Universiteit
Leiden
The Netherlands

Bachelor Computer Science & Datascience and Artificial Intelligence

Modeling social creativity with large-scale multi-agent systems and emergent properties of collective creative processes

Luuk Motz

Rob Saunders

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

15/01/2025

Abstract

This thesis explores the modeling of social creativity using large-scale multi-agent systems, advancing prior work by scaling simulations to hundreds of agents to examine emergent phenomena. Existing computational models have provided valuable insights into creative social behaviors within small agent populations but often relied on artificial constraints to observe these behaviors. In this study, we address both technical and theoretical challenges associated with scaling, introducing architectural innovations to enable efficient large-scale interactions, such as combining a kNN and ResNet for novelty detection. Through systematic experimentation, we reveal how emergent properties such as clique formation, stylistic evolution, and knowledge transfer manifest naturally in larger systems. The results demonstrate that scaling not only uncovers new patterns of collective creativity but also provides a deeper understanding of the self-organizing dynamics that drive creative communities. These findings contribute to the development of scalable computational creativity frameworks and offer insights into the mechanisms underlying creativity in both artificial and human systems. We find that although large-scale multi-agent simulations reveal more complex social structures compared to small-scale approaches, fully stable clique formation does not persist, indicating that truly emergent creativity at scale still faces key computational and design challenges.

Contents

1 Introduction

Social creativity is the collective process of idea generation and innovation within groups or societies and has become an increasingly important area of study in both cognitive science and artificial intelligence [1]–[3]. While traditional research has often focused on creativity as an individual cognitive process [4], there is a growing recognition that creativity is fundamentally shaped by social and cultural dynamics [5].

The computational modeling of social creativity has historically been constrained to small-scale simulations, typically involving fewer than twenty agents [6]. While these studies have provided valuable insights, these studies often required artificial constraints to demonstrate phenomena like clique formation. Multi-agent systems now offer a promising framework for modeling these complex social creative processes on larger scales [7]. By scaling simulations to thousands of agents, each with their own creative capabilities and decision-making processes, we can study emergent patterns of collective creativity that more naturally mirror real-world creative processes [8], [9].

While scaling the multi-agent simulations to thousands or even more agents, problems arise on a technical and theoretical level. From a technical perspective, the computational demands of modeling thousands of agents, each capable of generating and evaluating creative artifacts, require careful consideration of architecture and implementation [8]. The theoretical obstacle lies in determining the appropriate mechanisms for agent interaction, evaluation, and learning that can meaningfully capture the dynamics of creative communities in the real world at scale [10].

Previous research has identified several key phenomena in creative social systems, such as the formation of aesthetic niches or the inception of creative leadership [2], [6]. These studies have revealed how creative communities naturally organize themselves into distinct groups [2], [3]. However, these findings have largely emerged from small-scale simulations where researchers needed to carefully construct the initial conditions of the agents and the domain. Large-scale simulations show potential to study how these phenomena may emerge naturally from more realistic starting conditions and may reveal entirely new patterns of creative social organization [8], [9].

1.1 Motivation

This research is motivated by two key developments in computational creativity. First, while existing models like the Digital Clockwork Muse have demonstrated creative social behaviors in small agent populations [11], they required artificial constraints to produce phenomena like clique formation. This raises questions about whether such properties would emerge naturally in larger-scale simulations that better approximate real creative communities.

Recent advances in computational capabilities have made it possible to model complex social systems at unprecedented scales. As shown by Wei, Tay, Bommasani, *et al.*, certain behavioral properties only become observable when systems are scaled significantly [12]. Applied to social creativity, large-scale simulations could reveal emergent phenomena undetectable in smaller implementations. The interaction patterns and social structures that emerge at scale may provide insights into how creative communities self-organize.

Furthermore, understanding large-scale creative processes has practical implications. While

individual creative agents have been extensively studied, the collective dynamics of large creative communities remain poorly understood. By investigating how thousands of agents interact and influence each other's creative development, we can better understand the social mechanisms driving innovation in human creative communities. This knowledge could inform both the design of future computational creativity systems and our understanding of creativity in large-scale human organizations.

This research therefore aims to bridge the gap between small-scale proof-of-concept models and the complexity of real-world creative communities. By developing scalable frameworks for modeling social creativity, we can examine how emergent properties arise naturally from agent interactions rather than through artificially imposed constraints.

1.2 Research Aims and Question

The transition from small-scale proof-of-concept models to large-scale simulations of social creativity presents both theoretical and technical challenges that this research aims to address. While previous work has demonstrated the feasibility of modeling creative social behaviors in limited agent populations [6], the effects of scaling these systems to thousands of agents remain unexplored. This research investigates whether phenomena that required artificial constraints in small-scale implementations emerge naturally at larger scales.

1.2.1 Research Question

This study addresses the following primary research question: **How can larger multi-agent systems be used to model social creativity, and what emergent properties arise across different scales that might inform our understanding of collective creative processes and computational creativity models?**

1.2.2 Hypotheses

Based on theoretical foundations and preliminary investigations, we propose the following hypotheses:

1. **H1:** Large-scale multi-agent systems (>500 agents) will demonstrate emergent creative behaviors, particularly clique formation and stylistic evolution, without requiring the artificial constraints necessary in smaller implementations.

2. **H2:** Scaling effects will reveal qualitatively different patterns of creative behavior and social organization that are unobservable in small-scale simulations (<20 agents), without any artificial constraints.

These hypotheses address both the technical feasibility of scaling social creativity models and the potential emergence of new behavioral patterns. By investigating these hypotheses,we aim to broaden our understanding of collective creative processes and contribute to the development of more sophisticated computational creativity models [13]. The results could provide insights into how creative communities self-organize and evolve, with potential applications across various domains of computational creativity research. The systematic investigation

of these hypotheses requires careful consideration of both computational architecture and experimental design, which will be detailed in subsequent chapters. Through this investigation, we aim to bridge the gap between small-scale proof-of-concept models and the complexity of real-world creative communities [14].

1.3 Thesis overview

This thesis is structured as follows: Chapter 2 presents a comprehensive review of the literature on computational social creativity, multi-agent systems, and emergent phenomena. Chapter 3 details the methodology and technical implementation of our large-scale multi-agent framework. Chapter 4 presents the experimental setup where we conduct the simulations. Chapter 5 displays the results from the experiments. Chapter 6 and 7 discuss and conclude the thesis.

2 Related Work

2.1 History of Social Creativity

2.1.1 Understanding Creativity

The general definition of creativity is mainly associated with the ability to produce something new and unexpected. For something to be classified as creative, the artefact that was produced must be recognizable and novel. The exact process and emergence of creativity is still unknown [15]. To frame creativity within a computational and theoretical context, it is essential to address its dual nature. Creativity encompasses both exploration and exploitation [10]. Exploration involves the generation of novel ideas or artifacts, while exploitation focuses on refining and optimizing existing ones. Both components are not independent; they exist in a dynamic interplay, where exploration feeds into exploitation, and vice versa. This duality is fundamental to understanding how creativity operates within both individuals and groups.

2.1.2 Reframing Creative Processes

The study of social creativity emerged as a counterpoint to the romanticized notion of the lone creative genius, recognizing creativity as an inherently collective phenomenon that arises from complex social interactions [16]. Csikszentmihalyi advanced this understanding by highlighting how creativity emerges from the dynamic relationship between individual cognitive processes and broader social-cultural contexts [17]. He challenged the prevailing computational models of creativity that reduced it to mere problem-solving processes. These earlier models assumed that given sufficient domain knowledge and adequate heuristic search procedures, creative breakthroughs were simply a matter of time and computational power [17].

This critique of reductionist approaches has profound implications for computational modeling of creative processes. Rather than focusing solely on problem-solving mechanisms, effective models must account for what Csikszentmihalyi termed the "non-rational components" of creativity: the initial spark of interest that draws individuals to particular domains, their perseverance in pushing that domains boundaries, their willingness to question established frameworks and crucially, the social context that either nurtures or inhibits that creative development [17]. This suggest that multi-agent systems designed to study social creativity must incorporate mechanisms beyond simple problem-solving heuristics to capture the complex dynamics of collective creative processes.

2.1.3 The Domain-Individual-Field-Interaction (DIFI) Framework

The Domain-Individual-Field-Interaction (DIFI) presents the core foundation upon which this thesis is based. The idea of creativity emerges from the dynamic interaction between these three subsystems: domain, individual, and field, which fundamentally challenges the notion of creativity as a purely individual cognitive process [18]. The domain represents the corpus of knowledge accumulated and cultural artifacts within a creative field, produced by the individual. The individual draws upon the domain knowledge to produce novel works. The field, in Csikszentmihalyi original conception, functions as a selective mechanism

determining which works merit inclusion in the domain [18]. This selectivity operates through "gatekeepers"; individuals who occupy positions of influence and evaluation within their respective fields. For instance, in scientific domains, these gatekeepers might be journal editors or distinguished researchers whose judgments significantly influence what becomes accepted into the domain's body of knowledge.

This cyclical process creates a self-reinforcing system where creative development occurs through continuous interaction rather than isolated innovation. The domain provides the raw material and context that individuals need to create meaningfully within their field. Individuals then transform this knowledge in novel ways, producing variations that must survive evaluation by the field before potentially entering back into the domain. As Csikszentmihalyi argues, creativity emerges from this complete system of interactions rather than residing solely in any single component [18].

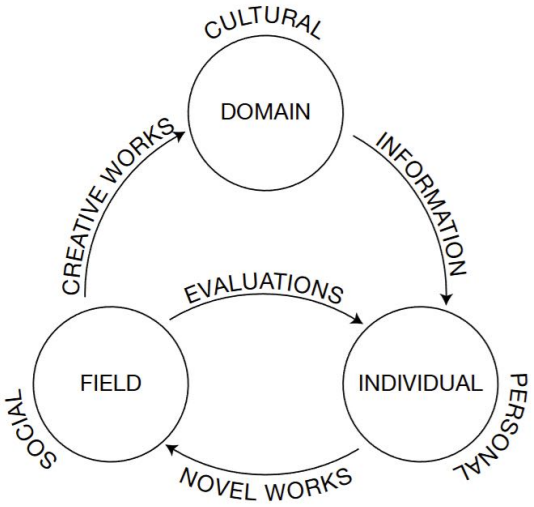


Figure 1: Csikszentmihalyi's Systems View of Creativity illustrates the dynamic interaction between domain, individual, and field in creative processes, adapted from Csikszentmihalyi [17] by Saunders [7]

.

The DIFI framework's emphasis on structured social interaction makes it particularly suitable for computational modeling through multi-agent systems. Each component can be explicitly represented: the domain as a repository of artifacts, individuals as agents capable of generating and evaluating these works, and the field as the collective evaluation mechanisms between agents. This direct mapping between theoretical framework and computational implementation provides a robust foundation for studying creativity through simulation and potentially at scale [8], [19], [20].

2.1.4 Computational Frameworks and Scaling Challenges

Computational frameworks for modeling social creativity have historically been constrained by the inherent complexity of simulating large-scale creative interactions. Early implementations, such as Saunders' Digital Clockwork Muse, demonstrated the feasibility of modeling creative

social behaviors but were limited to small agent populations of 10-20 individuals [11]. These constraints, while necessary for initial proof-of-concept work, significantly simplified the potential real life dynamics in these creative societies.

As technology progresses, the power of computation has increased significantly, allowing these large-scale models to run at previously unattainable scales. However, scaling these models introduces fundamental technical and theoretical challenges. As Wei, Tay, Bommasani, *et al.* demonstrate in their work on emergent behaviors in large-scale systems, capabilities and phenomena that are absent or invisible at smaller scales can emerge when systems are scaled up significantly [12]. This suggests that scaling social creativity models may reveal emergent properties that were unobservable in smaller implementations [12].

The technical challenges of scaling creative agent systems extend beyond mere computational resources. Each agent must maintain its own creative decision-making processes, memory of previous interactions, and evaluative mechanisms for assessing novelty. When scaled to thousands of agents, the resource requirements for maintaining individual learning models become prohibitive without careful architectural considerations and changes.

2.2 Individual Creative Agents

Individual creative agents form the fundamental building blocks of computational social creativity. While the broader framework emphasizes collective dynamics, understanding how individual agents process and generate creative works remains crucial for implementing effective multi-agent systems. Drawing from Saunders' work [21], individual creative agents must possess three core capabilities: the ability to generate artifacts, evaluate novelty, and maintain a form of creative drive or motivation.

The generative capabilities of creative agents typically involve domain-specific knowledge and production rules. In computational implementations, these often take the form of parameterized generative systems or evolutionary algorithms that can produce artifacts within a defined creative space [22]. The specific choice of generative system, while important for practical implementation, matters less than ensuring the system provides sufficient expressive range for meaningful creative exploration. This aligns with Wiggins' view [10] that creative systems must balance exploration of new possibilities with exploitation of known successful patterns.

Novelty evaluation represents perhaps the most crucial capability of individual creative agents, as it drives both their creative development and social interactions. Computational approaches to novelty detection have evolved from simple difference measures to sophisticated machine learning models. For instance, Pidhorskyi et al. (2018) propose a probabilistic approach using adversarial autoencoders for novelty detection, effectively computing the likelihood that a sample was generated by the inlier distribution [23]. These evaluation mechanisms must be computationally efficient enough for real-time interaction while remaining sensitive enough to detect meaningful creative variations. As Boden [13] argues, the ability to recognize and appreciate novelty forms the cornerstone of creative behavior, whether in human or artificial systems.

The third essential component, creative drive, manifests in computational agents through various mechanisms that maintain engagement in creative activities. This often takes the form of interest functions or curiosity mechanisms that prevent agents from settling into static

behavioral patterns. Saunders and Gero [6] implemented this through a hedonic function based on the Wundt curve, which models how interest varies with novelty. This approach recognizes that both too little and too much novelty can lead to disengagement, mirroring psychological theories of optimal arousal in human creativity [17].

2.2.1 Computational Approaches to Novelty Detection

While various approaches exist for computational novelty detection in creative systems, the choice of mechanism significantly impacts both the quality of creative evaluation and the scalability of the system. This choice introduces a significant trade-off that must be balanced in order to properly scale a system. Early implementations often relied on self-organizing maps (SOMs) for novelty detection [6], which proved effective for small agent populations but might face scalability challenges in larger systems.

The combination of deep feature extraction through ResNet and k-Nearest Neighbors (kNN) classification offers several advantages for large-scale creative systems. Pre-trained convolutional neural networks like ResNet provide robust, hierarchical feature representations that capture both low-level visual patterns and higher-level structural relationships [24]. This ability to extract meaningful features without domain-specific training makes them particularly suitable for evaluating abstract visual artifacts. Recent advancements in deep learning for novelty detection, such as adversarially learned one-class classifiers, also highlight the utility of leveraging deep architectures for identifying novel patterns in complex datasets [25].

