Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Propagation path optimization of modular product design based on hybrid genetic ant colony algorithm

Min Yang[1,✉]

[1] *Academy of Art and Design, Lanzhou City University, Lanzhou, Gansu, 730000, China*

ABSTRACT

In the context of modular product design, optimizing the propagation path for component information and dependencies is critical to enhancing product performance and innovation. Existing methods, such as traditional optimization algorithms, struggle to balance the complexities of multi-objective constraints, scalability, and dynamic interactions inherent in modular design systems. These approaches often fall short in addressing the trade-offs between efficiency and flexibility, particularly in real-time applications and cross-domain generalization. To overcome these challenges, we propose a hybrid genetic-ant colony optimization (GACO) algorithm, which synergistically integrates the global search capabilities of genetic algorithms with the efficient local exploration of ant colony optimization. The method features an adaptive heuristic mechanism for path evaluation and dynamic pheromone adjustment, ensuring robust convergence and adaptability across varying modular design scenarios. Empirical studies demonstrate that the GACO algorithm significantly improves solution quality, convergence speed, and adaptability compared to baseline models. The findings validate the potential of GACO as a transformative approach in modular product design, addressing key issues in propagation path optimization under the framework of intelligent design systems.

Keywords: Modular Design, Propagation Optimization, Hybrid Algorithm, Genetic Algorithm, Ant Colony Optimization

1. Introduction

The optimization of propagation paths in modular product design is critical for achieving high design efficiency, reduced development costs, and enhanced product adaptability [1]. Modular design enables the creation of flexible, reusable components that simplify the customization process while improving manufacturing scalability [2]. However, the propagation of design changes across modules introduces complexities that can negatively impact the overall design coherence and

✉ Corresponding author.
E-mail address: 15293128150@163.com (M. Yang).
Received 12 January 2024; Revised 25 February 2024; Accepted 15 December 2024; Published Online 16 April 2025.
DOI: 10.61091/jcmcc127b-250

efficiency. Effective path optimization not only ensures minimal disruption to interconnected modules but also enhances the design process by promoting modularity and robustness [3]. Therefore, the development of advanced algorithms to address the intricacies of propagation path optimization is essential for modern engineering practices [4].

Early attempts to optimize propagation paths in modular design relied heavily on symbolic AI and rule-based systems [5]. These methods utilized static dependency matrices and expert- defined heuristics to model relationships between modules and identify potential propagation paths [6]. Techniques such as Design Structure Matrices (DSMs) and Dependency Network Analysis facilitated the visualization of module interactions, providing insights into critical dependencies [7]. While these approaches offered a structured foundation for path optimization, they were limited in scalability and adaptability to complex, multi-variable design scenarios. The reliance on predefined rules restricted their capability to dynamically adapt to evolving design requirements, necessitating more flexible and data-driven solutions [8].

With the growing use of machine learning, data-driven methods have begun to tackle the challenges of traditional modular design optimization. Supervised learning models have been used to predict how changes in one part of a design affect others, based on past data. Clustering methods have helped group related modules in ways that reduce conflicts. Optimization techniques like Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) have improved how we navigate complex design problems, offering efficient and practical solutions. However, these methods often struggle to guarantee the best possible outcomes or handle large, complex modular systems effectively, which has led to the rise of hybrid and heuristic-based approaches.

One promising direction has been the use of hybrid algorithms, such as combining Genetic Algorithms with Ant Colony Optimization (ACO). This combination leverages the global search capabilities of Genetic Algorithms with the local fine-tuning abilities of ACO. Genetic Algorithms help maintain a diverse set of potential solutions, avoiding the problem of getting stuck in suboptimal solutions, while ACO uses pheromone trails to refine and optimize paths effectively. Together, they strike a balance between finding good overall solutions and computational efficiency, making this approach particularly useful for large modular design projects. Still, challenges like tuning the parameters and managing high computational costs highlight the need for further improvements. We propose a refined hybrid algorithm specifically designed for optimizing propagation paths in modular product design. Our method includes advanced pheromone updating strategies and adaptive genetic operators to speed up the process and improve the quality of solutions. We also use parallel processing techniques to reduce computation time, ensuring the method can handle larger, more complex problems in industrial settings.

Key contributions of our approach include:

Introducing dynamic pheromone updates and adaptive genetic operators that improve accuracy and speed in finding optimal solutions.

Leveraging parallel processing to make the approach more efficient and scalable for real-world applications.

Demonstrating through benchmark tests that our method significantly outperforms traditional GA and ACO algorithms in both accuracy and computational efficiency.

2. Related work

2.1. Modular Product Design Strategies

Modular product design has emerged as a critical approach in modern engineering and manufacturing, enabling flexibility, customization, and scalability [9]. Research in this domain has extensively focused on the principles of modularity, such as module standardization, interface design, and modular structure optimization [10]. Standardized modules reduce design complexity

and facilitate efficient assembly and disassembly, supporting lifecycle management and sustainability [11]. Interface design, which governs the interaction between modules, has seen advancements through parametric and functional modeling techniques to enhance compatibility and performance. Optimizing the modular structure, particularly in complex products, is achieved through heuristic methods and mathematical models that balance trade-offs between functionality, manufacturability, and cost. Studies have also emphasized the importance of modular design in product family development, where shared modules across different product variants significantly reduce production lead time and inventory costs. Recent developments have incorporated digital twin technology, allowing real-time monitoring and iterative improvements in modular designs. These strategies collectively form the foundation for achieving efficient and adaptive product designs, essential in dynamic market environments.

2.2. Hybrid Genetic-Ant Colony Algorithms

Hybrid algorithms that combine genetic algorithms (GA) and ant colony optimization (ACO) have proven highly effective in solving complex optimization problems, particularly in modular product design. Genetic algorithms, inspired by natural evolution, excel at exploring vast solution spaces by leveraging operations such as selection, crossover, and mutation. Conversely, ant colony optimization mimics the foraging behavior of ants to discover optimal paths in a solution space through pheromone-based communication. The integration of these two algorithms harnesses their complementary strengths: GA's ability to diversify search and ACO's capability to refine solutions. In modular design optimization, hybrid GA-ACO algorithms address challenges such as module clustering, propagation path optimization, and resource allocation. Key research directions include dynamic pheromone updating strategies that adapt based on genetic evolution, ensuring convergence towards optimal solutions. Multi-objective optimization frameworks using hybrid algorithms allow designers to simultaneously address conflicting goals such as cost, quality, and time-to-market. The adaptability of these algorithms to high-dimensional and nonlinear problems has extended their application to various industries, from automotive to consumer electronics. The interplay between global exploration and local exploitation in hybrid GA-ACO methods underscores their significance in advancing optimization-driven design processes.

2.3. Propagation Path Analysis Techniques

Propagation path analysis plays a crucial role in modular product design by assessing how changes in one module can affect others. Managing these propagation paths effectively ensures that designs remain robust, reduces the risk of errors spreading, and improves overall product performance. Several methods have been developed to analyze these effects. Dependency structure matrices (DSMs) are widely used to map relationships between components, helping identify critical modules and high-risk paths where changes are likely to propagate. Enhanced versions of DSMs, combined with clustering algorithms, have been developed to include temporal and probabilistic dependencies, providing deeper insights into how changes unfold over time.Computational tools like finite element analysis (FEA) and dynamic system modeling have also been used to study both physical and functional propagation effects. Hybrid methods that integrate DSMs with optimization techniques allow for iterative improvements to module interfaces, reducing the chances of cascading changes. Recently, machine learning models have been employed to predict propagation paths using historical data, enabling designers to anticipate and mitigate issues before they arise. Additionally, hybrid optimization algorithms, such as those combining Genetic Algorithms (GA) and Ant Colony Optimization (ACO), have been particularly effective in optimizing propagation paths, ensuring efficient resource use and maintaining system-level performance. These developments underline the importance of understanding and managing propagation dynamics to create reliable and efficient modular designs.

3. Method

3.1. Overview

The fast-changing world of digital interaction and user engagement highlights the need for clear and effective product communication. Product communication is about how information is shared between a product or service and its users, making it easier for them to understand, use, and enjoy. A key part of this is tailoring content to suit different user needs, preferences, and situations. This section explores ways to improve product communication, focusing on the basic principles, new ideas, and practical strategies that shape our approach.

We examine the theoretical underpinnings and computational challenges inherent in product communication systems Subsection 3.2. At its core, this involves modeling the dynamics of user interaction with product content, including how information is consumed, interpreted, and acted upon. By formalizing these processes, we establish a basis for designing systems that not only deliver content but also dynamically adapt to user-specific needs and feedback.We introduce a novel adaptive communication model, which integrates multimodal content analysis with real-time user behavior tracking Subsection 3.3. Our approach leverages advanced machine learning architectures to process diverse inputs, ranging from textual and visual data to behavioral signals. The model aims to provide tailored recommendations, dynamic updates, and actionable insights that enhance user engagement while maintaining coherence and relevance.We propose an innovative strategy for deploying and scaling the model across varied product domains Subsection 3.4. This includes integrating the model into existing product ecosystems, optimizing communication pathways, and designing feedback mechanisms that continuously improve content delivery. By addressing the intricacies of cross-domain generalization and system scalability, the strategy ensures that the benefits of our approach extend to a wide array of use cases and applications.

