efficient task scheduling in cloud computing environment," *J. Eng. Res.*, vol. 12, no. 4, pp. 768–780, 2024, doi: https://doi.org/10.1016/j.jer.2023.09.024.

[9] P. K. K. Ranganna, S. G. Matt, C.-L. Chen, A. B. Jayachandra, and Y.-Y. Deng, "Fitness Sharing Chaotic Particle Swarm Optimization (FSCPSO): A Metaheuristic Approach for Allocating Dynamic Virtual Machine (VM) in Fog Computing Architecture," *Computers, Materials \& Continua*, vol. 80, no. 2. pp. 2557–2578, 2024. doi: 10.32604/cmc.2024.051634.

[10] M. S. A. Khan and R. Santhosh, "Hybrid Optimization Algorithm for VM Migration in Cloud Computing," *Comput. Electr. Eng.*, vol. 102, p. 108152, 2022, doi: https://doi.org/10.1016/j.compeleceng.2022.108152.

[11] M. Selselejoo and H. Ahmadifar, "DT-GWO: A hybrid decision tree and GWO-based algorithm for multi-objective task scheduling optimization in cloud computing," *Sustain. Comput. Informatics Syst.*, vol. 47, p. 101138, 2025, doi: https://doi.org/10.1016/j.suscom.2025.101138.

[12] F. Casino, P. Lopez-Iturri, and C. Patsakis, "Cloud continuum testbeds and next-generation ICTs: Trends, challenges, and perspectives," *Comput. Sci. Rev.*, vol. 56, p. 100696, 2025, doi: https://doi.org/10.1016/j.cosrev.2024.100696.

[13] C. Liu, L. Ma, M. Zhang, and H. Long, "Optimizing cloud resource management with an IoT-enabled optimized virtual machine migration scheme for improved efficiency," *J. Netw. Comput. Appl.*, vol. 237, p. 104137, 2025, doi: https://doi.org/10.1016/j.jnca.2025.104137.

[14] J. Zhao, "Virtualization resource scheduling and optimization method based on swarm intelligent systems," *Intell. Syst. with Appl.*, vol. 25, p. 200469, 2025, doi: https://doi.org/10.1016/j.iswa.2024.200469.

[15] B. Liang, X. Dong, Y. Wang, and X. Zhang, "A high-applicability heterogeneous cloud data centers resource management algorithm based on trusted virtual machine migration," *Expert Syst. Appl.*, vol. 197, p. 116762, 2022, doi: https://doi.org/10.1016/j.eswa.2022.116762.

[16] D. Wang, Y. Li, W. Zhang, Z. Yu, Y.-C. Tian, and K. Li, "EVRM: Elastic Virtual Resource Management framework for cloud virtual instances," *Futur. Gener. Comput. Syst.*, vol. 165, p. 107569, 2025, doi: https://doi.org/10.1016/j.future.2024.107569.

[17] T. Hidayat, K. Ramli, N. Thereza, A. Daulay, R. Rushendra, and R. Mahardiko, "Machine Learning to Estimate Workload and Balance Resources with Live Migration and VM Placement," *Informatics*, vol. 11, no. 3. p. 50, 2024. doi: 10.3390/informatics11030050.

[18] M. Smendowski and P. Nawrocki, "Optimizing multi-time series forecasting for enhanced cloud resource utilization based on machine learning," *Knowledge-Based Syst.*, vol. 304, p. 112489, 2024, doi: https://doi.org/10.1016/j.knosys.2024.112489.

[19] A. Tandon and S. Patel, "DBSCAN based approach for energy efficient VM placement using medium level CPU utilization," *Sustain. Comput. Informatics Syst.*, vol. 43, p. 101025, 2024, doi: https://doi.org/10.1016/j.suscom.2024.101025.

[20] G. Vivar-Estudillo, R. J. P. Chimal, J. U. Muñoz-Minjares, O. Ibarra-Manzano, and C. Lastre-Domínguez, "Spectral feature extraction using the

UFIR iterative smoother algorithm for ECG signal classification," *Biomed. Signal Process. Control*, vol. 110, p. 108007, 2025, doi: https://doi.org/10.1016/j.bspc.2025.108007.
[21] L. Li, X. Li, L. Jiang, X. Su, and F. Chen, "A review on deep learning techniques for cloud detection methodologies and challenges," *Signal, Image Video Process.*, vol. 15, no. 7, pp. 1527–1535, 2021, doi: 10.1007/s11760-021-01885-7.
[22] J. Ning, L. Xie, J. Yin, and Y. Liu, "Cloud Removal Advances: A Comprehensive Review and Analysis for Optical Remote Sensing Images," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 18, pp. 15914–15930, 2025, doi: 10.1109/JSTARS.2025.3580718.
[23] M. Baygin *et al.*, "A Hand-Modeled Feature Extraction-Based Learning Network to Detect Grasps Using sEMG Signal," *Sensors*, vol. 22, no. 5. p. 2007, 2022. doi: 10.3390/s22052007.
[24] R. Li, H. Wan, J. Wang, S. Huo, and C. Guedes Soares, "A method for accurate extraction of three-dimensional point cloud feature data in loading test of container ships," *Ocean Eng.*, vol. 312, p. 119134, 2024, doi: https://doi.org/10.1016/j.oceaneng.2024.119134.
[25] G. Cheng, Z. Zhang, Q. Li, Y. Li, and W. Jin, "Energy Theft Detection in an Edge Data Center Using Deep Learning," *Math. Probl. Eng.*, vol. 2021, no. 1, p. 9938475, Jan. 2021, doi: https://doi.org/10.1155/2021/9938475.