The choice of kNN for novelty detection builds on established research into computational novelty detection [26]. Pimentel et al. provide a comprehensive review of distance-based techniques for anomaly detection, which are highly relevant for creative evaluation in large-scale systems. Unlike neural network-based approaches that require extensive training periods, kNN provides immediate novelty assessments while naturally supporting the development of individual creative preferences, giving agents the ability to change their repository over time.

The implementation of normally distributed novelty preferences among agents reflects empirical observations of human creative communities. However, the lack of a detailed quantitative model for this specific behavior highlights a gap in current literature. This investigation seeks to address this gap by exploring the role of normally distributed thresholds in fostering emergent behaviors in creative societies, inspired in part by general principles outlined in [27].

2.3 Multi-Agent Creative Systems

Individual creative agents constitute basic units of computational creativity research, but creativity manifests primarily through interactions within social networks and communities. The transition from single-agent to multi-agent approaches reflects this understanding, addressing both the collaborative and competitive aspects of creativity. Multi-agent systems offer a framework for modeling these intricate social dynamics, allowing for the investigation of phenomena such as stylistic evolution, creative influence, and the formation of artistic movements. These systems extend beyond simple aggregations of individual creative behaviors, incorporating the critical social mechanisms that shape collective creative processes.

2.3.1 Historical Development of MAS in Creativity Research

The application of multi-agent systems to study creativity represents a significant paradigm shift from traditional cognitive models to social and distributed perspectives [11]. Early research in computational creativity primarily focused on individual creative processes [28], but the recognition of creativity as an inherently social phenomenon led to the emergence of multi-agent approaches in the late 1990s.

The field's evolution began with simple cellular automata models exploring emergent behavior [29], before progressing to more sophisticated agent-based frameworks. A pivotal development came with Saunders and Gero's Digital Clockwork Muse [11], which demonstrated how creative behavior could emerge from interactions between artificial agents equipped with novelty-seeking mechanisms.

This transition to social models was further catalyzed by Csikszentmihalyi's systems view of creativity [30], which emphasized the role of social interactions and cultural context in creative processes. The subsequent development of MAS frameworks allowed researchers to computationally investigate these social dynamics, leading to insights about the emergence of creative communities and the diffusion of innovations [31].

Recent advances in computational power and artificial intelligence have enabled the scaling of these models from small groups of agents to large-scale social simulations [32], opening new avenues for understanding collective creativity in real-world contexts.

2.3.2 Artificial Creative Systems

The Digital Clockwork Muse represents a foundational framework for modeling social creativity through multi-agent interactions. At its core, the algorithm implements a cyclic process where agents generate artifacts, evaluate their novelty, and engage in social exchange based on their individual preferences and collective dynamics. Figure 2 provides a high-level architectural view of implementing the DIFI framework computationally, illustrating the essential components and their interconnections.

Early attempts to computationally model Csikszentmihalyi's DIFI framework took different approaches. Liu proposed the Dual Generate-and-Test model which conceptualized creativity as two interconnected generative cycles [19]. The first cycle operates at the individual level, implementing problem finding, artefact generation, and personal creativity testing. The second cycle operates at the societal level, where the individual serves as the generator and the field functions as a monolithic test of creativity. Liu's model suggested that this societal creativity test would likely require human judgment as an oracle to evaluate creative merit. This interpretation maintained a centralized view of creative evaluation, where a singular authority determines what enters the domain. In parallel, Saunders developed a distributed approach where creative evaluation emerged from the interactions of multiple agents [21]. This distributed evaluation better reflects the emergent nature of social creativity, where creative judgments arise from collective interactions rather than centralized authority.

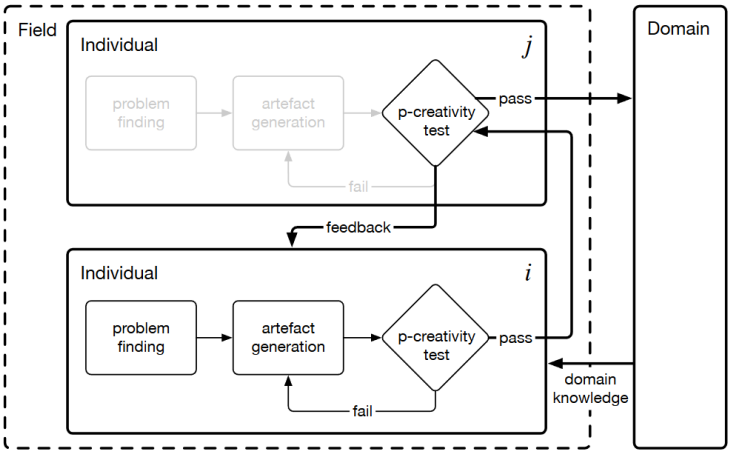


Figure 2: A minimal social creativity test in an artificial creative system, figure by Saunders [21].

Within this architecture, individual agents (denoted as i, j, ..., n) operate as independent creative entities, each capable of generating and evaluating artifacts. Although the framework accommodates a predefined number of agents, its scalability allows for populations ranging from small groups to large creative communities. This flexibility enables the investigation of emergent creative phenomena across different scales of social interaction.

The creative process begins with problem finding: Problem finding represents the agent's intrinsic mechanism for identifying unexplored creative spaces through stochastic generation and novelty-based evaluation. Although initial artefact generation may be random, agents may develop strategic approaches to creativity through their accumulated experiences directly affecting novelty evaluations. This is implemented through the personal creativity (p-creativity) test, which allows agents to assess the potential interest and value of their creative outputs before sharing them with the broader agent community.

The individual agent i would continue to generate artifacts until their individual p-creativity test had passed. If the artefact were not novel enough by the individual, it would be binned, and a new artefact would be generated. However, in the scenario where the agent i accepts their artefact as being novel, it would be broadcast to a different agent j. Agent j then performs their individual p-creativity test to determine if the artefact is novel enough. The p-creativity test performed by j is not equal to the test that i performed, because their perception of the perceived artifacts is different. If agent j accepts the artefact, it is added onto the domain.

2.3.3 Emergence of Creative Communities

Creative communities in artificial systems emerge through complex interactions between agents, much like their human counterparts. As agents interact and share artifacts, they naturally develop preferences and evaluation patterns that can lead to the formation of distinct creative groups. This emergence of community structure differs from explicitly programmed social behaviors, arising instead from the cumulative effects of individual creative decisions and evaluations [7].

In small-scale implementations (fewer than 20 agents), researchers often needed to artificially constrain agent interactions or impose structural limitations to observe phenomena like clique formation. For instance, [9] demonstrated that in populations of under 20 agents, artificially constraining communication pathways was necessary to observe the formation of creative subgroups. Similarly, [6] showed that in small agent populations, explicit limitations on interaction patterns were required to study the emergence of specialized creative roles and community structures.

However, studies in computational social creativity suggest that larger agent populations might naturally develop these social structures through their creative interactions, mirroring the way human creative communities organically form around shared aesthetic preferences and creative approaches [8]. This natural emergence of community structure becomes particularly relevant when scaling creative systems to populations more representative of real-world creative communities [2].

The emergence of creative communities exhibits several characteristic patterns that parallel human creative societies [1]. Agents tend to form loose associations based on compatible novelty preferences and shared creative interests. Over time, these associations can strengthen into more defined groups, each developing distinct creative styles and evaluation criteria. This natural clustering of creative agents provides a foundation for understanding more specific emergent properties in creative social systems and their relationship to human creative communities, as described by [3] and [5].

2.3.4 Modern Approaches to Social Network Analysis

The analysis of large-scale social networks has fundamentally transformed with advances in algorithmic approaches and computational capabilities. Traditional community detection methods were limited to networks of a few thousand nodes [33], but current methodologies can effectively analyze networks with millions of nodes and edges [34].

Community detection methods advanced significantly with the introduction of hierarchical approaches. The Louvain method [34] introduced a computationally efficient multi-level optimization technique based on iterative node aggregation. This reduced the computational complexity from previous methods while retaining detection accuracy. The later development of the Leiden algorithm [35] built upon this foundation by adding formal mathematical guarantees missing in the original method.

Modularity-based methods face an inherent resolution limit [36], making it impossible to detect communities below a certain relative size threshold. Research addressing this constraint led to the development of tunable resolution parameters [37], allowing analysis across multiple scales. However, subsequent analysis of the Louvain method uncovered significant limitations in community connectivity [35]. Quantitative evaluation showed that 25% of detected communities lacked proper connectivity, with a subset being completely disconnected. The Leiden algorithm [35] resolved these structural issues through provable connectivity guarantees and improved convergence properties, while reducing computational overhead. These methodological refinements established a more rigorous foundation for analyzing complex network structures.

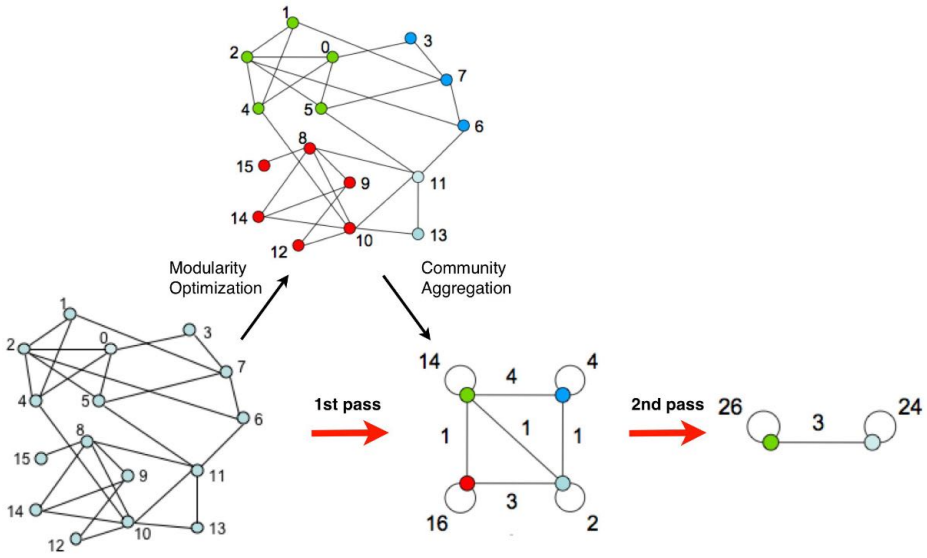


Figure 3: Hierarchical community detection process demonstrated across multiple iterations. The algorithm proceeds in two alternating phases: First, a local optimization phase where nodes are assigned to communities to maximize modularity. Second, a network reconstruction phase that aggregates the discovered communities into a new, compressed network representation. This process continues until modularity can no longer be improved [34].

Network analysis has been substantially enhanced by developments in machine learning and information theory. The Infomap algorithm [38] introduced an information-theoretic approach to community detection, defining communities through the patterns of information flow in networks. More recent developments incorporate deep learning techniques through node embeddings [39] and graph neural networks [40]. These methods map network structures into continuous vector spaces, enabling more sophisticated analysis of network properties and community structure. This mathematical framework has proven particularly effective for capturing temporal dynamics in evolving social networks.

2.4 Emergent Properties

The study of emergent properties in social creativity systems reveals how complex collective behaviors arise from simple individual interactions [8]. While traditional computational creativity research often focused on individual creative processes, scaling these systems to larger populations enables the observation of emergent social phenomena that mirror human creative communities [41]. These properties emerge naturally from the interactions between agents rather than being explicitly programmed, providing insights into how creative communities self-organize [21].

Recent research has demonstrated that emergent behaviors in creative multi-agent systems often parallel those observed in human artistic movements and scientific communities [42]. As these systems scale, they exhibit increasingly complex social structures and creative patterns that cannot be reduced to the properties of individual agents [43]. These emergent phenomena

include the spontaneous formation of aesthetic niches, the evolution of distinct artistic styles, and the development of hierarchical influence networks [44].

Of particular interest is how these systems naturally develop social structures that facilitate both innovation and tradition - a balance crucial for sustained creative development [45]. The emergence of these properties often requires a critical mass of agents, highlighting the importance of scale in studying computational social creativity [7].

2.4.1 Clique Formation and Social Clustering

In creative multi-agent systems, clique formation manifests as emerging tightly interconnected subgroups of agents that share similar aesthetic preferences or creative approaches. These clusters often exhibit higher internal interaction rates compared to their external connections [46]. Research has shown that such clustering can lead to localized creative dialects—distinct stylistic variations that emerge within cliques while maintaining broader community-level coherence [47]. This social structuring influences artifact generation by creating micro-communities that might serve as evolutionary pressure chambers for specific artistic traits.

2.4.2 Stylistic Evolution and Divergence

The emergence of distinct artistic styles in multi-agent systems occurs through a process of cultural drift and selective pressure [48]. As agents interact within their formed communities, they develop shared aesthetic preferences that guide their creative outputs. This leads to stylistic divergence between different agent clusters, while maintaining coherence within clusters [49]. Studies have demonstrated that such divergence increases with system scale, as larger populations enable the stable maintenance of multiple distinct artistic traditions [50].

2.4.3 Knowledge Transfer Patterns

The diffusion of creative innovations across agent populations follows identifiable patterns of knowledge transfer [51]. Dominant artifacts, those that significantly influence subsequent creative outputs, often emerge at the boundaries between different creative communities [52]. These artifacts serve as creative bridges, facilitating the exchange of aesthetic innovations between otherwise distinct groups. The rate and pattern of this knowledge transfer is heavily influenced by the underlying social network structure and the strength of inter-community connections [53].

3 Methodology

As previously described, this research builds upon and extends the Digital Clockwork Muse framework to investigate emergent properties in large-scale creative multi-agent systems [11]. The original framework demonstrated the feasibility of modeling creative social behaviors in small agent populations, typically fewer than twenty agents. These early implementations required artificial constraints to produce phenomena like clique formation and stylistic evolution. Our methodology aims to examine whether such properties emerge naturally when the system is scaled to thousands of agents. To achieve this, the original implementation of the Digital Clockwork Muse had to be carefully reconsidered. Some fundamental changes have been made to that framework and will be described thoroughly in this chapter, particularly focusing on the computational optimizations and architectural modifications necessary to support large-scale agent interactions while maintaining the core principles of social creativity.

3.1 Algorithmic Architecture of the Digital Clockwork Muse

Our implementation preserves the core mechanisms of Saunders' original work while adapting the architecture to support significantly larger agent populations. In this section we will explore the details of the framework.

The Digital Clockwork Muse implements a cyclic process where agents are able to generate, evaluate, and share creative artifacts within a social framework. The core algorithm, illustrated in Algorithm 1, defines both individual agent behavior and social interaction patterns that together constitute the creative system. Each agent operates autonomously while participating in a broader creative community through structured communication and evaluation processes.