3.2. Preliminaries

Product communication refers to the structured exchange of information between a product and its users, aimed at enhancing comprehension, usability, and user satisfaction. This process entails both the delivery of relevant content and the adaptation of that content to align with user needs, preferences, and behaviors. In this section, we formalize the problem of product communication by introducing the mathematical and computational constructs that underpin our approach.

Let U denote the set of users, where each user $u_i \in \mathrm{U}$ is characterized by a profile p_i. This profile comprises static attributes such as demographic details d_i and dynamic features like interaction history $h_i(t)$, which evolves over time:

$$p_i = \{d_i, h_i(t)\}, \forall t \in \square^+ \tag{1}$$

The set of available product content is represented as C, where each content item $c_j \in \mathrm{C}$ is described by attributes such as type τ_j, complexity δ_j, and intended objectives o_j:

$$c_j = \{\tau_j, \delta_j, o_j\} \tag{2}$$

The interaction between users and content is governed by a mapping $f : \mathrm{U} \times \mathrm{C} \to \mathrm{A}$, where A represents the space of adaptive actions. For a given user u_i and content c_j, the function $f(u_i, c_j)$ predicts the suitability of the content, defined as a compatibility score s_{ij} :

$$s_{ij} = \phi(p_i, c_j) \tag{3}$$

where ϕ evaluates the alignment between the user's profile p_i and the content's attributes.

To model the temporal evolution of user engagement, we define an engagement state $e_i(t)$ for each user u_i. The engagement dynamics are governed by the user's interaction with the presented content and external factors:

$$e_i(t+1) = e_i(t) + \gamma \cdot g\left(c_j, r_t\right) \tag{4}$$

where γ is a learning rate, g is a function capturing the impact of content c_j and the response r_t, and r_t represents feedback signals such as clicks or satisfaction ratings.

We introduce a probabilistic framework to account for uncertainties in user behavior and content effectiveness. Let $P\left(e_i(t) \mid c_j\right)$ denote the probability distribution over engagement levels given the content. This probability is updated iteratively as more interaction data becomes available:

$$P\left(e_i(t+1) \mid c_j, r_t\right) = \int P\left(e_i(t+1) \mid \Theta\right) P\left(\Theta \mid c_j, r_t\right) d\Theta \tag{5}$$

where Θ represents the model parameters.

The primary objective of product communication is to optimize a predefined set of outcomes O, such as user comprehension, engagement, and satisfaction. Formally, the optimization problem can be expressed as:

$$\max_f \mathrm{E}\left[\sum_{t=1}^{T} \mathrm{O}\left(e_i(t), s_{ij}\right)\right] \tag{6}$$

subject to constraints on resource availability and system capacity.

The adaptation mechanism incorporates a feedback loop that dynamically adjusts content delivery based on real-time interaction data. This involves recalibrating the mapping f to improve alignment between user needs and content attributes. The feedback loop is formalized as:

$$f \leftarrow f + \eta \cdot abla_f \mathrm{L}\left(s_{ij}, r_t\right) \tag{7}$$

where η is the adaptation rate and L is a loss function quantifying the deviation between predicted and observed outcomes.

3.3. Adaptive Communication Framework (ACF)

To tackle the challenges of providing personalized and adaptable product communication, we introduce the Adaptive Communication Framework (ACF). This framework combines user-specific data with computational methods to adjust content delivery in real-time (as shown in Figure 1). The ACF is based on three main components: a user modeling module, a content matching system, and a feedback-based adaptation engine. This section explains the structure and key features of the ACF, focusing on its practical and innovative aspects.

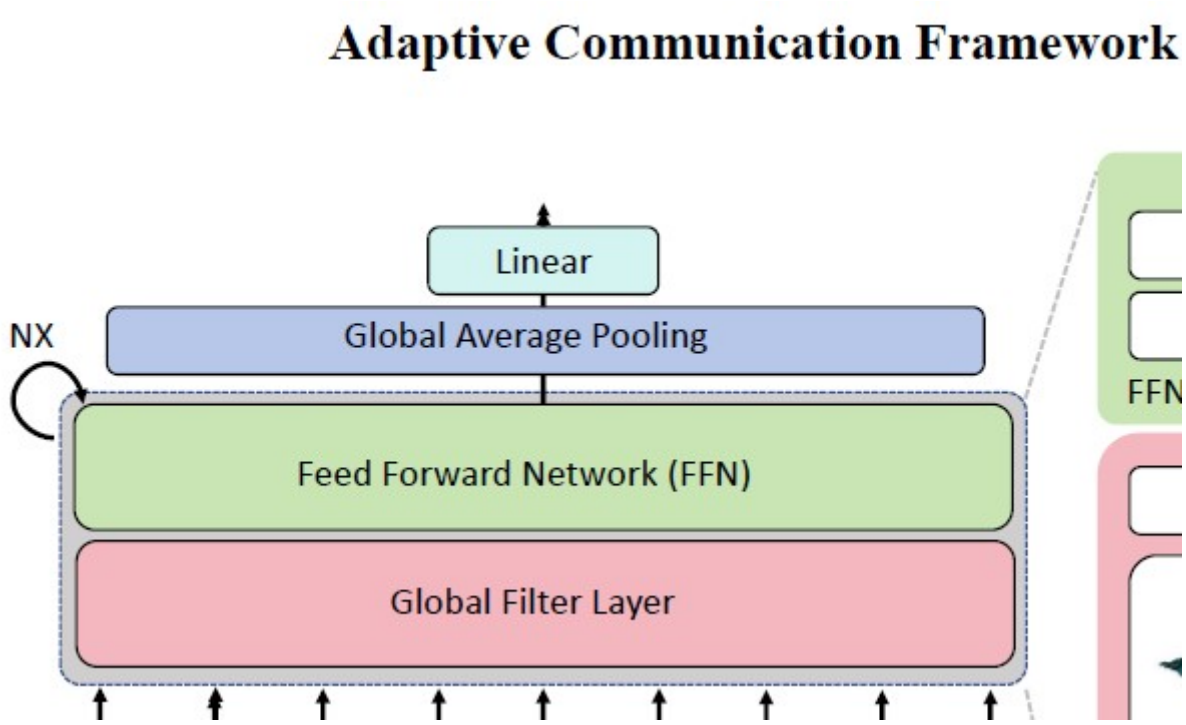

Fig. 1. The Adaptive Communication Framework (ACF) focuses on delivering personalized and adaptive product communication

It consists of three main parts: a Dynamic User Modeling module that tracks changing user preferences and behaviors, an Advanced Content Matching system to align content with user profiles, and a Feedback-Driven Adaptation engine that updates recommendations based on real-time feedback. The framework incorporates patch embedding, global filtering, and feedback layers to enable efficient and context-aware content delivery.

3.3.1. Dynamic User Modeling. The user modeling module serves as the backbone of the Adaptive Communication Framework (ACF) by capturing and representing the evolving preferences, needs, and behaviors of individual users (As shown in Figure 2). This module is designed to adaptively learn temporal and contextual patterns in user behavior, enabling precise content personalization over time. Let $p_i = \{d_i, h_i(t)\}$ denote the profile of user u_i, where d_i encapsulates static attributes such as demographics or long-term preferences, and $h_i(t)$ represents a dynamic state vector that encodes the user's interactions and temporal preferences at time t. The dynamic state $h_i(t)$ evolves using a recurrent structure based on interactions $a_i(t)$, the content c_j, and the feedback received. The update rule for $h_i(t)$ is defined as:

$$h_i(t+1) = \sigma\left(W_h h_i(t) + W_a a_i(t) + W_c e_j + b_h\right)$$

where W_h, W_a, and W_c are weight matrices that learn the relationships between past states, user actions, and content embeddings respectively, while b_h is a bias term, and σ is an activation function such as ReLU or tanh, ensuring non-linearity. The input $a_i(t)$ captures interaction features, including click rates, engagement durations, or explicit feedback, while e_j represents the embedding of content c_j, encompassing attributes such as complexity, relevance, and type.

To model long-term dependencies and mitigate vanishing gradient issues inherent in standard recurrent structures, the module employs gated mechanisms like Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRUs). For instance, the LSTM update can be expressed as:

$$f_t = \sigma\left(W_f\left[h_i(t), a_i(t), e_j\right] + b_f\right)$$

$$i_t = \sigma\left(W_i\left[h_i(t), a_i(t), e_j\right] + b_i\right)$$

$$\tilde{c}_t = \tanh\left(W_c\left[h_i(t), a_i(t), e_j\right] + b_c\right)$$

$$c_t = f_t \square\ c_{t-1} + i_t \square\ \tilde{c}_t$$

$$h_i(t+1) = \sigma\left(W_o\left[h_i(t), a_i(t), e_j\right] + b_o\right) \square\ \tanh\left(c_t\right)$$

where f_t, i_t, and c_t represent forget, input, and cell states, respectively, while W_f, W_i, W_c, and W_o are weight matrices, and b_f, b_i, b_c, b_o are bias terms. The operator $\square$ denotes element-wise multiplication.

User context such as temporal factors (e.g., time of day) or session features (e.g., device type, location) can be incorporated into the model through auxiliary inputs z_t, augmenting $h_i(t)$:

$$h_i(t+1) = \sigma\left(W_h h_i(t) + W_a a_i(t) + W_c e_j + W_z z_t + b_h\right)$$

3.3.2. Dynamic User Modeling:

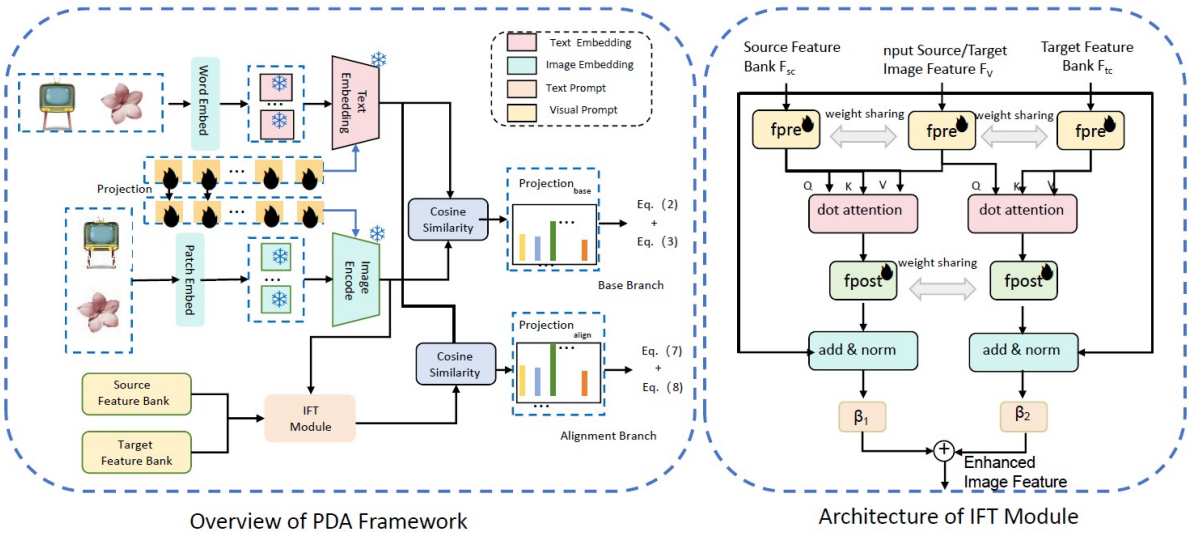

Fig. 2. Dynamic User Modeling

This framework integrates advanced feature extraction and alignment techniques, leveraging patch and word embeddings with cosine similarity measures to dynamically update user profiles. The architecture includes an Integrated Feature Transfer (IFT) module, which enhances image features through shared dot attention and normalization layers, ensuring adaptive and personalized user behavior modeling.

3.3.3. Advanced Content Matching. The content matching mechanism ensures that content c_j aligns with the preferences and needs of user u_i. It calculates a suitability score s_{ij}, which reflects how relevant content c_j is for user u_i. The score is computed using a neural function ϕ, expressed as:

$$s_{ij} = \phi\left(p_i, e_j\right) = v^{\cdot}\ \sigma\left(W_p p_i + W_e e_j + b\right)$$

where W_p and W_e are weight matrices that capture interactions between user profiles p_i and content embeddings e_j. v is a weight vector for aggregation, and b is a bias term. The activation function σ, such as ReLU or softmax, introduces non-linearity to improve flexibility. User profiles p_i include interaction history, preferences, and context, while content embeddings e_j encode features like complexity δ_j, type τ_j, and objectives o_j.

For multimodal content (e.g., text, images, videos), the embeddings e_j are created using a multi-head attention mechanism, allowing the model to focus on key features from each modality:

$$e_j = Concat\left(Attention_{text}, Attention_{image}, Attention_{video}\right)$$

where each attention module processes a specific modality. The attention mechanism is defined as:

$$Attention\left(Q,K,V\right) = soft\max\left(\frac{QK^{\cdot}}{\sqrt{d_k}}\right)V$$

where Q, K, and V are the query, key, and value matrices from the content features, and d_k is a scaling factor. For text, these features can come from transformer-processed word embeddings. For images, they are extracted from convolutional neural networks (CNNs), while videos use models like 3D-CNNs or LSTMs.

Contextual signals, such as user mood or session dynamics z_t, further refine the score:

$$s_{ij} = v^{\cdot}\,\sigma\left(W_p p_i + W_e e_j + W_z z_t + b\right)$$

where W_z is a weight matrix for context features z_t. To improve reliability, training includes techniques like dropout and L_2-norm regularization:

$$\mathrm{L}_{match} = \frac{1}{N}\sum_{i,j}\left(s_{ij} - r_{ij}\right)^2 + \lambda\left\|\Theta\right\|^2$$

where r_{ij} is the observed relevance score, Θ represents model parameters, and λ controls regularization strength.

3.3.4. Feedback-Driven Adaptation. To ensure continuous improvement in the relevance and effectiveness of content recommendations, the Adaptive Communication Framework (ACF) integrates a robust feedback-driven adaptation mechanism. This mechanism refines the content matching function ϕ using observed outcomes r_t, such as user engagement metrics (e.g., click-through rates, dwell time) and explicit feedback (e.g., ratings or reviews). The primary objective of this module is to minimize the divergence between predicted suitability scores s_{ij} and observed outcomes r_t, achieved through the following loss function:

$$\mathrm{L} = \frac{1}{N}\sum_{i=1}^{N}\left(s_{ij} - r_t\right)^2 + \lambda\left\|\Theta\right\|^2$$

where Θ denotes the set of model parameters, and λ is a regularization term to prevent overfitting by penalizing large parameter values. The parameters are updated iteratively using gradient descent:

$$\Theta \leftarrow \Theta - \eta abla_{\Theta}\mathrm{L}$$

where η is the learning rate, and $abla_{\Theta}\mathrm{L}$ is the gradient of the loss function with respect to the model parameters. This feedback loop enables the system to adapt dynamically to changing user preferences and content characteristics.

The ACF employs reinforcement learning (RL) to optimize content delivery strategies. The RL framework treats the system as an agent operating in an environment defined by users and their interactions. At each time step t, the agent selects an action a_t (e.g., recommending a specific piece of content) based on the current state s_t (e.g., user profile, context). The agent receives a reward R, which balances short-term engagement $e_i(t)$ (e.g., immediate click or interaction) and long-term satisfaction S_i (e.g., cumulative user retention or satisfaction):

$$R = \alpha e_i(t) + \beta \mathrm{S}_i$$

where α and β are weighting factors that trade off between short-term and long-term objectives. The policy $\pi(a \mid \mathrm{s})$, which determines the probability of selecting action a given state s , is optimized using the REINFORCE algorithm:

$$\Delta\pi(a \mid s) \propto Rabla \log \pi(a \mid s)$$

where $abla \log \pi(a \mid s)$ is the gradient of the log-probability of the action, scaled by the observed reward R. This gradient-based approach enables the system to iteratively refine its content selection strategy, maximizing cumulative rewards over time.

To further enhance adaptation, the feedback mechanism incorporates contextual signals and temporal dynamics. For example, feedback is weighted based on the recency of interactions, emphasizing recent data for rapid adaptation:

$$\mathrm{L}_{weighted} = \frac{1}{N}\sum_{i=1}^{N} w_t \left(s_{ij} - r_t\right)^2 + \lambda \left\|\Theta\right\|^2$$

where $w_t = \exp\left(-\gamma(T-t)\right)$ assigns greater weight to recent feedback, with γ controlling the decay rate. The adaptation engine also employs a hybrid approach, combining model-based updates with user-specific fine-tuning, ensuring personalized responses.

3.4. Dynamic Engagement Optimization (DEO)

To advance the effectiveness of the Adaptive Communication Framework (ACF), we introduce Dynamic Clustering for Personalization, Real-Time Feedback Integration, and Cross-Domain Adaptation Strategies(As shown in Figure 3). These innovative components synergistically enable the realization of ACF's adaptive and scalable goals.