Algorithm 1: The Original Digital Clockwork Muse algorithm, illustrating the core mechanisms of individual agent behavior and social interaction in a multi-agent creative system, by Saunders [21].

while $t <$ *total simulation time* **do**
  for *each agent* i *in field* F **do**
    update interest h_i for artefact a_i, $h_i = H_i(N_i(a_i))$
    while *message queue* Q_i *is not empty* **do**
      remove artefact a^n from Q_i, sent by agent n
      calculate the hedonic value $h_i^n = H_i(N_i(a^n))$
      update memory M_i to include a^n
      send feedback including h_i^n to sender agent n
      if $h_i^n >$ *domain submission threshold* (τ_D) **then**
        submit artefact a^n to domain D
      end
      if $h_i^n > h_i$ **then**
        adopt received artefact, $a_i \leftarrow a^n$, $h_i \leftarrow h_i^n$
      end
    end
    generate new artefact a_i' from a_i
    calculate the hedonic value $h_i' = H_i(N_i(a_i'))$
    update memory M_i to include a_i'
    if $h_i' >$ *communication threshold* (τ_C) **then**
      select an agent m from F, where $m \neq i$
      send artefact a_i' to agent m
    end
    if $h_i' > h_i$ **then**
      adopt generated artefact, $a_i \leftarrow a_i'$, $h_i \leftarrow h_i'$
    end
    update interest level $S_i = \alpha S_i + (1 - \alpha)h_i$
    if $S_i <$ *boredom threshold* (τ_B) **then**
      retrieve artefact a^d from domain D
      calculate the hedonic value $h_i^d = H_i(N_i(a^d))$
      update memory M_i to include a^d
      if $h_i^d > h_i$ **then**
        adopt retrieved artefact $a_i \leftarrow a^d$, $h_i \leftarrow h_i^d$
      end
    end
  end
end

The algorithm operates in discrete time steps, during which each agent i in the field F processes three main phases of creative activity. In the first phase, agents handle incoming communications by processing their message queue Q_i. For each artifact a^n received from another agent n, the receiving agent calculates a hedonic value h_i^n using its novelty detection function N_i and hedonic function H_i. This evaluation serves two purposes: it provides feedback

to the sending agent and determines whether the artifact merits inclusion in the domain D based on a domain submission threshold τ_D.

The second phase focuses on artifact generation and evaluation. Each agent generates a new artifact a_i' based on their current artifact a_i. The hedonic value h_i' of this new artifact is calculated using the same novelty detection and hedonic functions. If this value exceeds a communication threshold τ_C, the agent selects another agent m and shares the artifact. This selective communication mechanism models the discriminating nature of creative individuals who only share works they deem sufficiently interesting.

The final phase implements a form of creative drive through an accumulated interest level S_i, updated as a weighted sum of previous interest and current hedonic value:

$$S_i = \alpha S_i + (1 - \alpha) h_i$$

When this interest level falls below a boredom threshold τ_B, the agent retrieves an artifact a^d from the domain D. This mechanism ensures continuous creative exploration even when direct agent interactions fail to maintain sufficient interest.

Crucially, all evaluations of artifacts, whether received from other agents, newly generated, or retrieved from the domain, use the same novelty detection and hedonic functions from that specific agent. This consistency in evaluation creates a coherent creative personality for each agent while allowing for diversity across the population through variations in these functions' parameters (and the agents internal state). The algorithm thus implements both the individual generative aspects of creativity and the social dynamics of creative communities described in Csikszentmihalyi's systems view [17].

This architecture supports scaling to larger agent populations primarily because each agent's decision-making remains local, which is based only on their individual evaluation functions and immediate interactions. However, the collective behavior emerging from these local interactions becomes increasingly complex and potentially more representative of real creative communities as the number of agents grows. The challenge in scaling lies not in the algorithmic structure itself, but in efficiently implementing the novelty detection, artifact generation, and communication mechanisms for large agent populations.

3.2 Artefact Generation

The artifact generation system in this research serves primarily as a vehicle for studying social creativity rather than as a focus of the research itself. We implemented a quaternion-based expression tree system inspired by Saunders' Digital Clockwork Muse, though with a simplified set of operations. This system generates abstract visual artworks through the manipulation of quaternion values across a coordinate space, producing images that exhibit sufficient visual complexity to study novelty detection and creative social behaviors. Our implementation uses expression trees that combine basic quaternion operations to transform input coordinates into color values. Each node in these trees represents either a mathematical operation (such as addition, multiplication, or trigonometric functions), a constant value, or coordinate information. The system employs a subset of the original Digital Clockwork Muse operations, focusing on those that reliably produce visually distinct results while maintaining computational efficiency for large-scale simulations.

For social creativity research, the specific method of image generation is not fundamentally important, as long as the system provides agents with sufficient freedom to generate visually distinguishable artifacts with adequate complexity. While the abstract mathematical patterns produced may seem simplistic compared to contemporary AI art, they serve our research purposes effectively by creating distinct visual outputs that agents can meaningfully evaluate. We explored alternative approaches, including modern text-to-image models like Stable Diffusion, but these presented significant challenges. Such an approach would require agents to evolve text prompts over time, necessitating large language model calls for each agent to maintain coherent and meaningful text evolution. In contrast, mathematical art generation through quaternion operations provides an inherently complex yet computationally efficient solution. While Stable Diffusion and similar models can generate more visually sophisticated and recognizable artwork, the computational overhead would make large-scale social creativity simulations impractical. However, exploring such advanced generative models in social creativity research remains an interesting direction for future work, particularly as computational capabilities continue to advance.

3.2.1 Expression Trees and Quaternion Operations

The expression trees are constructed using three types of nodes: constants, coordinate transformations, and operations. Each node in the tree takes quaternion inputs and produces quaternion outputs, which are ultimately mapped to RGB colors. The system maintains a reduced set of operations compared to the original Digital Clockwork Muse, focusing on operations that provide visual diversity while remaining computationally efficient.

Category	**Operation**
Constants	Q_IDENTITY, Q_I, Q_J, Q_K
Basic Arithmetic	Addition, Subtraction, Multiplication, Division, Power, Cube
Trigonometric and Hyperbolic	sin, cos, tan, sinh, cosh
Special	exp, log, sqrt, abs, mod2, floor
Transformations	normalize, conjugate, rotate45, ripple, swirl, blend
Coordinate and Logical	coord, imin, imax, ilog, isin

Table 1: Quaternion Operations Used in Artifact Generation

Each expression tree is evaluated across a 32x32 coordinate grid, where the coordinates are first converted to quaternion form before being processed through the tree. However, this model is not limited to the 32x32 resolution, as it can scale to any, but for the purpose of our simulation and keeping the computation cost as low as possible, larger images are not a necessity. to The final quaternion output at each coordinate is converted to RGB values through a normalization process that maps the quaternion components to the [0,255] range.

The structure and depth of the expression trees significantly impact both the visual complexity of the generated artifacts and the computational requirements of the system. Through empirical testing, we established a generation depth range of 6 to 10 levels for new expressions, which provides sufficient complexity while maintaining computational efficiency in large-scale simulations. At the lower bound of 6 levels, expressions can still

produce meaningfully distinct patterns through combinations of basic operations, coordinate transformations, and trigonometric functions. The upper bound of 10 levels allows for more intricate compositions while avoiding the diminishing returns in visual complexity and substantial computational overhead observed at greater depths.

3.2.2 Breeding and Evolution

The artifact generation process begins when an agent enters the simulation with no prior experience or stored artifacts. In this initial state, the agent creates a random expression tree using the operations described in Table 1 (Full table 3, with a depth randomly sampled between 6 and 10 levels. This first expression serves as the agent's starting point for creative exploration. As agents interact within the simulation and evaluate artifacts (both their own and those received from others), they maintain a personal repository of expressions that they have seen before (Either their own generated works, or the ones they have approved of from other agents). When generating new artifacts, agents utilize this repository through a breeding mechanism that operates on the mathematical expressions themselves, rather than working with the resulting RGB images. We made this distinction, as breeding at the expression level allows for the preservation and combination of the mathematical structures that produce particular visual effects.

For each new generation after the initial random creation, an agent selects a parent expression from its repository of experienced artifacts and breeds it with its current expression. The breeding process combines these parent expressions by randomly selecting subtrees from each and creating a new expression that inherits properties from both parents. This mechanism allows for the preservation of successful mathematical patterns while introducing variations that might lead to novel visual effects. The resulting child expression maintains a depth within the established bounds (6-10 levels) to ensure computational efficiency.

3.2.3 Expression Complexity and Visual Results

The generated artifacts are rendered as 32×32 pixel images for the simulation, with each pixel's color determined by evaluating the expression tree at that coordinate position. Figure 4 demonstrates how increasing expression tree depth leads to progressively more complex visual patterns. At depth 2, the expressions produce simple, geometric patterns using basic quaternion operations. As the depth increases to 4 and 6, we observe more intricate combinations of operations, resulting in more sophisticated visual effects. At depths 8 through 12, the expressions create complex, layered patterns through the nested combination of multiple operations.

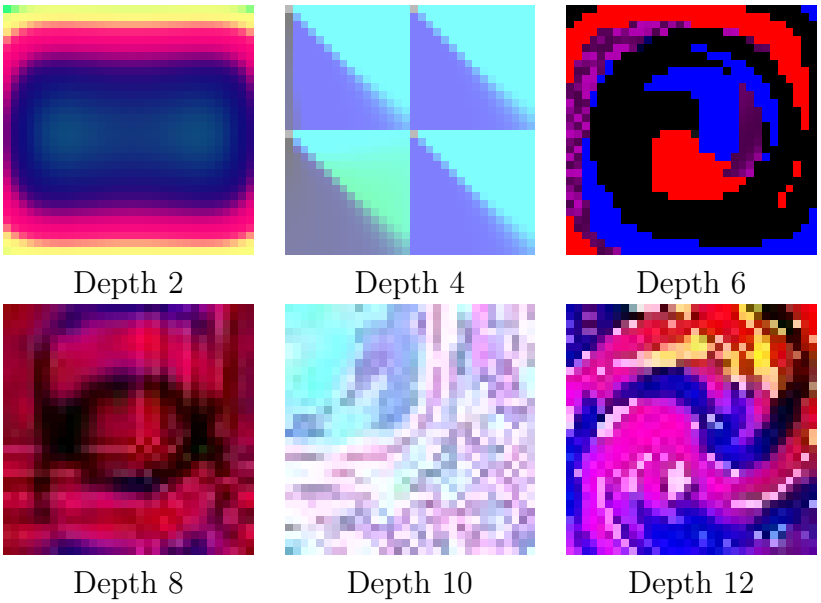


Figure 4: Visualization of 32×32 images generated by the system. The images correspond to expression trees with increasing depths: 2, 4, 6, 8, 10, and 12. As the depth increases, the complexity of the patterns also grows, showcasing the system's ability to generate intricate and visually diverse artifacts.

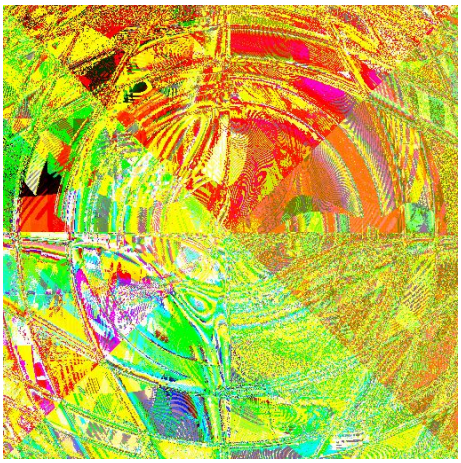
The expressions themselves reveal the increasing complexity of the mathematical operations that produce these visual patterns. For example, at depth 2, we see a simple composition of two operations (swirl and cosh), while at depth 12, the expression combines dozens of operations in intricate ways. This growth in complexity is not merely quantitative; the layering of operations allows for qualitatively different visual effects to emerge.

Image	Depth	Expression Tree
Image 1	2	(swirl (cosh (coord)))
Image 2	4	(iexp (imax (imin (sqrt (coord)) (norm (coord))) (/ (ilog (coord)) (mod2 (coord)))))
Image 3	6	(- (sin (cos (cosh (sinh (conj (coord)))))) (cosh (* (log (swirl (sinh (coord)))) (imin (cos (* (coord) (coord))) (ripple (conj (coord)))))))
Image 4	8	(ilog (blend (sqrt (sinh (cube (rot45 (/ (ripple (coord)) (cos (coord))))))) (exp (conj (iexp (tan (spiral (log (coord))))))))
Image 5	10	(abs (abs (log (swirl (+ (floor (sinh (isin (cube (exp (coord)))))) (mod2 (rolR (cosh (wave (* (coord) (coord))))))))))
Image 6	12	(- (log (- (rot45 (sqrt (sqrt (cosh (inv (rolR (cos (cosh (isin (coord))))))))))) (swirl (rot45 (- (ilog (imax (wave (log (mod2 (swirl (coord))))) (inv (isin (spiral (sqrt (coord))))))) (inv (sinh (norm (norm (sqrt (+ (coord) (coord)))))))))) (spiral (imax (swirl (cos (sinh (sin (ripple (rolR (conj (- (spiral (coord)) (log (coord))))))))) (imax (cos (log (* (tan (floor (sin (mod2 (* (coord) (coord)))))) (/ (ripple (pow (+ (- (coord) (coord)) (sinh (coord))))) (abs (imax (* (pow (coord)) (ripple (coord))) (blend (exp (coord)) (sin (coord))))))))) (coord)))))

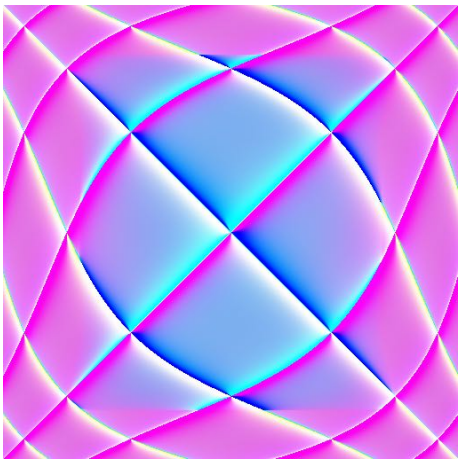
Table 2: Expression Trees for 32×32 Images at Varying Depths, showing the increasing complexity as the depth of expression trees increase.

Table 2 provides the complete expression trees that generated each image, illustrating how the mathematical complexity grows with depth. Even a small increase in depth can substantially expand the range of possible patterns, as each additional level allows for more intricate combinations of quaternion operations.

While our simulation uses 32×32 resolution images to maintain computational efficiency across large agent populations, the expression-based generation system scales effectively to higher resolutions. Figure 5 demonstrates this scalability by rendering two complex expressions at 512×512 resolution with a depth of 16, showcasing the fine detail and intricate patterns possible with our generation system



(a) First high-resolution (512x512) artifact generated with depth 16.