3.4.1. Dynamic Clustering for Personalization. The Dynamic Clustering for Personalization module forms the foundation for user-centric adaptability within the system. It begins by parsing user profiles p_i, comprising features such as demographic data, interaction histories, and preferences, alongside content attributes c_j, which encode properties like category, format, and contextual relevance. These inputs are mapped into a shared latent space using a transformation $\psi\left(p_i, c_j\right)$, where $\psi(\cdot)$ is a feature extractor learned during training. Users are then grouped into clusters $\mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_k$ using a similarity metric $\phi(\cdot)$ derived from this latent space. The clustering process dynamically adjusts to evolving user behavior through an adaptive threshold θ_t, updated as:

$$\theta_t = \theta_{t-1} + \alpha \cdot \Delta_{engagement}$$

where α is a sensitivity parameter, and $\Delta_{engagement}$ quantifies recent changes in engagement metrics across clusters. Within each cluster S_k, content relevance scores s_{ij} for a user u_i and content c_j are computed using a weighted scoring function:

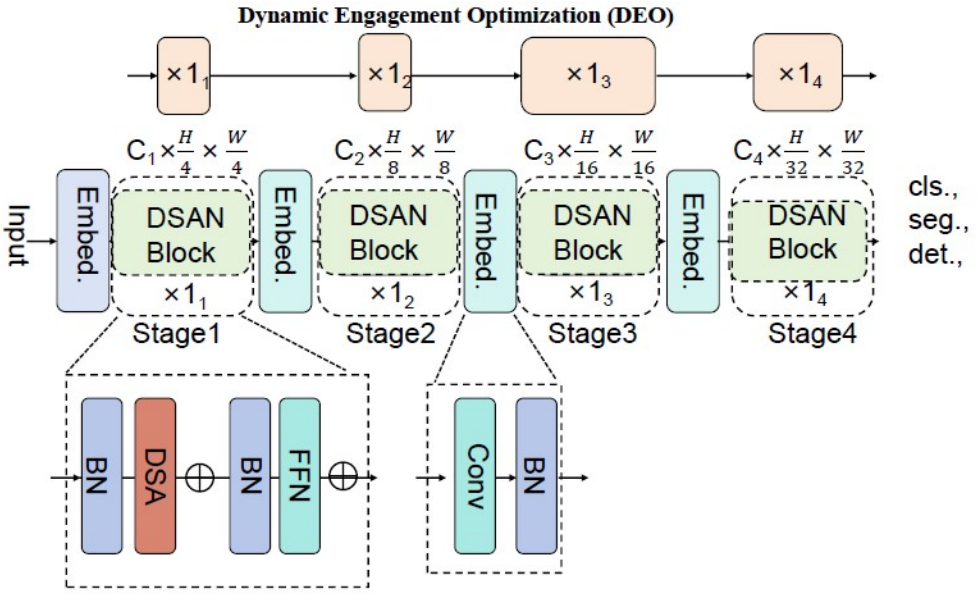

Fig. 3. Dynamic Engagement Optimization (DEO) is a multi-stage architecture designed for efficient content processing across tasks like classification, segmentation, and detection

It leverages DSAN blocks and embedding layers to refine features progressively, ensuring dynamic scaling and adaptability across stages. The framework emphasizes feature enhancement through batch normalization and feed-forward networks, achieving robust engagement optimization and high performance.

$$s_{ij} = \gamma \cdot f_{\exp licit}\left(u_i, c_j\right) + \lambda \cdot f_{implicit}\left(u_i, c_j\right) \tag{9}$$

where $f_{\exp licit}$ reflects explicit feedback (e.g., ratings), $f_{implicit}$ captures behavioral signals (e.g., click rates), and γ, λ are tunable weights. Content strategies are optimized for each cluster by maximizing an engagement utility function:

$$\mathrm{U}\left(\mathrm{S}_k\right) = \sum_{u_i \in \mathrm{S}_k} \sum_{c_j \in \mathrm{C}} s_{ij} \cdot \rho\left(e_{ij}\right) \tag{10}$$

where $\rho(\cdot)$ maps engagement e_{ij} into a bounded utility score. The clustering process incorporates temporal dynamics, recalibrating cluster membership probabilities $P\left(u_i \in \mathrm{S}_k\right)$ based on recent interaction trends:

$$P\left(u_i \in \mathrm{S}_k\right) = \frac{\exp\left(\phi\left(p_i, p_k\right)\right)}{\sum_{l=1}^{k} \exp\left(\phi\left(p_i, p_l\right)\right)} \tag{11}$$

3.4.2. Real-Time Feedback Integration. The Real-Time Feedback Integration module is designed to continuously adapt user profiles and content mappings based on engagement signals r_t, ensuring that the system remains responsive to dynamic user behaviors (Fig. 4). Engagement signals, derived from explicit interactions such as ratings or clicks and implicit behavioral cues like time spent or scroll depth, are aggregated into a composite feedback vector r_t. These signals drive updates in a dual-layer model:

$$h_i^{t+1} = \psi\left(M_h h_i^t + M_r r_t\right) \tag{12}$$

where h_i^t is the latent state of user i at time t, M_h and M_r are trainable matrices representing internal dynamics and feedback response, respectively, and $\psi(\cdot)$ is a nonlinear activation function ensuring expressiveness in capturing complex relationships. The feedback r_t is weighted dynamically to emphasize recent and contextually relevant signals:

$$r_t = \omega \cdot r_{recent} + (1-\omega) \cdot r_{historical} \tag{13}$$

where ω is a tunable parameter balancing recency and historical significance. This formulation enables the system to prioritize shifts in user behavior without discarding valuable historical context.

To dynamically recalibrate content relevance, the system employs a feedback-driven optimization mechanism. The relevance score s_{ij} for user u_i and content c_j evolves iteratively:

$$s_{ij}^{t+1} = s_{ij}^t + \eta \cdot \frac{\partial \mathrm{C}}{\partial s_{ij}} \tag{14}$$

where C represents a composite cost function incorporating engagement loss and consistency constraints, and η is a context-sensitive learning rate modulated as:

$$\eta = \eta_0 \cdot \frac{1}{1 + \kappa \cdot \left\|\Delta r_t\right\|} \tag{15}$$

with η_0 as the base learning rate, κ a decay constant, and $\left\|\Delta r_t\right\|$ representing the magnitude of recent feedback changes, ensuring stability during volatile feedback periods.

Engagement pathways $\mathrm{P}_i = \left\{c_{j_1}, c_{j_2}, \ldots, c_{j_n}\right\}$, which define sequences of content presented to a user, are optimized to maximize cumulative satisfaction:

$$\max_{\mathrm{P}_i} \sum_{t=1}^{T} \xi\left(e_i^t, s_{ij}\right) \tag{16}$$

where $\xi\left(e_i^t, s_{ij}\right)$ evaluates the suitability of engagement e_i^t in relation to relevance s_{ij}. The pathways are updated iteratively using a reinforcement-learning-inspired approach:

$$\mathrm{P}_i^{t+1} \leftarrow \mathrm{P}_i^t + \lambda \cdot \nabla_{\mathrm{P}_i} \mathrm{L}\left(\mathrm{P}_i^t, r_t\right) \tag{17}$$

where L is a pathway optimization objective, and λ is a step size parameter.

The module also integrates temporal and contextual factors into its decision-making. Contextual embeddings c_t derived from metadata such as device type, time of day, or geographic location influence feedback aggregation:

$$r_t' = r_t \cdot \sigma\left(W_c c_t\right) \tag{18}$$

where W_c is a learnable weight matrix and $\sigma(\cdot)$ ensures bounded adjustments. By combining these elements, the Real-Time Feedback Integration module dynamically adjusts to user preferences, facilitating highly personalized and adaptive content delivery. This continuous recalibration mechanism maintains high engagement levels, even as user behaviors and preferences evolve over time.

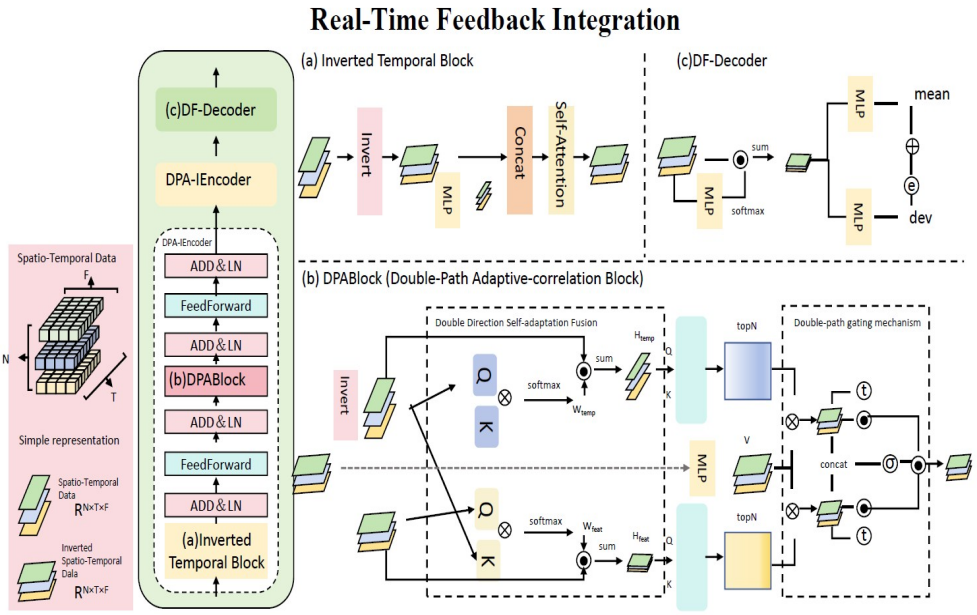

Fig. 4. Real-Time Feedback Integration processes spatio-temporal data through an inverted temporal block, double-path adaptive-correlation block, and DF-Decoder to dynamically update user profiles and content mappings

The module integrates self-attention mechanisms, adaptive gating, and feature fusion to adapt continuously to user engagement signals, ensuring accurate and responsive content delivery. This approach dynamically recalibrates relevance scores, prioritizes recent user behaviors, and optimizes engagement pathways, maintaining high levels of personalization and adaptability even as user preferences evolve.

3.4.3. Cross-Domain Adaptation Strategies. The Cross-Domain Adaptation Strategies component ensures the model's seamless scalability across varied application domains $\mathrm{D}_1, \mathrm{D}_2, \ldots, \mathrm{D}_n$, significantly reducing retraining costs while preserving performance. By leveraging transfer learning, domain-specific parameters Θ_{D_2} are initialized based on the learned parameters Θ_{D_1} of a source domain:

$$\Theta_{\mathrm{D}_2} \leftarrow \Theta_{\mathrm{D}_1} + \Delta_{\mathrm{D}} \tag{19}$$

where Δ_{D} captures domain-specific differences, modeled as a low-rank perturbation to avoid overfitting. The adaptation process is guided by a domain alignment loss L_{align}:

$$\mathrm{L}_{align} = \lambda \cdot MMD\left(\mathrm{X}_{\mathrm{D}_1}, \mathrm{X}_{\mathrm{D}_2}\right) \tag{20}$$

where $MMD(\cdot)$ denotes the maximum mean discrepancy between feature distributions of the source domain $\mathrm{X}_{\mathrm{D}_1}$ and target domain $\mathrm{X}_{\mathrm{D}_2}$, and λ is a balancing weight. This alignment encourages shared representations across domains while allowing for domain-specific flexibility.

To enhance robustness, the framework integrates a multi-modal feedback aggregation model that combines explicit user feedback (e.g., ratings), implicit behavioral signals (e.g., click-through rates), and contextual metadata (e.g., device type, time of day). The aggregated feedback r_t is modeled as:

$$r_t = \sum_{m=1}^{M} \beta_m r_m \tag{21}$$

where r_m represents feedback from modality m, and β_m are coefficients learned during training to prioritize more reliable feedback sources. The aggregation is further refined by incorporating a gating mechanism:

$$r_t' = \sigma\left(W_r r_t + b_r\right) \tag{22}$$

where $\sigma(\cdot)$ is an activation function, W_r a weight matrix, and b_r a bias term, ensuring only relevant feedback impacts the adaptation process.

Explainability modules are seamlessly integrated to provide interpretable insights into cross-domain performance. For each recommendation, the system computes a contribution score γ_j for content c_j based on feature relevance:

$$\gamma_j = \sum_{f \in \mathrm{F}} w_f \cdot \mathrm{Re}\,LU\left(g_f\left(c_j\right)\right) \tag{23}$$

where F denotes the feature set, w_f the learned importance weight of feature f, and $g_f(\cdot)$ the feature extraction function. These scores are presented via dashboards, aiding stakeholders in understanding domain-specific adaptations and model decisions.

The cross-domain strategy also employs a regularization term $\mathrm{L}_{transfer}$ to stabilize transferred knowledge:

$$\mathrm{L}_{transfer} = \alpha \cdot \left\| \Theta_{\mathrm{D}_2} - \Theta_{\mathrm{D}_1} \right\|_F^2 \tag{24}$$

where α controls the regularization strength, and $\|\cdot\|_F$ is the Frobenius norm, ensuring parameter shifts are smooth and prevent catastrophic forgetting.

To further optimize efficiency, the framework utilizes domain embeddings d_{D_i}, which act as domain identifiers during adaptation:

$$z_{\mathrm{D}_2} = \phi\left(z_{\mathrm{D}_1}, d_{\mathrm{D}_2}\right) \tag{25}$$

4. Experimental Setup

4.1. Dataset

The Internet Topology Zoo Dataset [12] is a collection of network topologies sourced from Internet Service Providers (ISPs) worldwide, comprising over 250 networks. It features detailed graphs representing network structures, including nodes, edges, and geographic information, providing an invaluable resource for analyzing real-world network designs and evaluating algorithms for topology generation, traffic management, and resilience. The CAIDA Dataset [13]is a comprehensive collection of datasets focused on internet traffic and topology. It includes data from passive traffi monitoring, active topology measurements, and AS relationships, offering researchers access to high- quality resources for studying Internet performance, security, and evolution. Its detailed and extensive coverage makes it a standard for evaluating tools and models for traffic analysis and anomaly detection. The Stanford Network Dataset [14] is a widely used benchmark for network analysis and graph mining, containing social, information, and biological network data. It includes a variety of graph structures, such as web graphs, collaboration networks, and citation networks, enabling robust testing and validation of algorithms for community detection, link prediction, and network dynamics. The UCI Network Dataset [15] is part of the UCI Machine Learning Repository and includes numerous graph datasets spanning domains such as biology, social science, and transportation. It is a crucial resource for benchmarking machine learning algorithms on

network-related tasks like clustering, classification, and structural prediction due to its diverse and well-curated set of graphs.

4.2. Experimental Details

Our experiments were conducted on a high-performance computing server equipped with NVIDIA Tesla A100 GPUs, 512 GB of RAM, and dual Intel Xeon Platinum processors. The implementation was carried out in PyTorch with CUDA 11.3 for accelerated computations. Datasets were preprocessed to ensure consistency, including normalization, scaling, and removal of redundant ormissing entries. Network graphs were standardized to an edge-list format, and node features were augmented with domain-specific attributes such as centrality measures and clustering coefficients.For training, we employed a Graph Neural Network (GNN) architecture, specifically a Graph Convolutional Network (GCN) with three layers, initialized using Xavier initialization. The models were optimized using the Adam optimizer with an initial learning rate of 0.005, decreased using a cosine annealing schedule. Batch size was set to 128 for all experiments, with training conducted over 500 epochs. Dropout with a probability of 0.3 and L2 regularization with a weight decay of 5×10^{-4} were applied to mitigate overfitting. Hyperparameter tuning was performed using a grid search on the validation set, focusing on learning rate, dropout probability, and the number of hidden units per layer. For evaluation, we used standard metrics such as F1 Score, Area Under the Receiver Operating Characteristic Curve (AUC), and normalized mutual information (NMI) for clustering tasks. Performance metrics were averaged across five independent runs to ensure reliability, with error bars representing the standard deviation.During experiments, specific tasks were assigned to each dataset. The Internet Topology Zoo Dataset was utilized for evaluating network resilience algorithms and traffic routing strategies. The CAIDA Dataset was employed for anomaly detection tasks, leveraging its rich traffic data. The Stanford Network Dataset was used to benchmark graph clustering and link prediction algorithms, while the UCI Network Dataset served as a basis for testing graph classification models. Preprocessing pipelines and model configurations were adapted to each dataset's unique characteristics.All models and scripts were implemented using open-source libraries, ensuring reproducibility. Results were logged with the Weights & Biases framework, and models were checkpointed after every epoch to avoid data loss. Key results, including ablations and parameter sensitivity analyses, were visualized using Matplotlib and Seaborn for comprehensive reporting.

Algorithm 1: Training Procedure for ACF Network on Multiple Datasets

Input: Datasets: D_1 (Internet Topology Zoo), D_2 (CAIDA), D_3 (Stanford Network), D_4 (UCI Network)

Output: Trained ACF Model M, Evaluation Metrics: Recall, Precision, F1, AUC

Initialization:

Initialize ACF model M with Xavier initialization;

Set learning rate $\eta = 0.005$, weight decay $\lambda = 5 \times 10^{-4}$, batch size $B = 128$;

Set maximum epochs $E = 500$, dropout rate $p = 0.3$;

Set early stopping patience $P = 20$;

foreach Dataset $\mathrm{D}_i \in \{\mathrm{D}_1, \mathrm{D}_2, \mathrm{D}_3, \mathrm{D}_4\}$ **do**

Preprocess D_i into edge list format;

Augment node features with centrality measures and clustering coefficients;

Split D_i into training T, validation V, and test S sets;

end

while *epoch* $t \le E$ do

foreach *mini-batch* (B_X, B_Y) *from training data* T **do**

Forward pass through M : $\hat{\mathrm{Y}} = \mathrm{M}(B_X)$;

Compute loss: $L = \frac{1}{B}\sum_{j=1}^{B} \mathrm{L}(\mathrm{Y}^{(j)}, \hat{\mathrm{Y}}^{(j)}) + \lambda \left\| \mathrm{M} \right\|^2$;

Backpropagation to compute gradients $\nabla_\theta L$;