(b) Second high-resolution (512x512) artifact generated with depth 16.

Figure 5: Examples of artifact generation at 512×512 resolution with expression depth 16, demonstrating the system's capability to produce detailed, complex patterns when not constrained by the computational requirements of large-scale social simulation.

These generated images show that this model of evolutionary artifact generation can effectively be used for our social creativity simulation.

3.3 Novelty Detection

One of the main components in modeling social creativity is the ability for agents to detect and evaluate novelty in artifacts. In the original implementation by Saunders, novelty detection was accomplished by using a self-organizing-map (SOM). A SOM can be used to analyze artifacts by mapping high-dimensional image features onto a lower-dimensional grid, preserving the relationships between them. This allows the SOM to act as a novelty detector by identifying images that do not conform to the patterns learned from the input data. Rather than processing raw images directly (Originally 32x32x3), the system first transformed artifacts into binary edge representations using a combination of Laplacian edge detection and fixed intensity thresholding. This preprocessing step reduced the dimensionality of the input while preserving the essential structural features of the artifacts.

While the SOM approach proved effective for small-scale simulations, scaling the system to thousands (or potentially more in the future) of agents presents significant computational challenges. Training individual neural networks for each agent would be prohibitively expensive in terms of computational resources and time, particularly given the continuous nature of the learning process required for novelty detection. Additionally, as agents need to maintain their own unique perspectives and preferences, sharing a single neural network across multiple agents would compromise the system's ability to model diverse creative behaviors, polarizing the research aims.

These limitations motivated the exploration of finding a different and more modern approach to this problem. The requirements for such a system were threefold: it needed to be computationally efficient enough to be deployed across thousands of individual agents, provide interpretable novelty assessments that could inform agent behavior, and avoid the computational overhead of training periods that would make large-scale deployment impractical. Additionally, the chosen approach needed to maintain the ability for each agent to develop unique novelty preferences while supporting rapid evaluation of incoming artifacts.

After evaluating various novelty detection mechanisms, k-Nearest Neighbors (kNN) emerged as a promising solution. The kNN algorithm determines novelty by calculating distances between a new input and its closest neighbors in the agent's accumulated experience space. This approach offers several advantages: it requires no training phase, operates through simple distance calculations that can be efficiently parallelized on modern hardware, and naturally supports individual agent perspectives through their unique collections of reference points.

3.3.1 kNN-based Approach

The kNN-based approach centers on each agent maintaining their own collection of artifacts that serves as a personal reference library for novelty detection. This collection grows dynamically as agents generate new artifacts and receive interesting works from other agents. When evaluating novelty, an agent compares a new artifact against their existing collection using distance calculations to determine how different it is from previously encountered works. This personal collection effectively represents the agent's accumulated creative experience and shapes their evaluation of novelty.

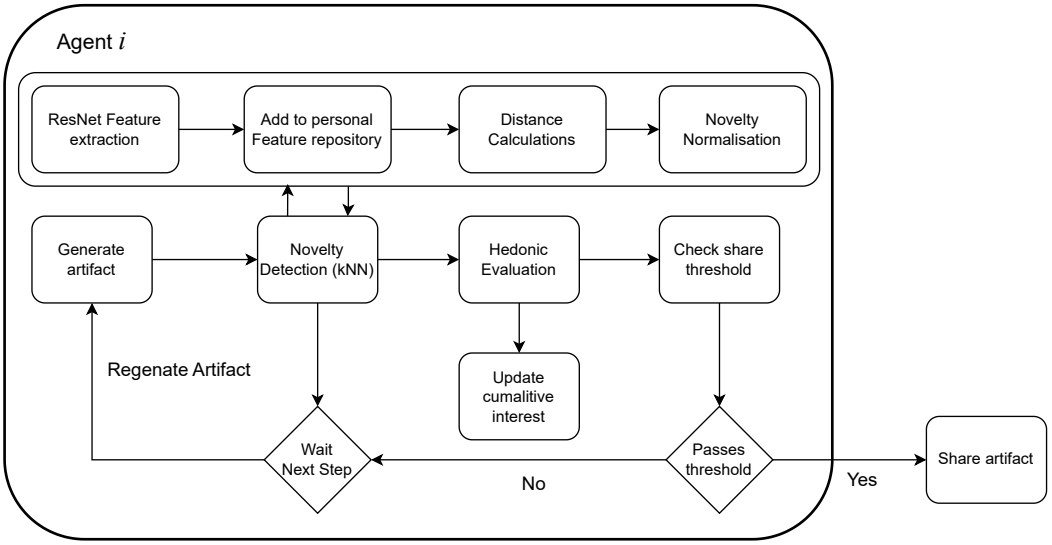


Figure 6: Workflow diagram showing how Agent i processes artifacts through the kNN-based novelty detection system. The process includes feature extraction, adding the features to the personal repository, calculating the novelty value, , performing the novelty normalization, calculating the interest value and updating it and the and decision-making for artifact sharing.

For calculating distances between artifacts, the system uses Euclidean distance in the feature space created by ResNet (detailed explanation in the next subsection). This distance metric was chosen for its computational efficiency and ability to capture meaningful differences between artifacts. When a new artifact is evaluated, the agent calculates its distance to all artifacts in their collection, identifies the k nearest neighbors, and computes the average distance to these neighbors, which is then divided by the standard deviation of those k distances. This normalization helps distinguish between cases where the nearest neighbors are uniformly distant versus cases with high variance in the local neighborhood. The resulting score serves as the base novelty score where a larger normalized distance indicates higher novelty, with the standard deviation helping to account for different density regions in the feature space.

The k-value in our kNN is crucial for our simulation. Therefore we must carefully select the optimal k-value for each agent at the correct time step. The approach to determining the optimal k-value for novelty detection evolved during implementation. While initially considering a dynamic k-value that would scale with each agent's artifact collection size, the final implementation instead employs an automated elbow method. This method systematically identifies the optimal k-value by analyzing the point where increasing k yields diminishing returns in clustering quality. By implementing the elbow method, the system can automatically determine an appropriate k-value for each agent's novelty assessment without requiring manual tuning or potentially unstable dynamic scaling approaches.

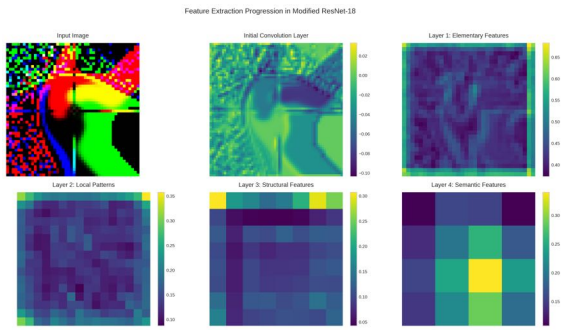
To ensure that these systems can scale, a batching feature has been added to the kNN implementation. When agents need to process multiple artifacts simultaneously from their message inbox, the system employs batch processing through CUDA streams rather than handling each artifact individually. To prevent the linearly increasing amount of operations

required during the inbox phase, calculating distances between feature vectors takes a much larger amount of time because each inbox was previously processed sequentially. While artifact generation remains sequential to preserve the intentional nature of creative exploration, the evaluation of incoming artifacts is parallelized through batching and GPU acceleration. The batching mechanism primarily focuses on optimizing inbox operations, where agents frequently need to evaluate multiple incoming artifacts from other agents in the community. Using CUDA streams, each agent can process their entire inbox of messages in parallel, calculating novelty scores for all received artifacts simultaneously.

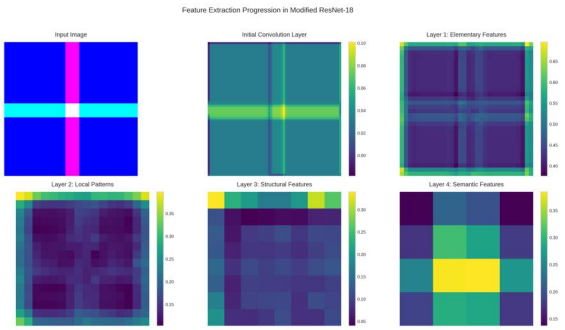
3.3.2 ResNet Feature Extraction

Our feature extraction pipeline makes use of a modified ResNet-18 architecture to extract features from the generated images. Through extensive testing, we found that the choice of which network layer to use for feature extraction significantly impacts the discrimination capabilities between different artistic styles. While it might appear logical to use either the earliest layers for maximum detail preservation or the deepest layers for maximum abstraction, through iterative testing of the ResNet we found several constraints.

The feature extraction process begins with the initial convolutional layer, followed by four ResNet blocks. Figure 3 shows this progression using two fundamentally different input images, revealing how each layer transforms the visual information. Upon examining Layer 3, we observe that despite the input images having substantially different visual characteristics, the feature activations become remarkably homogeneous, with nearly identical activation patterns in several regions. This excessive homogenization makes Layer 3 unsuitable for our purposes, as it would significantly impair the system's ability to distinguish between different artistic styles. Layer 4 demonstrates similar limitations in feature discrimination.



(a) Feature activation maps for input image A, showing progression through network layers (Conv1, Layer1-4). Note the preservation of structural details in earlier layers and increasing abstraction in later layers.



(b) Feature activation maps for input image B, demonstrating consistency in feature extraction patterns across different inputs while highlighting individual variations in structural elements.

Figure 7: Comparative visualization of ResNet-18 feature extraction across network depths, showing the progression from detailed structural features to abstract representations for two distinct input images, noting that for image A and B, layers 3 and 4 are almost indistinguishable from each other.

Although the initial convolutional layer and Layer 1 preserve more structural details, they present significant computational challenges for our multi-agent simulation. With thousands of agents needing to process and compare features simultaneously, utilizing these early layers would be computationally prohibitive due to their high dimensionality and detail preservation. Layer 2 emerged as the optimal extraction point, providing an effective balance between computational efficiency and feature discrimination capabilities.

It is important to note that selecting Layer 2 as our feature extraction point does not circumvent the processing of earlier layers. The ResNet-18 architecture still processes images through the initial convolutional layer and Layer 1 before reaching Layer 2. The key distinction lies in which layer's output we utilize for our feature vectors. Earlier layers, while rich in structural information, would introduce excessive computational overhead in our kNN-based novelty detection system and potentially bias the system toward pixel-level similarities rather than higher-level stylistic features. Indeed, if such low-level feature comparison were desirable, direct comparison of the original images would suffice. Layer 2 provides an appropriate level of abstraction that captures stylistic elements while maintaining computational feasibility for large-scale multi-agent interactions.

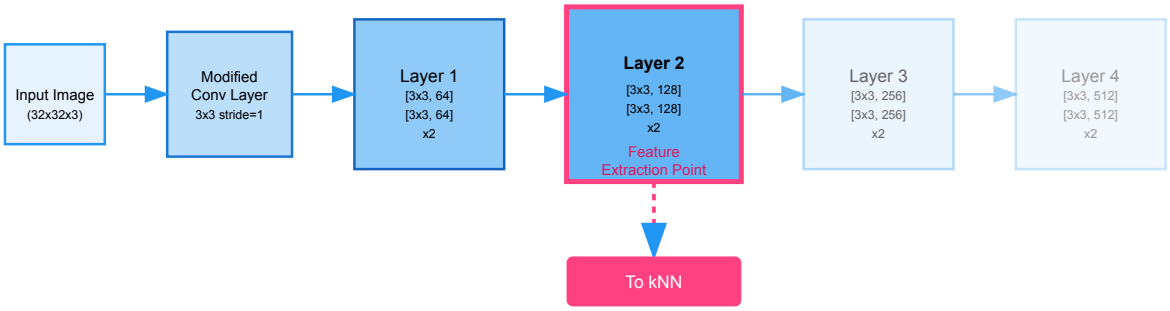


Figure 8: Modified ResNet-18 architecture used for feature extraction. The network processes 32x32x3 input images through successive layers, with Layer 2 serving as the optimal feature extraction point (highlighted). The modified convolutional layer uses a stride of 1 and 3x3 kernel to accommodate the smaller input dimensions. Each ResNet block contains two 3x3 convolutional layers with the specified number of filters.

To ensure consistent feature comparison, we normalize all features extracted from Layer 2 using L2 normalization. This ensures all feature vectors have unit length, making distance calculations in our kNN-based novelty detection more reliable and meaningful. The dimensionality of these feature vectors remains a tunable parameter as we continue to optimize the balance between computational efficiency and discriminative power.

A practical consideration in our implementation involves adapting ResNet-18 for 32x32 input images, which are significantly smaller than the network's original training dimensions. The original training dimensions of Resnet-18 were images sized 224x224, a 7 times increase

compared to our 32x32 images. We addressed this by modifying the initial convolutional layer's stride and kernel size. While this adaptation works effectively with Layer 2 features, it represents another issue against using earlier layers, as they demonstrate increased sensitivity to these architectural modifications.

3.3.3 Novelty Score Normalization

The novelty scores generated through the kNN-based approach operating on ResNet features require normalization to effectively utilize the full range of the Wundt curve. In practice, the raw novelty scores from ResNet feature comparisons tend to cluster within a narrow range, typically between 0.1 and 0.2, with occasional outliers. This compressed distribution would only activate a small portion of the Wundt curve, limiting the system's ability to distinguish between different levels of novelty effectively.

To address this, we implement a dynamic normalization approach using a rolling window of recent novelty scores. The normalization process uses the 1st and 99th percentiles as bounds, rather than absolute minimum and maximum values, to reduce the impact of extreme outliers. For a given raw novelty score x, the normalized score x_{norm} is calculated as:

$$x_{norm} = \frac{x - P_1}{P_{99} - P_1} \tag{1}$$

where:

- P_1 is the 1st percentile of the rolling window of novelty scores
- P_{99} is the 99th percentile of the rolling window of novelty scores

The normalized scores are then clipped to ensure they fall within the [0,1] range:

$$x_{final} = \max(0, \min(1, x_{norm})) \tag{2}$$

This normalization scheme maintains a rolling window of the last 10,000 novelty scores, providing a dynamic adaptation to the evolving distribution of novelty values within the system. By regularly updating the percentile bounds (every 3 steps in our implementation, except for the first 5 steps to ensure proper initialization), the normalization adapts to shifts in the overall novelty distribution that may occur as agents develop their creative preferences and the domain evolves.

The resulting normalized novelty scores span the full [0,1] range, allowing for proper use of the Wundt curve's entire response range and enabling more meaningful differentiation between varying degrees of novelty in the artifacts.

3.3.4 Calculating Interest

The raw novelty scores obtained through the kNN-based approach require transformation into meaningful interest values that can guide agent behavior. Following Saunders' original work, we implement this transformation using a modified Wundt curve, which models the relationship between novelty and hedonic value in a way that captures the principle that both too little and too much novelty can be undesirable. Our implementation extends the original dual-sigmoid approach by incorporating normally distributed novelty preferences across agents.