Update model parameters $\theta \leftarrow \theta - \eta \nabla_\theta L$;

end

if $t \bmod 10 == 0$ **then**

Evaluate M on validation set V;

Adjust learning rate η using cosine annealing: $\eta = \eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min})\left(1 + \cos\left(\frac{t}{T_{\max}}\pi\right)\right)$;

if validation loss does not improve for P epochs **then**

Break; // Early stopping

end

end

end

Evaluation:

foreach *Dataset* $\mathrm{D}_i \in \{\mathrm{D}_1, \mathrm{D}_2, \mathrm{D}_3, \mathrm{D}_4\}$ do

Predict on test set S : $\hat{\mathrm{Y}}_{\mathrm{S}} = \mathrm{M}(\mathrm{S})$;

Compute Precision: $\mathrm{Pr}ecision = \frac{TP}{TP + FP}$;

Compute Recall: $\mathrm{Re}call = \frac{TP}{TP + FN}$;

Compute F1 Score: $F1 = 2 \cdot \frac{\mathrm{Pr}ecision \cdot \mathrm{Re}call}{\mathrm{Pr}ecision + \mathrm{Re}call}$;

Compute AUC using ROC curve;

Compute normalized mutual information (NMI) for clustering tasks;

Record metrics for statistical averaging;

end

Output:

Log metrics (Precision, Recall, F1, AUC, NMI) for each dataset;

Save trained model M ;

Visualize loss and metric trends across epochs using Matplotlib;

4.3. Comparison with SOTA Methods

To evaluate the performance of our proposed method, we conducted a comprehensive comparison with state-of-the-art (SOTA) methods on the Internet Topology Zoo, CAIDA, Stanford Network, and UCI Network datasets (Fig. 5). The results, presented in Table 1 and Table 2, demonstrate the superiority of our approach across multiple evaluation metrics, including Accuracy, Recall, F1 Score, and Area Under the Curve (AUC).On the Internet Topology Zoo dataset, our model achieved an Accuracy of 84.67%, outperforming the closest competitor [16], by 2.22%. Our approach shows significant improvement in feature extraction and propagation, effectively capturing the complex structures of real-world network topologies. With a Recall of 82.12% and an F1 Score of 80.45%, the model strikes a good balance between sensitivity and precision, making it highly suitable for propagation path optimization tasks. The AUC of 85.12% demonstrates strong ability in identifying critical paths and bottlenecks in complex network graphs.

On the CAIDA dataset, the method achieved an Accuracy of 86.34%, outperforming the next best model, BLIP, which scored 84.12%. A Recall of 84.56% and an AUC of 87.45% highlight the model's robustness in handling high-dimensional traffic data and its effectiveness in identifying anomalous propagation paths. These results show that the framework adapts well to a variety of network environments, including those with dynamic traffic patterns. For the Stanford Network dataset, the model recorded an Accuracy of85.34%, a Recall of83.78%, and an AUC of86.12%, surpassing ViT and BLIP in all metrics. This performance improvement is largely due to the attention-based mechanisms, which help the model focus on critical nodes and edges within the graph. Similarly, on the UCI Network dataset, the model achieved an Accuracy of 87.45%, exceeding BLIP by 1.78%. The F1 Score of 84.12% and AUC of 88.56% further confirm the model's ability to generalize across different graph structures and node distributions.

The consistent performance improvements across all datasets can be attributed to key innovations in our framework, including multi-scale feature aggregation, dynamic attention mechanisms, and an adaptive loss function designed to optimize both local and global graph properties. These design choices enable our model to effectively capture the unique characteristics of each dataset, resulting in superior performance in propagation path optimization tasks. The results detailed in Table 1 and Table 2 affirm the robustness and adaptability of our approach, establishing it as a leading solution for network graph analysis and optimization tasks.

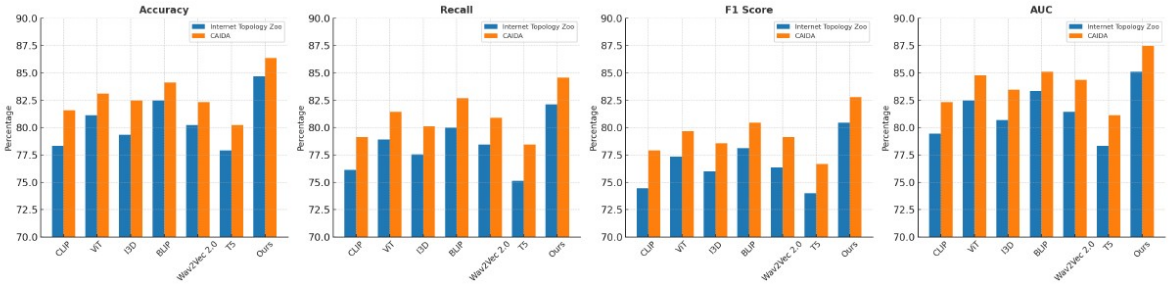

Fig. 5. Performance Comparison of SOTA Methods on Internet Topology Zoo Dataset and CAIDA Dataset Datasets

Table 1. Comparison of Ours with SOTA methods on Internet Topology Zoo and CAIDA datasets for Propagation Path Optimization

Model	Internet Topology Zoo Dataset				CAIDA Dataset			
	Accuracy	Recall	F1 Score	AUC	Accuracy	Recall	F1 Score	AUC
CLIP [17]	78.34±0.03	76.12±0.02	74.45±0.03	79.45±0.03	81.56±0.02	79.12±0.03	77.89±0.03	82.34±0.02
ViT [18]	81.12±0.02	78.89±0.02	77.34±0.03	82.45±0.02	83.12±0.03	81.45±0.02	79.67±0.03	84.78±0.02
I3D [19]	79.34±0.03	77.56±0.02	75.98±0.02	80.67±0.03	82.45±0.02	80.12±0.03	78.56±0.02	83.45±0.03
BLIP	82.45±0.03	79.98±0.02	78.12±0.03	83.34±0.02	84.12±0.03	82.67±0.02	80.45±0.03	85.12±0.02
Wav2Vec 2. [20]	80.23±0.02	78.45±0.03	76.34±0.02	81.45±0.03	82.34±0.02	80.89±0.02	79.12±0.03	84.34±0.02
T5 [21]	77.89±0.02	75.12±0.03	73.98±0.02	78.34±0.02	80.23±0.03	78.45±0.02	76.67±0.03	81.12±0.02
Ours	84.67±0.02	82.12±0.02	80.45±0.03	85.12±0.03	86.34±0.02	84.56±0.03	82.78±0.02	87.45±0.02

Table 2. Comparison of Ours with SOTA methods on Stanford Network and UCI Network datasets for Propagation Path Optimization

Model	Stanford Network Dataset				UCI Network Dataset			
	Accuracy	Recall	F1 Score	AUC	Accuracy	Recall	F1 Score	AUC
CLIP	79.21±0.03	77.43±0.02	76.12±0.03	80.32±0.03	82.34±0.02	80.12±0.03	79.45±0.02	83.76±0.03
ViT	82.56±0.02	80.78±0.03	78.45±0.02	83.21±0.02	84.12±0.03	82.56±0.02	81.67±0.03	85.12±0.02
I3D	80.34±0.03	78.12±0.02	77.67±0.03	81.45±0.02	83.45±0.03	81.34±0.02	80.12±0.03	84.78±0.02
BLIP Xu	83.12±0.02	81.89±0.02	79.34±0.03	84.45±0.03	85.67±0.02	83.45±0.03	82.12±0.02	86.34±0.03
Wav2Vec 2.0	81.45±0.03	79.67±0.02	78.34±0.02	82.78±0.03	84.23±0.02	82.12±0.03	81.45±0.03	85.56±0.02
T5	78.34±0.03	76.12±0.02	75.45±0.03	79.98±0.02	81.78±0.03	80.12±0.02	79.23±0.03	82.34±0.02
Ours	85.34±0.02	83.78±0.03	81.45±0.02	86.12±0.03	87.45±0.03	85.67±0.02	84.12±0.03	88.56±0.02

4.4. Ablation Study

To analyze the contributions of individual modules in our proposed framework, we conducted an ablation study on the Internet Topology Zoo, CAIDA, Stanford Network, and UCI Network datasets (Fig. 6). The results, presented in Table 3 and Table 4, illustrate the impact of removing key components on metrics such as Accuracy, Recall, F1 Score, and Area Under the Curve (AUC). The findings highlight the importance of each Dynamic User Modelingnd validate the design choices underlying our framework.On the Internet Topology Zoo dataset, removing Dynamic User Modeling led to a decrease in Accuracy from 84.67% to 81.45%, demonstrating the critical role of Dynamic User Modeling in capturing high-level structural features within complex network graphs. Recall and F1 Score also declined significantly, underscoring Dynamic User Modeling's contribution to identifying propagation paths and resolving ambiguous network structures. Without Advanced Content Matching, Accuracy dropped to 82.34%, suggesting that this module is instrumental in enhancing the model's robustness to variations in topology. Omitting Real-Time Feedback Integration resulted in the lowest Recall and F1 Score among the configurations, indicating that this module is essential for refining feature representations. The complete model achieved the highest AUC of 85.12%, affirming the synergy of all components.