The interest value for a given novelty score is calculated as the difference between reward and punishment cumulative Gaussian functions:

$$H(x) = R(x) - \alpha P(x)$$

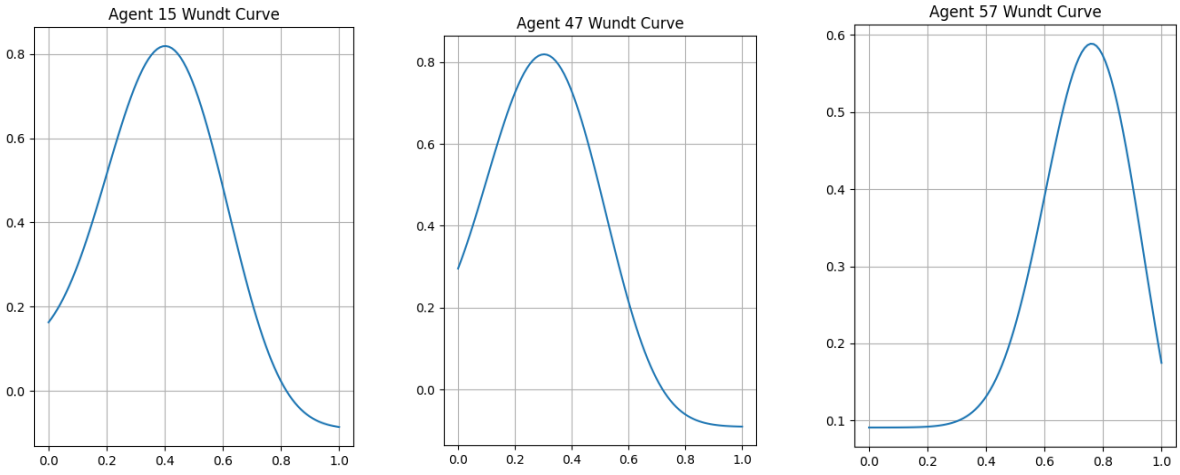
where $R(x)$ and $P(x)$ are cumulative Gaussian functions:

$$R(x) = F(x|\mu_r, \sigma_r)$$
$$P(x) = F(x|\mu_p, \sigma_p)$$

The cumulative Gaussian function $F(x|\mu, \sigma)$ is defined as:

$$F(x|\mu, \sigma) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x - \mu}{\sigma\sqrt{2}}\right)\right]$$

Each agent's reward and punishment thresholds are determined by their individual preferred novelty level, which is sampled from a normal distribution ($\mu = 0.5$, $\sigma = 0.155$). The standard deviation of 0.155 was chosen as it provides adequate coverage of novelty preferences between 0.1 and 0.9 from the mean of 0.5. The reward mean μ_r is set to $\max(0.1, p - 0.2)$ and the punishment mean μ_p to $\min(0.9, p + 0.2)$, where p is the agent's preferred novelty level. Both reward and punishment functions use a standard deviation of 0.15, and the punishment weight α is set to 1.2.



(a) Agent with preferred novelty of 0.4115 (b) Agent with preferred novelty of 0.313 (c) Agent with preferred novelty of 0.8551

Figure 9: Wundt curves for three different agents showing distinct novelty preferences. The curves demonstrate how agents develop individual response patterns to novelty, with peaks aligned to their preferred novelty values. The Y-axis represent the interest value, the X-axis represents the input novelty value.

An agent's accumulated interest is updated using a temporal smoothing function:

$$S_i = \alpha S_i + (1 - \alpha) h_i$$

where $\alpha = 0.35$ controls the decay rate of previous interest values, and h_i is the current hedonic value. The resulting interest values directly influence agent decision-making regarding artifact adoption and sharing behaviors within the creative community, with responses ranging from maximum aversion (-1) through neutrality (0) to maximum interest (1).

The specific parameters chosen for the implementation of the Wundt curve will be systematically evaluated during the experiments to assess their impact on emergent creative behaviors. This implementation creates a natural diversity in creative preferences across the agent population, with each agent's Wundt curve shaped by their individual novelty preference, as demonstrated by the varying peak locations and curve shapes in Figure 9.

3.4 Social Policies

The social policies define the manner of interaction between the agents. Our implementation maintains the core social interaction framework established in the Digital Clockwork Muse while adapting it for large-scale simulations. The social policy system consists of three main components: artifact sharing mechanics, dynamic thresholds, and a boredom mechanism.

The basic artifact sharing flow begins when an agent generates a new artifact. The agent first performs a self-evaluation using their novelty detection mechanism. If the novelty score exceeds the agent's self threshold, the artifact is shared with N randomly selected agents, where N is a configurable parameter. This differs from the original Digital Clockwork Muse implementation, which broadcast to all agents in simulations of fewer than 20 agents. Each receiving agent evaluates the artifact against their domain threshold, and successful evaluations result in the artifact's addition to the domain.

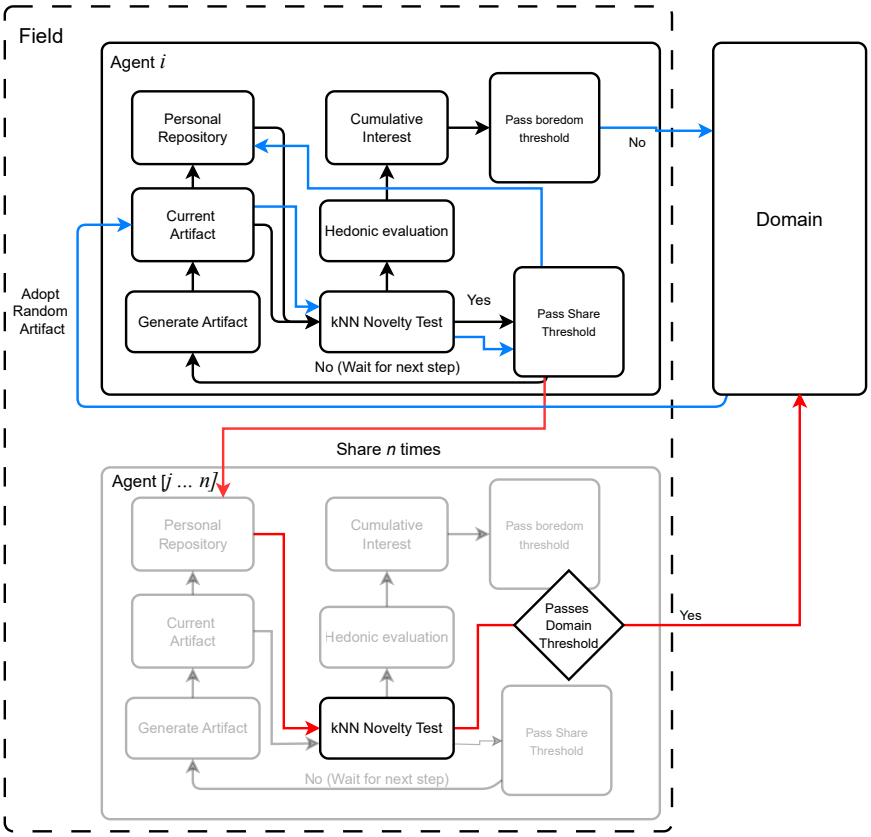


Figure 10: Agent social interaction flow diagram showing how artifacts flow between agents and the domain. Agent i evaluates and generates artifacts, potentially sharing them with other agents ($j...n$) based on novelty thresholds. Successful evaluations by receiving agents can result in artifacts being added to the shared domain. If cumulative interest is below the boredom threshold, a random artifact from the domain is adopted and added to the agent's repository if it passes the self (share) novelty test. The red arrows indicate the artifact sharing path, the blue arrows indicate the adoption path, while the black arrows show the internal processing flow within each agent.

The number of agents N that receive each shared artifact is a static value set at initialization, rather than dynamically adjusted during the simulation. This represents a key parameter for investigating how different sharing patterns affect the emergence of creative behaviors at scale.

The system implements dynamic thresholds using rolling windows of the 100 most recent evaluations. Three distinct thresholds govern agent behavior:

- **Self threshold for sharing artifacts:**
  $\text{self\_threshold} = \text{top\_percentile}(\text{step\_interests\_self}, 80)$
  When an agent evaluates their own generated artifact, this threshold determines whether it is interesting enough to share with other agents. step_interest_self is the variable of the average interest from self evaluation across all agents. The default value is set at 0.1,

allowing all agents to share something in the first step with a relatively low threshold.

- **Domain threshold for artifact acceptance:**
  domain_threshold = top_percentile(step_interests_other, 80)
  When an agent receives an artifact from another agent, this threshold determines whether it is novel enough to be added to the domain. step_interest_other is the variable of the average interest from incoming artifacts evaluation across all agents. The default value is set at 0.1, allowing agents to contribute their artifact to the domain.

- **Boredom threshold for domain access:**
  boredom_threshold = top_percentile(current_interests, 10)
  If an agent's interest level falls below this threshold, they become bored and can retrieve a random artifact from the domain for potential adoption. The boredom threshold is set at 0.2, corresponding to the bottom 10% of agents.

The implementation maintains a rolling window size of 100 for threshold calculations, balancing the responsiveness and stability of the simulation. These rolling windows are updated at the end of each simulation step, after all agents have completed their cycles. After each batch processing cycle, threshold values are synchronized across all agents to maintain system consistency. The window size and percentile values for thresholds can be adjusted to explore different social dynamics in the experimental phase.

The boredom mechanism activates when an agent's interest level falls below the boredom threshold. When triggered, the agent may retrieve and evaluate a random artifact from the domain. If this artifact passes the agent's novelty detection, it becomes their new base for generation, providing a path out of creative stagnation. This mechanism ensures the domain serves as an active component of the creative system rather than merely an archive.

The domain itself maintains a simple structure without regularization or self-checking mechanisms. Once an artifact is added to the domain, it remains there indefinitely. While this implementation serves our current research needs, future work could explore extensions such as temporal dropout or domain curation mechanisms.

3.5 Performance Optimization

Performance optimization might be the most crucial component of this entire project. While we have modernized the basic architecture of the model, we still need to make sure that our model is actually scalable, to the point that a simulation with a few thousand agents is possible in a reasonable time. The optimization processes went through several phases. Initially, the focus was on the basic functionality before addressing any performance bottlenecks, to make sure that the architecture actually worked as intended.

The initial implementation relied entirely on CPU-based processing, using scikit-learn for the kNN computations and Mesa to handle the simulation. One characteristic of Mesa is that it is deterministic and sequential, therefore, one agent must for complete it's cycle before another can start on their cycle for that step. While functional, this approach proved computationally prohibitive when scaling beyond a few dozen agents. Early performance analysis revealed that the primary constraint was not memory utilization but rather computational throughput, as we were fully bound to CPU.

The first phase of optimization focused on transitioning computationally intensive components to GPU-accelerated implementations. The kNN algorithm, being central to novelty detection and heavily used across all agents, was rewritten from its scikit-learn implementation to a custom PyTorch-based solution. Similarly, the artifact generation pipeline and feature extraction components were re-implemented using PyTorch to use GPU acceleration. This initial GPU transition already provided a significant increase in iteration speed, as the limitting factor now became the speed of the GPU, rather than the CPU.

However, while the components now used the full speed of the GPU, they still operated sequentially, processing one agent after another. This sequential processing approach, while functional, failed to leverage one of the GPU's key advantages: batch processing capability. This meant that the simulation could processes each agent in parallel rather than having to wait before the previous agent was finished with their cycle. Performance analysis revealed two primary bottlenecks in the GPU-enabled but sequential implementation. First, the inbox processing phase, where agents evaluate artifacts received from other agents, created a computational bottleneck as each artifact was processed individually through the feature extraction and novelty detection pipeline (While they could have been batched). Second, the artifact generation phase.

To implement the batching required careful reconsideration of Mesa's sequential nature while finding ways to aggregate operations across multiple agents. This meant that the steps were now done on the model level rather than the agent level, and the agents essentially became a data-structure rather than a processing component of the model.

3.5.1 Batch Processing

The inbox processing phase was redesigned to handle message evaluation in batches. Instead of each agent individually processing their received artifacts, the system now collects all pending evaluations across agents and processes them simultaneously. This batch approach uses CUDA streams to parallel process feature extraction and novelty detection for multiple artifacts. The implementation maintains separate streams for feature extraction and novelty calculations, allowing for pipeline parallelism where one batch of artifacts can undergo feature extraction while another completes novelty assessment. Artifact generation similarly benefited from batch processing. The original implementation generated artifacts sequentially, with each agent evaluating their expression tree and producing images one at a time. The optimized system now groups artifact generation into batches, utilizing CUDA streams to parallel process multiple expression trees. This batching mechanism primarily focuses on optimizing the quaternion operations that form the core of artifact generation, allowing multiple agents to simultaneously evaluate their expression trees and generate images.

3.5.2 Algorithmic Optimization

Beyond batch processing, significant performance improvements were achieved through algorithmic optimizations of key computational components. The k-Nearest Neighbors (kNN) implementation, also underwent substantial refinement. Rather than using a fixed k-value or simple scaling rules, the system implements an automated elbow method to dynamically determine optimal k-values for each agent. This approach systematically identifies the point

where increasing k yields diminishing returns in clustering quality, ensuring efficient novelty detection while minimizing unnecessary distance calculations. The thresholding mechanisms for novelty evaluation were optimized through the implementation of efficient percentile calculations. Rather than computing exact percentiles across all historical novelty scores, the system maintains rolling windows of scores and uses approximate quantile calculations. This approach provides comparable thresholding accuracy while significantly reducing computational overhead, particularly important for large-scale simulations where maintaining complete historical data becomes prohibitive. Distance calculations, fundamental to both kNN and novelty assessment, were optimized through several techniques. The implementation uses batched tensor operations for distance computations, making use of GPU acceleration for the matrix operations. Feature normalization, necessary for consistent distance calculations, employs fast approximation techniques that maintain numerical stability while reducing computational cost. These optimizations particularly benefit the simulation at scale, where the number of distance calculations grows quadratically with the agent population.

3.6 Data collecting and analyses

To investigate the results of the forthcoming experiments, we track the following data: network communication, individual agent behaviors, domain evolution, and evaluation patterns. Our analysis framework combines quantitative metrics with network analysis techniques to examine both local agent interactions and global system properties.

Network communication is tracked through directed interaction graphs, recording both successful and failed communication attempts between agents. We analyze these networks using the Louvain community detection method to identify emergent social structures and measure standard network metrics including density, clustering coefficients, and path lengths. To understand the temporal evolution of these networks, we examine community formation and stability over time, supplemented by centrality measures (betweenness and eigenvector) to identify influential agents within the network.

At the individual agent level, we track interest levels, novelty preferences, and contribution patterns. Each agent's evaluation behavior is analyzed through their k-NN parameters and acceptance/rejection patterns. These metrics allow us to investigate how individual preferences and behaviors contribute to collective creative dynamics.

Domain-level analysis focuses on growth patterns and artifact distribution. We measure the rate of domain expansion, acceptance rates of submissions, and the evolution of artifact diversity over time. These measurements help identify potential domain saturation effects, where the rate of novel contributions may decrease as the domain matures.

Data collection is performed through Tensorboard, a platform that easily allows real-time data visualization during the training/simulating process. The data is then extracted and analyzed after using python libraries to create graphs or other meaningful insights.

4 Experiment

This research investigates how creative social behaviors emerge in large-scale multi-agent systems through systematic experimentation with different system parameters. Our experimental design aims to test two central hypotheses: first, that large-scale systems naturally demonstrate emergent creative behaviors without requiring artificial constraints, and second, that these behaviors manifest qualitatively differently across varying scales. Through careful parameter selection and controlled comparisons, we examine how population size, feature complexity, and communication patterns influence the emergence of creative social structures.

4.1 Variables and Parameters

The experimental framework manipulates three primary variables to investigate their effects on emergent creative behaviors:

Population Size We examine creative emergence across three distinct scales:

- Small-scale: 20 agents (baseline, matching Saunders' original work)
- Medium-scale: 50, 100, and 250 agents
- Large-scale: 500 agents

These population levels allow us to observe the transition from constrained small-scale dynamics to emergent large-scale behaviors. They also deem as a baseline from Saunders' work, to see if clique formation or other emergent properties arise without limitations. As stated in the hypothesis, we assume that there will be no clique formation or other emergent properties at this scale due to the inherent complex nature of the artifact generation and the novelty detection. We limit the population size to 500 agents because the computational requirements become too high beyond that point, to efficiently run the experiments for over 5000 steps.

Communication Range We implement fixed communication ranges to examine how different sharing patterns affect creative behavior across population sizes: 1 agent (minimal sharing), 2 agents, 4 agents, 8 agents, 16 agents, 32 agents, and broadcast (sharing with all agents). This fixed-value approach allows us to study how specific communication ranges impact the emergence of creative social structures, regardless of population size. For example, when an agent shares with just 2 others in a large population, we can observe whether localized creative communities naturally emerge. Conversely, broader sharing ranges like 16 or 32 let us examine how increased connectivity affects pattern formation and creative development. Fixed sharing numbers enable us to investigate whether cliques form based on intrinsic social dynamics rather than as artifacts of relative population percentages. Additionally, comparing behaviors across different fixed ranges helps reveal whether creative communities exhibit distinct characteristics when agents have more limited versus broader communication capabilities. The broadest fixed sharing value (32) provides an upper bound for examining localized communication effects, while the broadcast option serves as a control

case representing unconstrained communication similar to Saunders' original work. Note that all communication selection is done randomly, rather than selectively. Over 5000 steps, this random selection helps prevent any artificial patterns from forming in communication networks and ensures that agents have opportunities to interact with the entire population rather than becoming trapped in local networks. While random selection might seem simplistic, it provides a neutral baseline for studying emergent behaviors without the influence of selective communication. Future work could explore more sophisticated approaches where agents learn to selectively choose their communication partners based on past interactions or shared creative preferences.

Feature Dimensionality To investigate how the complexity of artifact representation affects creative evaluation, we test three ResNet feature dimension settings:

- 32 dimensions: Minimal feature representation
- 64 dimensions: Standard feature set
- 128 dimensions: Enhanced feature detection

All of these will be tested for n=20, for every sharing parameter. We will then determine what feature dimensions will be utilized for the future experiments where we test for emergent properties.

4.2 Control Conditions

To validate that observed clique formations and creative behaviors are genuine emergent phenomena rather than artifacts of the system design, we implement two control conditions. These controls will be tested with a population of 20 agents using broadcast communication, matching Saunders' original experimental setup. The first control, Random Communication Control, modifies the communication mechanism by having agents share artifacts with randomly selected peers rather than using the share threshold. While maintaining identical communication frequency as the experimental condition, this control tests whether observed social patterns emerge from genuine creative evaluation or merely from the communication structure itself. All other system parameters remain constant, with random selection simply replacing the interest-based peer choice mechanism. The second control, Uniform Novelty Preference Control, eliminates the diversity in evaluation criteria by initializing all agents with identical novelty preferences. This control specifically tests whether clique formation depends on the heterogeneity of creative preferences within the population. While maintaining all other aspects of the system, including interest calculation and communication mechanisms, this condition allows us to isolate the role of diverse evaluation criteria in the emergence of social structures. These control conditions help isolate the specific mechanisms responsible for clique formation and creative development within the system. By comparing results between experimental and control conditions, we can determine whether observed patterns arise from meaningful creative interactions or are artifacts of the system's basic structure.

4.3 Success Criteria

To evaluate our hypotheses regarding emergent behaviors in large-scale multi-agent systems, we establish quantifiable success criteria focused on network formation and communication patterns. The primary indicator of emergent social structure is the modularity score of agent interaction networks, calculated using the Louvain algorithm. We consider a modularity score exceeding 0.3 as evidence of significant community structure, with the requirement that these structures must persist for at least 1000 simulation steps to be considered stable.

Communication patterns serve as our second metric. We examine the evolution of key system thresholds (boredom, communication, domain) across different population sizes and sharing parameters. Distinct separation and stabilization patterns between configurations would indicate scale-dependent behaviors.

We also track the temporal evolution of network structures, assessing the formation and stability of communities over time. The emergence of persistent, well-defined clusters in larger populations (n>100) without requiring artificial constraints would support our hypotheses.

Control validation forms our final criterion. Using both random communication and uniform novelty preference controls, we expect significantly different network structures in experimental conditions. Success requires demonstrably higher clustering coefficients and modularity scores in experimental conditions compared to controls, with reproducibility verified across five independent simulation runs.

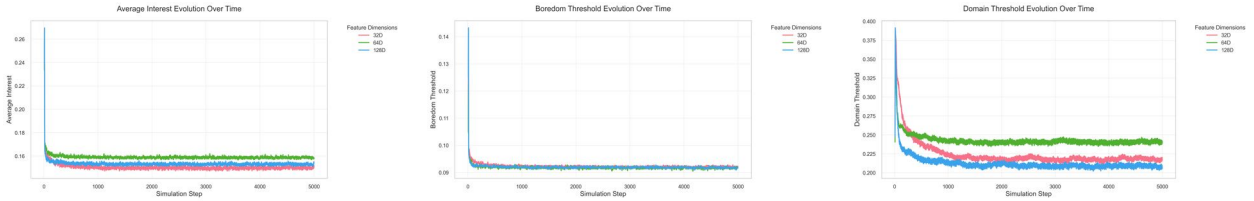
4.4 Hardware used

The hardware used to run these experiments was diverse and catered to different computational needs. The simulations were performed on a combination of machines: a personal computer, Google Colab, and Kaggle. The personal computer utilized an Nvidia RTX 3090 GPU with 24GB of VRAM, offering substantial computational power for local experiments. On Google Colab, both the Nvidia A100 GPU and the T4 GPU were employed, providing significant acceleration for large-scale simulations and data processing tasks. Kaggle's environment leveraged the T4x2 GPU setup; however, the slower CPU in this platform introduced a notable overhead, impacting the overall performance and runtime efficiency of the experiments.

5 Results

5.1 ResNet-18 Feature Dimensions

Prior to conducting the full-scale experiments comparing different population sizes and sharing parameters, we first determined the optimal feature dimensionality for our ResNet implementation. We tested three configurations - 32, 64, and 128 dimensions - using a population of 20 agents across all sharing parameters. The 64-dimensional feature representation



(a) Average interest evolution over simulation steps

(b) Boredom threshold evolution over simulation steps

(c) Domain threshold evolution over simulation steps

Figure 11: Comparison of system behavior across different feature dimensions (32D, 64D, and 128D) over 5000 simulation steps with 20 agents. The 64D configuration (green) demonstrates optimal performance with higher sustained interest levels and domain thresholds while maintaining stable boredom thresholds.

(shown in green in Figure 11) demonstrated superior performance across multiple metrics. As shown in Figure 11a, it maintained the highest average interest levels throughout the simulation, stabilizing at approximately 0.16 compared to 0.15 for 128D and 0.145 for 32D configurations. This higher sustained interest suggests that the 64D features provided optimal discriminative power for novelty detection, enabling agents to maintain engagement with the creative process more effectively. The boredom threshold evolution (Figure 11b) remained remarkably consistent across all feature dimensions, stabilizing around 0.09 after an initial settling period. This consistency indicates that the boredom mechanism functions robustly regardless of feature dimensionality, suggesting it effectively maintains creative drive independent of the chosen feature space. Most notably, the domain threshold (Figure 11c) showed significant variation between configurations. The 64D implementation maintained a consistently higher domain threshold (approximately 0.24) compared to both 32D (0.22) and 128D (0.21) configurations. This higher threshold indicates that the 64D feature space enables more selective artifact acceptance into the domain while still maintaining higher average interest levels, suggesting a better balance between novelty discrimination and creative engagement. Based on these results, we selected the 64-dimensional feature configuration for all subsequent experiments. This choice represents an optimal balance between computational efficiency and feature expressiveness, providing sufficient discriminative power without the additional computational overhead of the 128D configuration or the reduced performance of the 32D configuration.

5.2 Computational performance issues

The implementation and scaling of our multi-agent creative system revealed significant computational challenges inherent in large-scale social creativity simulations. While smaller populations (n<100) performed efficiently, scaling to larger agent populations imposed substantial computational overhead. At 500 agents, even with GPU acceleration on an NVIDIA A100, each simulation step required 15-30 seconds of processing time, primarily due to the quadratic growth in agent interactions and the cumulative computational cost of feature extraction and novelty detection across the population. This computational intensity made simulations beyond 1000 iterations impractical for larger populations, as CPU overhead from agent management and message passing became a significant bottleneck despite GPU acceleration of core computations. These constraints particularly impacted our ability to test broader sharing parameters in larger populations, while we could fully evaluate all sharing parameters (1,2,4,8,16,32,broadcast) for populations up to 100 agents, we were limited to share parameters of 1,2,4,8,16 for 250 agents, and only parameters 1 and 2 for 500 agents. This experience highlights the significant computational challenges in scaling social creativity simulations and suggests that future research in this domain will require both algorithmic optimizations and more sophisticated computational resources.

5.3 System Results

Our experiments across different population sizes and sharing parameters yielded several key metrics tracked over 5000 simulation steps. The results show both network formation characteristics and system threshold behaviors that governed the agent interactions.

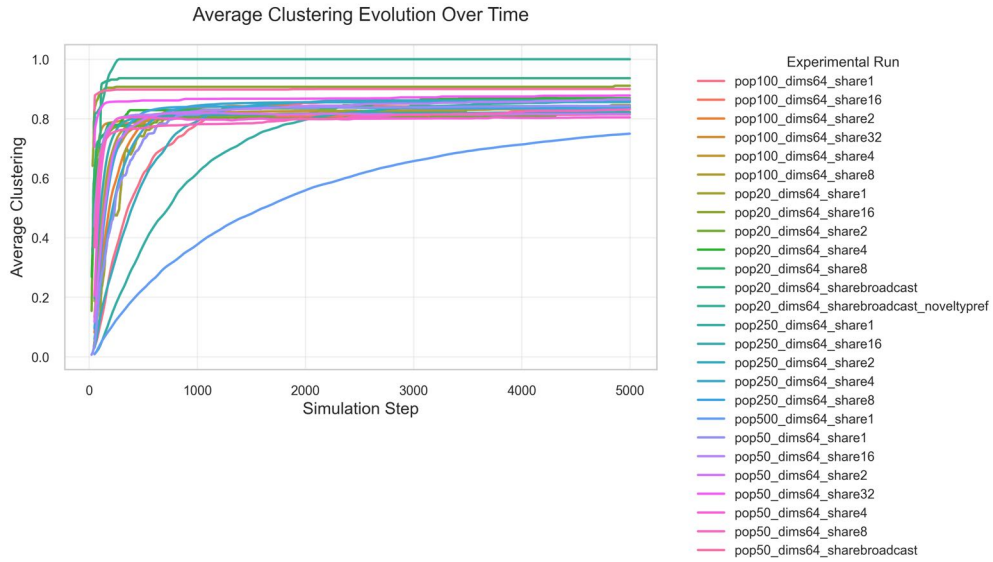


Figure 12: Average clustering coefficient evolution over 5000 simulation steps across different population sizes (20-500 agents) and sharing parameters (1-32 agents and broadcast). Higher values indicate stronger local connectivity between agents.

Figure 12 shows the evolution of average clustering coefficients across all experimental

configurations. The clustering coefficients show rapid initial growth in the first 1000 steps, followed by stabilization at different levels depending on population size and sharing parameters. Population size of 500 with share parameter 1 shows the slowest convergence, while populations of 20 agents reach stable clustering values more quickly. Noting that the control condition with the equal novelty preference has a clustering value of 1, meaning that it is a fully connected graph. It takes a larger amount of steps for a network to get more connected over time, which decreases with the amount of

Figure 13: Average interest evolution over 5000 simulation steps across different experimental configurations. Interest levels indicate agents' responses to received artifacts based on their novelty preferences.

Figure 13 presents the evolution of average interest levels throughout the simulation. Interest levels show initial volatility in the first 500 steps before converging to stable values between 0.12 and 0.17, with different configurations maintaining distinct steady-state levels. The uniform novelty preference control condition (pop20_dims64_sharebroadcast_noveltypref) maintains the lowest stable interest level at approximately 0.12.

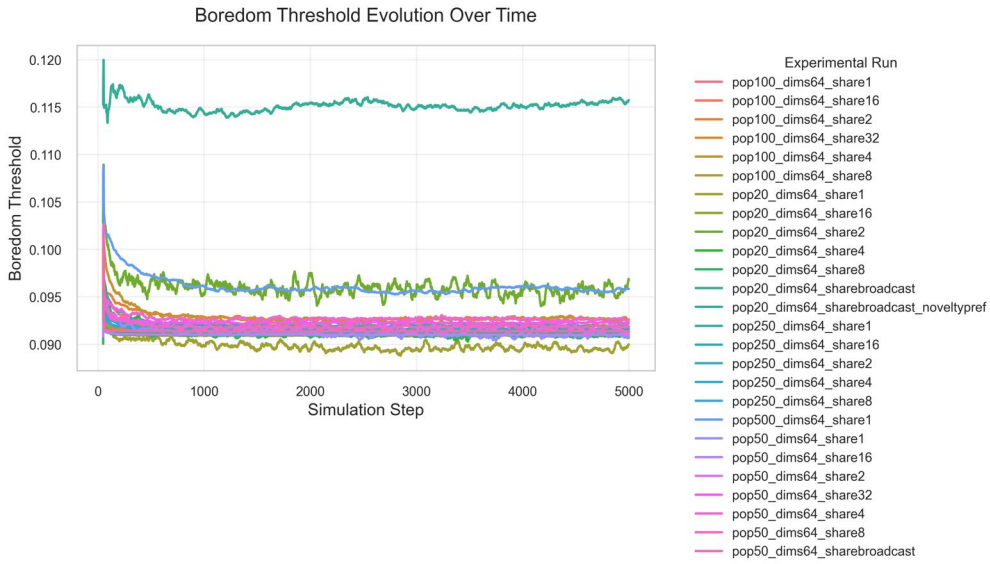


Figure 14: Boredom threshold evolution over simulation time for all experimental configurations. The threshold determines when agents seek new artifacts from the domain due to insufficient interest in current interactions.