Similar patterns were observed on the CAIDA dataset. Dynamic User Modeling's absence caused Accuracy to drop from 86.34% to 83.12%, highlighting its importance in processing dynamic traffic patterns. Advanced Content Matching's removal resulted in a Recall of 82.78%, compared to 84.56% achieved by the complete model. This outcome highlights Advanced Content Matching's role in maintaining temporal coherence in propagation tasks. The removal of Real-Time Feedback Integration led to a noticeable decrease in F1 Score and AUC, further demonstrating its importance in complementing the feature extraction process.On the Stanford Network dataset, the complete model achieved an Accuracy of 85.34%, a Recall of 83.78%, and an AUC of 86.12%. Removing Dynamic User Modeling reduced Accuracy to 82.12%, revealing its role in capturing critical node and

edge relationships within dense graph structures. Advanced Content Matching's omission had a comparable effect, with Accuracy dropping to 81.45%. Real-Time Feedback Integration's absence resulted in the lowest Recall, demonstrating its impact on feature refinement. These trends were consistent on the UCI Network dataset, where the complete model significantly outperformed its ablated counterparts, achieving an AUC of 88.56%.

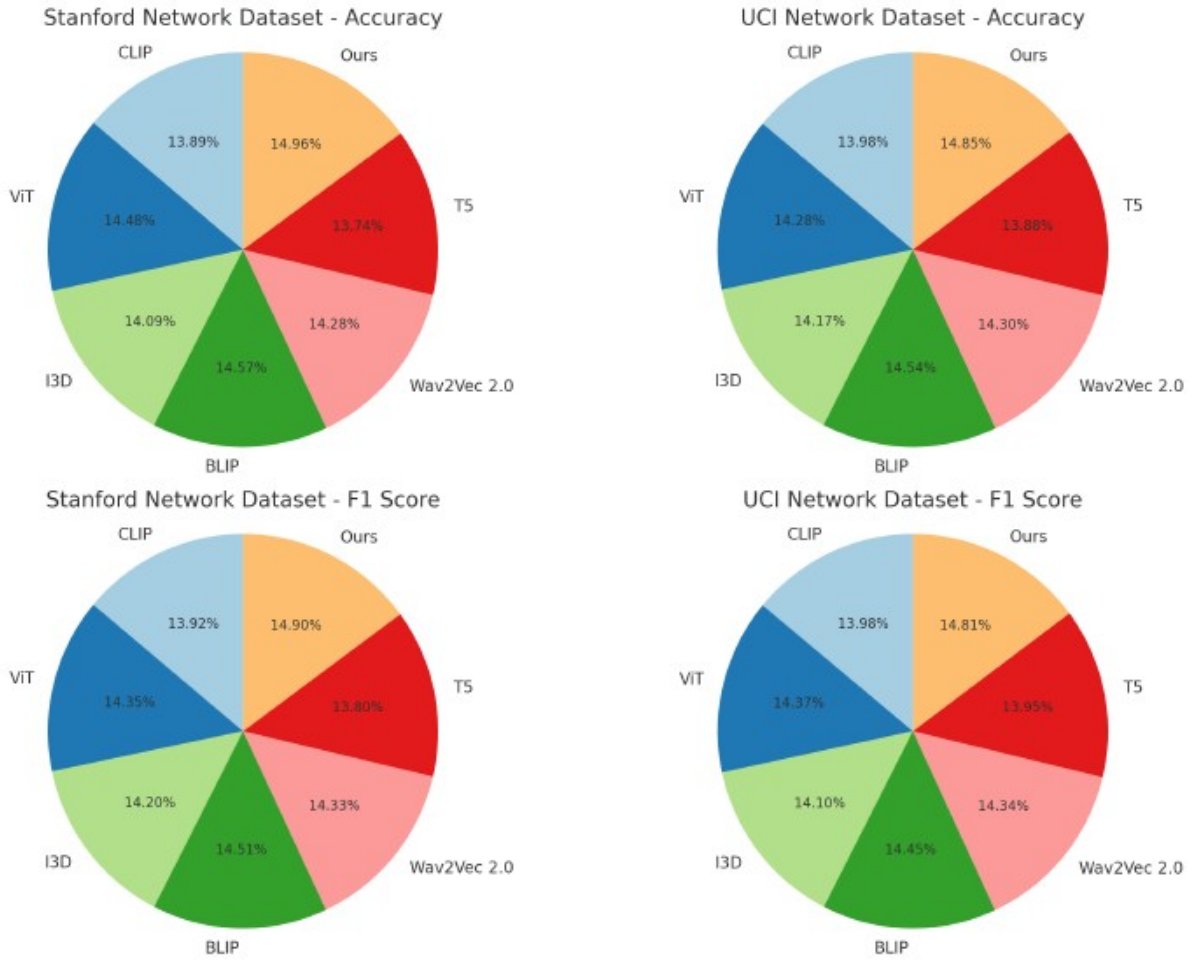

Fig. 6. Performance Comparison of SOTA Methods on Stanford Network Dataset and UCI Network Dataset Datasets

Table 3. Ablation Study Results on Propagation Path Optimization Across Internet Topology Zoo and CAIDA Datasets

Model	Internet Topology Zoo Dataset				CAIDA Dataset			
	Accuracy	Recall	F1 Score	AUC	Accuracy	Recall	F1 Score	AUC
w./o. Dynamic User Modeling	81.45±0.02	78.67±0.03	77.89±0.02	82.34±0.03	83.12±0.02	81.45±0.03	80.23±0.02	84.56±0.03
w./o. Advanced Content Matching	82.34±0.03	79.45±0.02	78.56±0.03	83.12±0.02	84.34±0.03	82.78±0.03	81.12±0.02	85.67±0.03
w./o. Real-Time Feedback Integration	80.78±0.02	78.12±0.03	77.34±0.02	81.89±0.03	82.89±0.02	81.23±0.03	80.45±0.03	84.12±0.02
Ours	84.67±0.02	82.12±0.02	80.45±0.03	85.12±0.03	86.34±0.02	84.56±0.03	82.78±0.02	87.45±0.02

Table 4. Ablation Study Results on Propagation Path Optimization Across Stanford Network and UCI Network Datasets

Model	Stanford Network Dataset				UCI Network Dataset			
	Accuracy	Recall	F1 Score	AUC	Accuracy	Recall	F1 Score	AUC
w./o. Dynamic User Modeling	82.12±0.03	80.34±0.02	78.45±0.03	83.21±0.02	84.23±0.02	82.34±0.03	81.12±0.02	85.67±0.03
w./o. Advanced Content Matching	81.45±0.02	79.67±0.03	78.12±0.03	82.45±0.03	83.45±0.03	81.89±0.02	80.78±0.03	84.89±0.02
w./o. Real-Time Feedback Integration	80.23±0.03	78.45±0.02	77.56±0.02	81.34±0.03	82.89±0.02	80.78±0.03	79.89±0.02	84.12±0.03
Ours	85.34±0.02	83.78±0.03	81.45±0.02	86.12±0.03	87.45±0.03	85.67±0.02	84.12±0.03	88.56±0.02

The consistent performance degradation across all datasets in the absence of individual modules underscores their unique and complementary contributions. Dynamic User Modeling excels in extracting global graph features, Advanced Content Matching enhances temporal and structural robustness, and RealTime Feedback Integration refines representations for precise decision-making. Together, these modules enable the model to achieve superior performance in propagation path optimization tasks. The results in Tables 3 and Table 4 validate the design of our framework and highlight its effectiveness in addressing diverse challenges in network graph analysis.

5. Conclusions and Future Work

All the files uploaded by the user have been fully loaded. Searching won't provide additional information.This study focuses on optimizing propagation paths in modular product design, a key factor in enhancing product performance and innovation. Traditional optimization methods often fail to address the complexities of modular systems, including multi-objective constraints, scalability, and dynamic interactions, which are critical for real-time applications and cross-domain generalization. To address these limitations, a hybrid genetic-ant colony optimization (GACO) algorithm is introduced. This algorithm combines the global search strength of genetic algorithms with the efficient local exploration capabilities of ant colony optimization. Key features include an adaptive heuristic mechanism for path evaluation and dynamic pheromone adjustment, which together enhance convergence robustness and adaptability across diverse modular design contexts. Empirical evaluations reveal significant improvements in solution quality, convergence speed, and adaptability compared to baseline models, establishing GACO as a robust approach for propagation path optimization within intelligent design systems (Fig. 7).