Figure 14 illustrates the boredom threshold evolution across all experimental configurations. Most configurations stabilize between 0.09 and 0.095 after initial fluctuations, with population 250 share parameter 1 maintaining a higher threshold around 0.115. The control condition shows distinct behavior with consistently lower threshold values around 0.09.

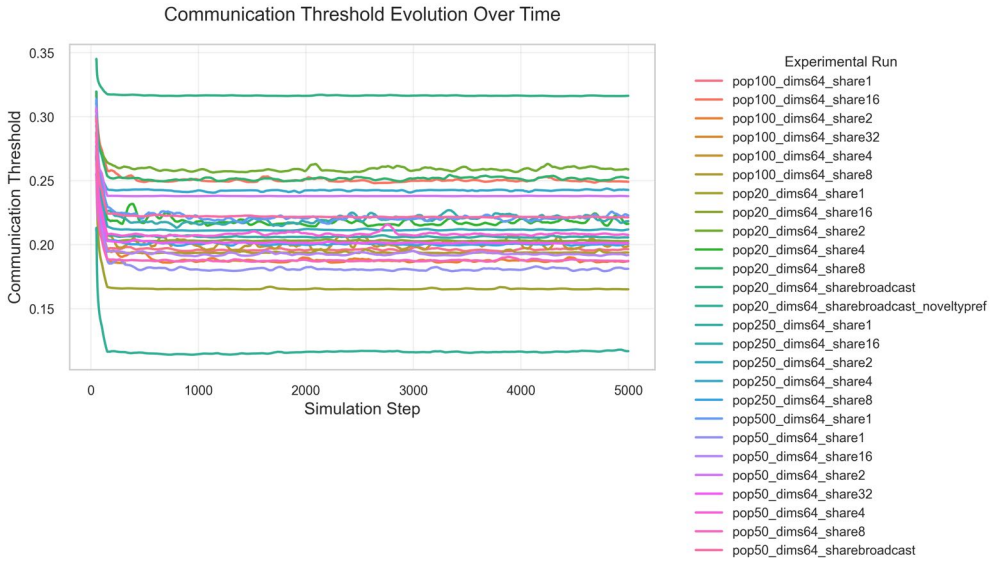


Figure 15: Communication threshold evolution over simulation time. This threshold determines when agents share their generated artifacts with other agents in the population.

Figure 15 shows the evolution of communication thresholds. After initial volatility in the first 500 steps, configurations establish distinct stable levels between 0.12 and 0.32.

The control condition maintains the lowest stable threshold at approximately 0.12, while population 20 with broadcast sharing maintains the highest threshold at 0.32.

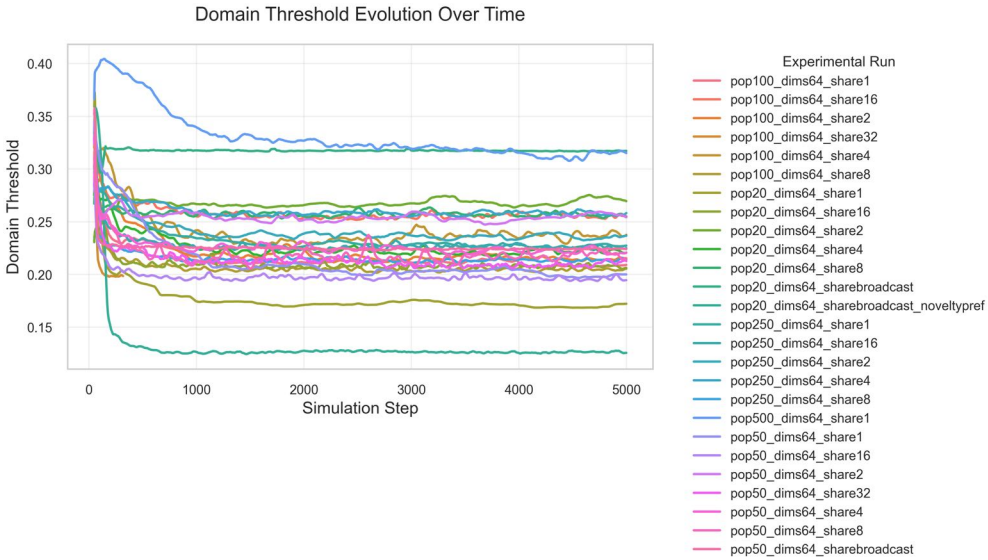
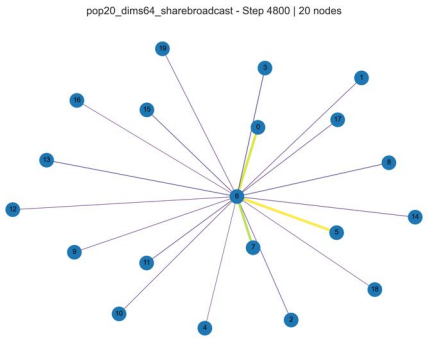


Figure 16: Domain threshold evolution across different experimental configurations. This threshold governs the acceptance of artifacts into the shared domain.

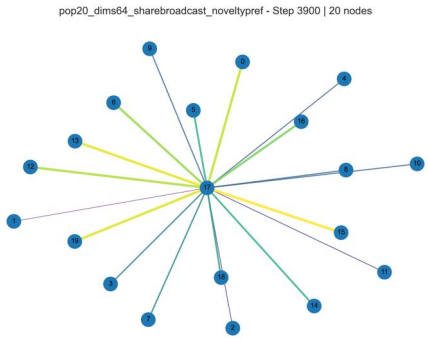
Figure 16 presents the domain threshold evolution throughout the simulation period. Population 500 with share parameter 1 shows the most distinctive behavior, starting at 0.4 before gradually converging to 0.32 after 4000 steps. Other configurations stabilize between 0.17 and 0.27, with the control condition maintaining the lowest domain threshold at approximately 0.13. All threshold metrics demonstrate rapid initial adjustment periods followed by long-term stability, with distinct separation between different population sizes and sharing parameters. The control condition consistently shows divergent behavior from experimental configurations across all metrics, maintaining lower threshold values throughout the simulation. These results show clear differences between control conditions and experimental configurations, particularly in clustering coefficients and interest levels.

5.4 Network structures

After examining system-level metrics, we analyzed the emergent network structures across different experimental configurations. We begin by examining the baseline broadcast communication cases.



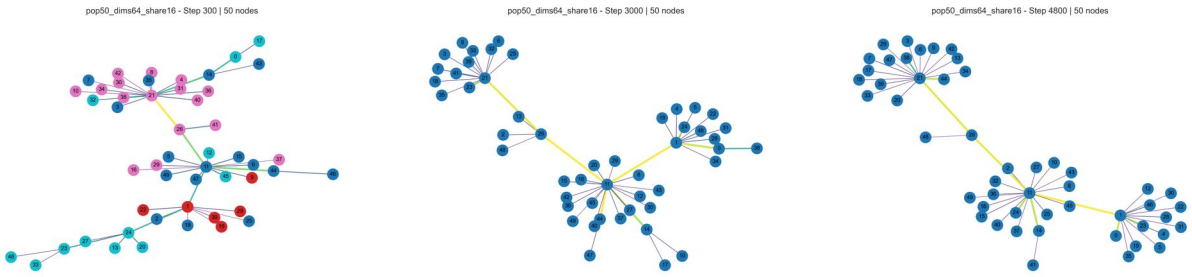
(a) Network structure for population 20 with broadcast sharing



(b) Network structure for control condition (uniform novelty preferences)

Figure 17: Comparison of network structures between experimental broadcast and control conditions with 20 agents. Node positions are determined by a force-directed layout algorithm, where relative distances between nodes indicate connection strength. Edge colors represent different types of creative interactions, with yellower edges indicating stronger connections.

Figure 17 presents network structures from two key configurations: the standard broadcast sharing condition and the control condition with uniform novelty preferences. The node positions in these visualizations are determined by a force-directed layout algorithm (spring layout). Nodes that share more, or stronger, connections tend to appear closer together, forming visually distinct clusters. The absolute positions (the x- and y-coordinates) do not represent any real-world measurement; they serve only as a spatial arrangement to improve interpretability. Both networks show high connectivity, with each node connected to most other nodes in the network, reflecting the broadcast communication parameter. However, subtle differences emerge in the connection patterns. In the standard broadcast condition (Figure 17a), node 6 emerges as a central hub with varying connection strengths to other nodes, indicated by the different edge colors. In contrast, the control condition (Figure 17b) shows more uniform connection patterns radiating from node 17, with a more even distribution of edge colors suggesting more homogeneous interaction strengths.



(a) Early phase (Step 300) (b) Middle phase (Step 3000) (c) Late phase (Step 4800)

Figure 18: Temporal evolution of network structure for population 50 with share parameter 16. Node positions are determined by a force-directed layout algorithm, where distances represent connection strengths. Edge colors indicate interaction strength, with yellow representing stronger connections. The network evolves from initial scattered connections to a stable three-hub structure.

The evolution of network structure in larger populations reveals distinct phases of organization. Figure 18 shows three time points in the development of a 50-agent population with share parameter 16. In the early phase (Step 300), the network shows initial organization around multiple nodes, with agents 21, 11, and 1 emerging as potential hubs. The connections are relatively scattered, with varying edge strengths indicated by the diverse color patterns. By the middle phase (Step 3000), the network has self-organized into a clear three way hub structure, with agents 21, 11, and 1 establishing themselves as primary connection points. These hubs are connected by strong bridges (indicated by yellow edges), while maintaining their own clusters of connected agents. The force-directed layout algorithm reveals the natural separation of these clusters through spatial positioning. The late phase (Step 4800) demonstrates the stability of this three-hub structure, with the same major hubs maintaining their positions and connection patterns. The network maintains its fundamental organization while showing minor adjustments in the peripheral connections. This stability suggests that the three-hub structure represents a natural equilibrium state for this population size and sharing parameter.

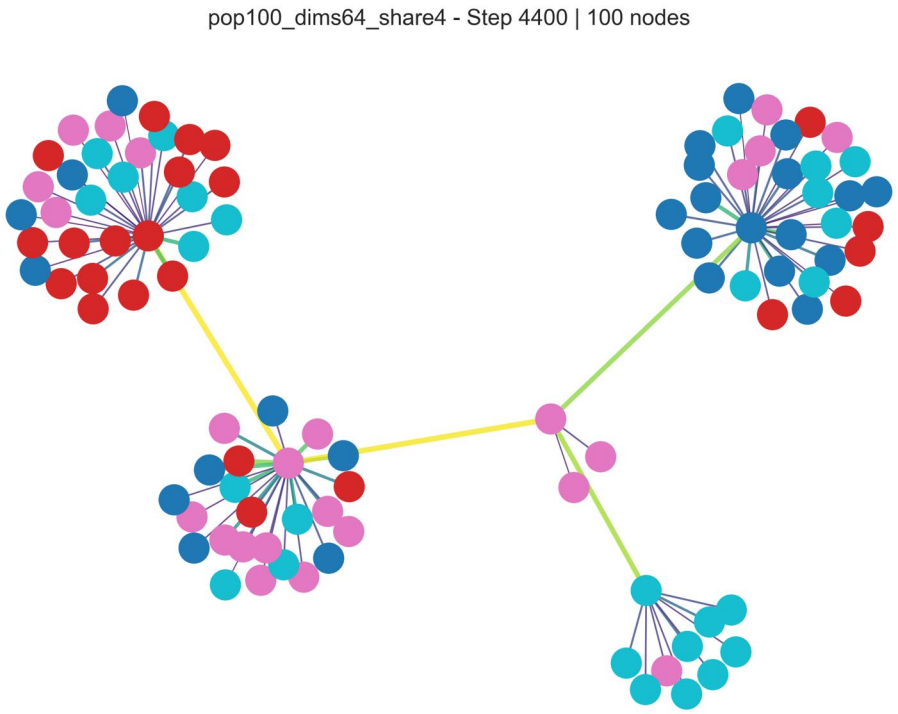
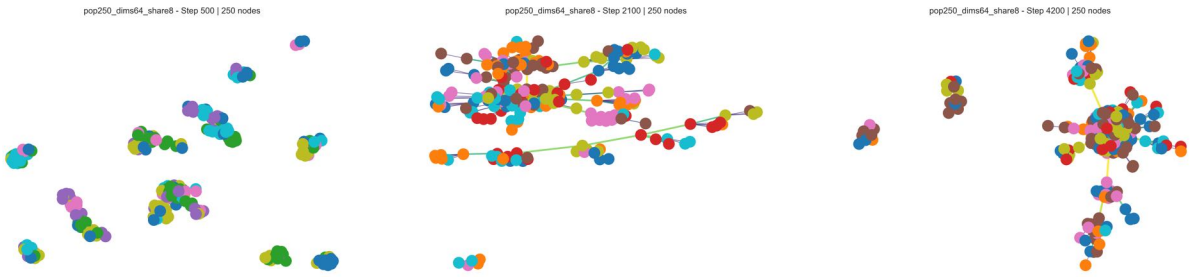


Figure 19: Network structure for population 100 with share parameter 4 at step 4400, demonstrating distinct cluster formation. The network exhibits four clearly defined communities connected by strong bridging links (yellow/green edges). Node colors indicate community membership as detected by the Louvain algorithm.