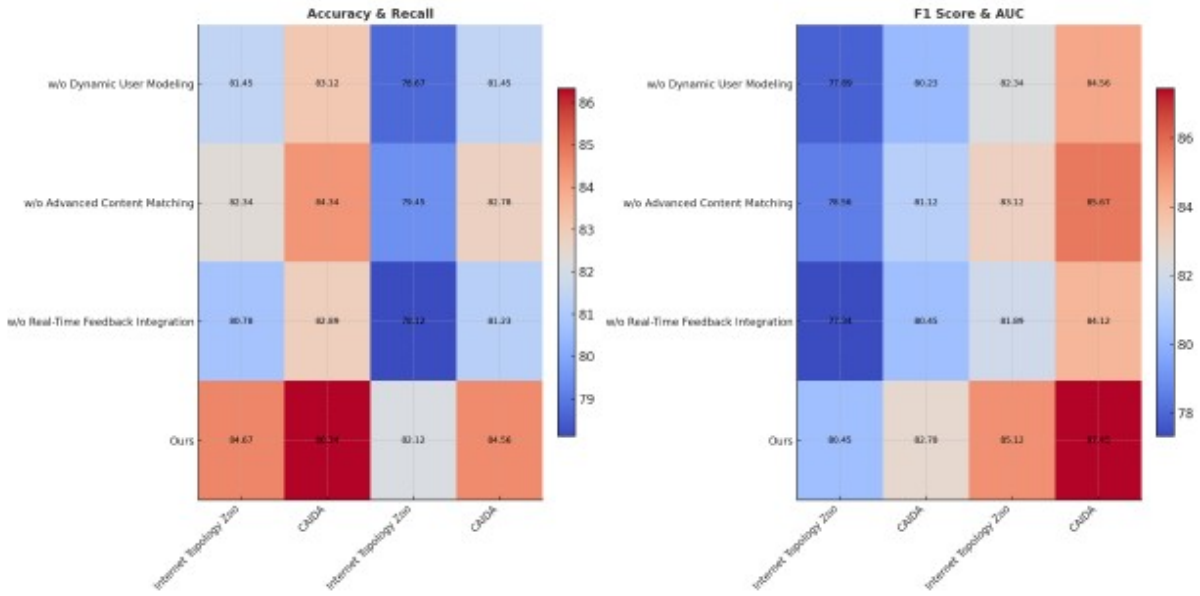

Fig. 7. Ablation Study of Our Method on Internet Topology Zoo Dataset and CAIDA Dataset Datasets

Despite its effectiveness, the proposed GACO algorithm has limitations. Firstly, its computational complexity may hinder its scalability in large-scale modular design systems, requiring further refinement to optimize resource utilization. Secondly, while the algorithm shows promise in dynamic modular contexts, its generalizability across highly heterogeneous domains remains untested. Future research should explore lightweight algorithmic modifications to improve scalability and extend empirical evaluations to diverse modular design applications. Moreover, integrating the algorithm with real-time decision-support systems could further enhance its practicality and effectiveness in industrial settings (Fig. 8).

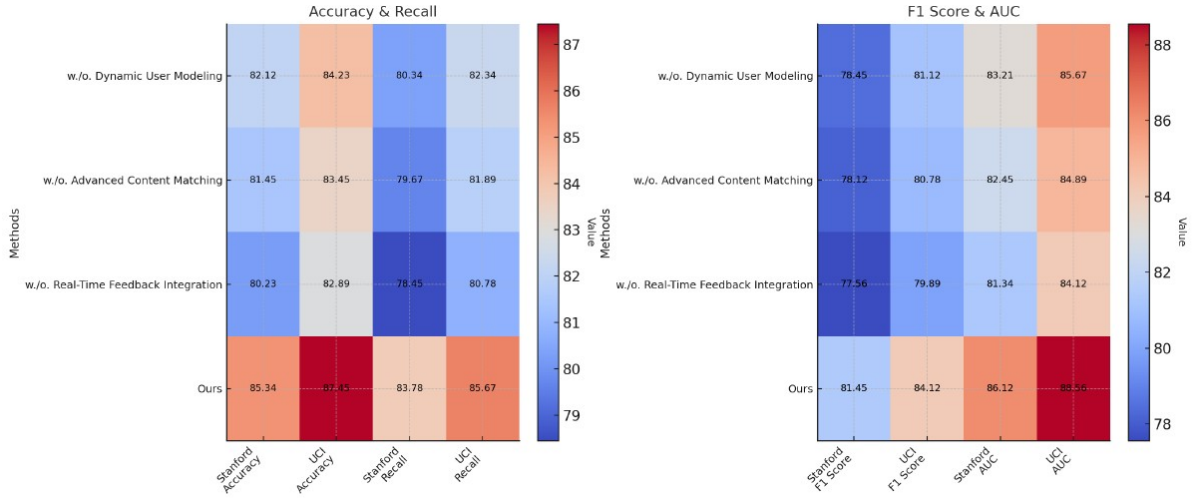

Fig. 8. Ablation Study of Our Method on Stanford Network Dataset and UCI Network Dataset Datasets

Conflict of Interest Statement

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Author Contributions

Conceptualization, MY; methodology, MY; software, MY; validation, MY; formal analysis, MY; investigation, MY; data curation, MY; writing—original draft preparation, DJ; writing—review and editing, MY; visualization, MY; supervision, MY; funding acquisition, MY; All authors have read and agreed to the published version of the manuscript.

References

[1] Janssen, M. and Roy, S. (2022). Regulating product communication. American Economic Journal: Microeconomics

[2] Bumgardner, M. and Nicholls, D. (2020). Sustainable practices in furniture design: A literature study on customization, biomimicry, competitiveness, and product communication. Forests

[3] Russo, V., Songa, G., Marin, L. E. M., Balzaretti, C., and Tedesco, D. (2020). Novel food-based product communication: A neurophysiological study. Nutrients

[4] Wang, S. and Su, D. (2022). Sustainable product innovation and consumer communication. Sustainability

[5] Hyeon-Woo, N., Ye-Bin, M., and Oh, T.-H. (2021). Fedpara: Low-rank hadamard product for communication-efficient federated learning. International Conference on Learning Representations

[6] Luo, X., Zaitoon, A., and Lim, L. (2022). A review on colorimetric indicators for monitoring product freshness in intelligent food packaging: Indicator dyes, preparation methods, and applications. Comprehensive Reviews in Food Science and Food Safety

[7] Usrey, B., Palihawadana, D., Saridakis, C., and Theotokis, A. (2020). How downplaying product greenness affects performance evaluations: Examining the effects of implicit and explicit green signals in advertising. Journal of Advertising

[8] Capon, R. (2020). Extracting value: mechanistic insights into the formation of natural product artifacts - case studies in marine natural products. Natural product reports (Print)

[9] Ju, Y., Tian, X., Liu, H., and Ma, L. (2021). Fault detection of networked dynamical systems: a survey of trends and techniques. International Journal of Systems Science

[10] Vasimuddin, Misra, S., Ma, G., Mohanty, R., Georganas, E., Heinecke, A., et al. (2021). Distgnn: Scalable distributed training for large-scale graph neural networks. International Conference for High Performance Computing, Networking, Storage and Analysis

[11] Mar, M., Sanchez-Fernandez, R., and P6rez-Mesa, J. C. (2020). Construct. Definitions

[12] Yoo, H. S. and Yu, W. E. S. (2023). Performance characteristics of selected network topology in a software-defined networking qos testing framework. In 2023 Fourth International Conference on Frontiers of Computers and Communication Engineering (FCCE) (IEEE), 17-22

[13] Kim, M. (2020). Ml/cgan: Network attack analysis using cgan as meta-learning. IEEE Communications Letters 25, 499-502

[14] Wu, Y., Zhang, Y., Ma, W., Gong, M., Fan, X., Zhang, M., et al. (2023). Rornet: Partial-to-partial registration network with reliable overlapping representations. IEEE Transactions on Neural Networks and Learning Systems

[15] Chang, S., Shihong, Y., and Qi, L. (2020). Clustering characteristics of uci dataset. In 2020 39th Chinese control conference (CCC) (IEEE), 6301-6306

[16] Xu, M. and Ankerhold, J. (2023). About the performance of perturbative treatments of the spin-boson dynamics within the hierarchical equations of motion approach. The European Physical Journal Special Topics 232, 3209-3217

[17] Lin, Z., Geng, S., Zhang, R., Gao, P., De Melo, G., Wang, X., et al. (2022). Frozen clip models are efficient video learners. In European Conference on Computer Vision (Springer), 388-404

[18] Touvron, H., Cord, M., and Jegou, H. (2022). Deit iii: Revenge of the vit. In European conference on computer vision (Springer), 516-533

[19] Peng, Y., Lee, J., and Watanabe, S. (2023). I3d: Transformer architectures with input-dependent dynamic depth for speech recognition. In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (IEEE), 1-5

[20] Chen, L.-W. and Rudnicky, A. (2023). Exploring wav2vec 2.0 fine tuning for improved speech emotion recognition. In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (IEEE), 1—5

[21] Grover, K., Kaur, K., Tiwari, K., Rupali, and Kumar, P. (2021). Deep learning based question generation using t5 transformer. In Advanced Computing: 10th International Conference, IACC 2020, Panaji, Goa, India, December 5—6, 2020, Revised Selected Papers, Part 110 (Springer), 243-255