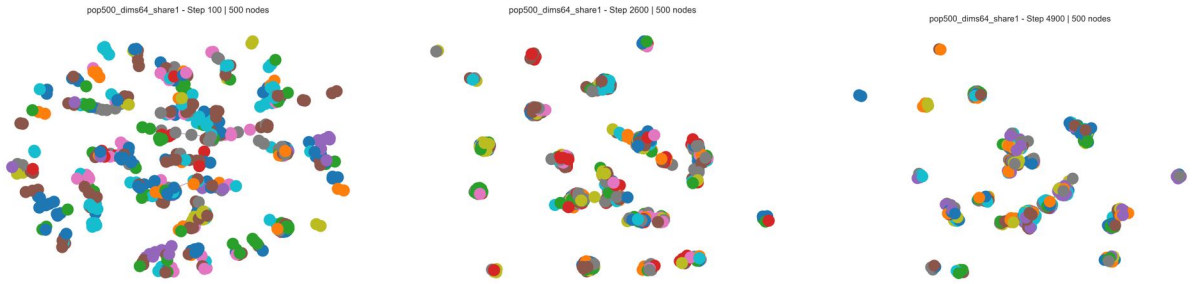
The most visually(!) striking example of emergent clustering behavior appears in the 100-agent population with share parameter 4 (Figure 19). This configuration produces four distinct clusters, each organized around a central hub node. The clusters vary in size but maintain similar structural characteristics, with peripheral nodes connected primarily to their local hub. The force-directed layout algorithm clearly separates these communities, reflecting their relative independence in the interaction space. The clusters are connected by strong bridging links, shown in yellow and green edges, forming a chain-like meta-structure. These bridges represent robust communication channels between communities while maintaining their distinct identities. The limited sharing parameter (4) appears crucial in allowing these distinct communities to emerge naturally, as it prevents the oversaturation of connections that occurs with higher sharing values. Each cluster shows a similar internal organization: a central hub node with multiple direct connections radiating outward in a star-like pattern. The hub nodes serve as local centers of influence, mediating most of the cluster's external communications through the bridge connections. This hierarchical organization emerged naturally from the system's dynamics, without any explicit programming of hub roles or cluster structures.



(a) Initial phase (Step 500) (b) Transition phase (Step 2100) (c) Stable phase (Step 4200)

Figure 20: Evolution of network structure in a 250-agent population with share parameter 8. The system progresses from initially scattered small clusters through a transitional chain-like structure to a final state with distinct, interconnected communities.

The 250-agent population demonstrates a more complex clustering behavior evolving through three distinct phases. Initially (Step 500), agents form numerous small, isolated clusters distributed across the interaction space. During the transition phase (Step 2100), these clusters begin connecting through bridge nodes, creating an extended chain-like structure with multiple branch points. By the stable phase (Step 4200), the network consolidates into several major clusters connected by strong bridging links, with smaller satellite communities maintained at the periphery. This progression reveals how larger populations naturally organize into hierarchical community structures, with both major and minor creative clusters coexisting in a stable configuration.



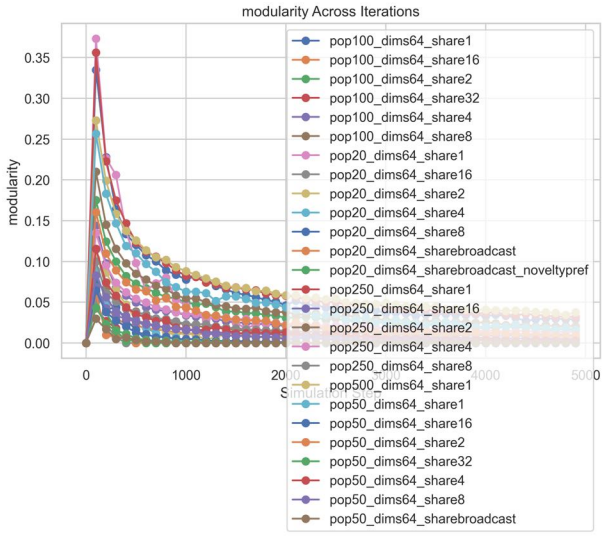
(a) Initial distribution (Step 100) (b) Mid-simulation (Step 2600) (c) Final state (Step 4900)

Figure 21: Network evolution in a 500-agent population with minimal sharing (share=1). The system fails to develop large-scale community structures, instead forming numerous small, isolated clusters.

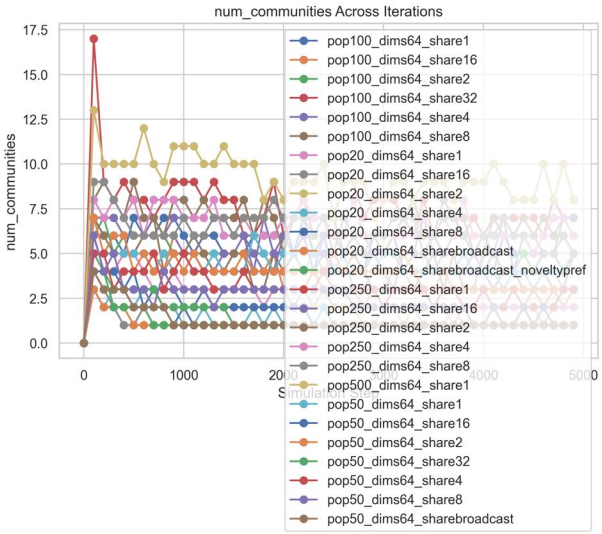
The 500-agent population with minimal sharing parameter (share=1) demonstrates the limits of cluster formation in large-scale creative networks. Initially, agents form a scattered distribution with minimal structure. As the simulation progresses, instead of developing into larger communities as seen in smaller populations, the network fragments into numerous small clusters of 2-5 agents. These micro-clusters remain largely isolated, with minimal bridge connections between groups. By the final state, the network maintains this fragmented structure, suggesting that the minimal sharing parameter prevents the formation of larger coherent

communities in populations of this scale. This fragmentation is notably different from the organized clustering seen in smaller populations or with higher sharing parameters, indicating a threshold effect in the relationship between population size and sharing requirements for coherent community formation.

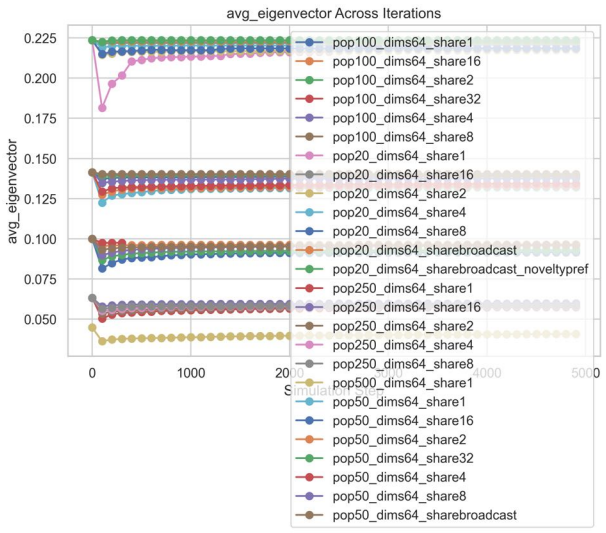
5.5 Network Statistics



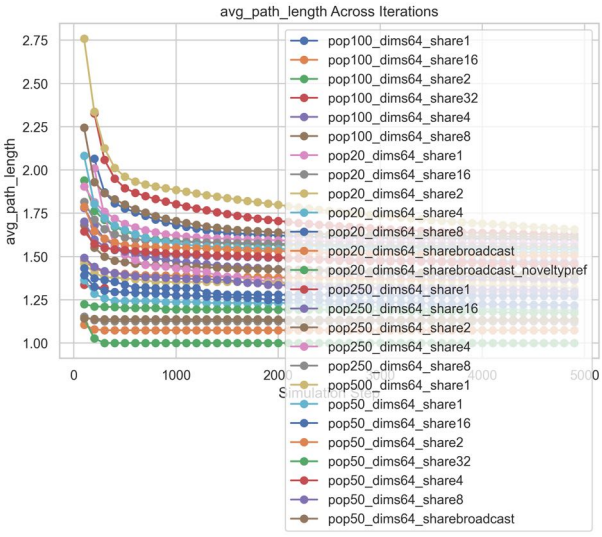
(a) Evolution of network modularity across different experimental configurations, showing initial community formation followed by increasing interconnection.



(b) Number of detected communities over time, demonstrating population size effects on community formation.



(c) Average eigenvector centrality evolution, indicating influence distribution patterns across network configurations.



(d) Average path length development, showing network connectivity patterns across different scales.

Figure 22: Evolution of network metrics across different population sizes and sharing parameters over 5000 simulation steps.

Modularity measurements (Figure 22a) show an initial spike within the first 100 steps across all configurations, reaching maximum values between 0.35-0.37. This is followed by a rapid decline and stabilization below 0.1 after approximately 2000 steps. The uniform novelty

preference control condition (pop20_dims64_sharebroadcast_noveltypref) consistently shows the lowest modularity values, indicating minimal community structure. Community detection analysis (Figure 22b) reveals distinct patterns based on population size. Population 500 with share parameter 1 maintains the highest number of communities, consistently showing 8-10 distinct groups. Medium-sized populations (n=100, 250) stabilize between 3-7 communities depending on sharing parameters, while smaller populations (n=20) typically maintain 1-3 communities. The broadcast sharing conditions consistently show the lowest number of communities across all population sizes. Eigenvector centrality measurements (Figure 22c) demonstrate clear stratification based on population size. Larger populations (n=500) exhibit lower average centrality values (0.04-0.05), while smaller populations maintain higher values (0.20-0.22 for n=20). This indicates more evenly distributed influence in larger networks compared to concentrated influence in smaller ones. Average path lengths (Figure 22d) show initial volatility before stabilizing between 1.0 and 2.75. Population 500 with share parameter 1 exhibits the longest average paths, starting at 2.75 and gradually decreasing to 1.8. Most other configurations stabilize between 1.2 and 1.6, with the control condition maintaining the shortest path length at 1.0. Higher sharing parameters consistently result in shorter average path lengths across all population sizes.

6 Discussion

The computational requirements of scaling multi-agent systems proved to be a fundamental limiting factor in this research. Our experimental results demonstrate that the relationship between system scale and computational cost follows a steeper curve than initially projected, directly impacting the scope and depth of our investigation. Architectural decisions that allowed more sophisticated agent interactions in small populations became significant bottlenecks when scaled to larger groups.

The primary bottleneck came from the cumulative cost of data tracking, novelty detection and the image generation. Despite implementing GPU acceleration and batch processing optimizations (Section 3.5), simulations with 500 agents required 15-30 seconds per step on an NVIDIA A100 GPU. This computational intensity made extended simulations impractical for larger populations, limiting our ability to fully explore emergent behaviors over longer time periods. The constraint became particularly evident in our communication experiments, where we could only test limited sharing parameters (1 and 2) for the 500-agent population, compared to the full range of parameters (1-32 and broadcast) available for populations under 100 agents. As seen from the results, the 500 agents simulations require a lot more time to converge, compared to the smaller simulations. However, when viewed from a practical stand point, it would make sense that the amount of shares would have to scale with the population, in order to achieve similar convergence time.

Our batch processing implementation, while effective for optimizing GPU utilization, could not fully mitigate these scaling issues, as GPU overhead remained an issue. The fundamental architecture of the system, particularly the need for individual agents to maintain their own creative preferences and evaluation criteria, are the root cause of this issue. This manifests clearly in the network structures shown in Figure 21, where the 500-agent population with minimal sharing demonstrates qualitatively different behavioral patterns compared to smaller populations. These computational constraints highlight a crucial challenge in scaling social creativity models: the trade-off between population size, interaction complexity, and computational feasibility. While we attempted to demonstrate potential emergent creative behaviors in populations up to 500 agents, significantly larger than previous studies [6], the results suggest that further scaling may require fundamental architectural innovations.

The interest levels (Figure 13) demonstrate a consistent trend across configurations, stabilizing between 0.12 and 0.17, with the control condition maintaining the lowest stable interest at 0.12. This stabilization, occurring within the first 500 steps, suggests that the interest calculation mechanism functions effectively, regardless of population size. Communication and domain thresholds (Figure 15 and 16) exhibit more varied behavior, with larger populations showing higher initial thresholds that gradually decrease over time. Notably, the 500-agent population with share parameter 1 displayed unique characteristics, starting at 0.4 and converging to 0.32 after 4000 steps, which is significantly higher than other configurations. The behavior of this simulation indicates that larger populations with restricted sharing develop more selective communication patterns.

Inherently, the interest values are arbitrary and carry no real value, only to the agents themselves, and as long as the same novelty detection mechanism is applied consistently over all simulations. Regardless of the absolute values observed in our experiments, the key factor is that the interest calculation provides sufficient differentiation for agents to make

meaningful decisions about artifact sharing and evaluation. The stability we see in interest ranges could be entirely different at larger scales, but this wouldn't invalidate the system dynamics as long as the relative differences between interest values stay significant.

Visual inspection of the network structures, specifically Figure 19, appears to show clearly defined cliques. However, when compared to the modularity value, a criterion we set ourselves, the results fall short. Our requirement for emergent behavior was to maintain a modularity value above 0.3 for at least 1000 steps. In the specific simulation, the modularity peaked briefly at 0.32 but quickly dropped well below 0.1 before stabilizing. This rapid decline indicates that while the visual clustering suggests organization, the quantitative measure of modularity does not support the claim of stable emergent properties.

For the 500 agent simulations (Figure 21), the clustering appears more diffuse, with intermediary agents acting as bridges between groups. This creates the appearance of dynamic interactions and potential knowledge transfer. However, the modularity values for these simulations remained consistently below 0.25, failing to meet the stability criterion. The lack of sustained modularity above 0.3 further challenges the evidence for emergent properties in larger populations.

The communication patterns and threshold dynamics provide additional insight. In the 100 agent simulations, communication thresholds stabilized around 0.18 after 500 steps, as shown in Figure 15. Domain thresholds followed a similar trend, stabilizing at 0.15. These values indicate some degree of stabilization, but the lack of separation between configurations suggests that the dynamics are not scale-dependent as hypothesized.

In the 500 agent simulation, thresholds started higher but gradually decreased over time. Communication thresholds settled at 0.22, while domain thresholds converged around 0.19 (Figure 16). Despite these trends, the larger population introduced more variability between the generated artifacts, which could be argued as a form of emergent property due to increasing complexity. Unlike smaller populations, where interactions are more predictable and localized, larger populations appear to generate more dynamic and less uniform patterns of behavior. This increasing complexity might represent an emergent property of the network, highlighting how scale introduces new dynamics that are not evident in smaller simulations. While these patterns do not align with our original criteria for stable modularity, they provide evidence that larger-scale systems behave qualitatively differently, which aligns with some definitions of emergent behavior.

Before coming to a conclusion, we analyze the extracted network statistics. Eigenvector centrality decreases as population size grows, indicating a shift toward distributed influence across agents. This aligns with observations of reduced centralization in larger networks.

Path length analysis shows shorter average paths in smaller networks, which appears to be very low. This analysis would mean that the agents are interconnected in a way that you could reach any agent within 1-2 links, meaning that the clusters are not very deep.

The length of communities over time does not demonstrate anything remarkable, other than the fact that the highest values also reflect on the modularity and the average path length. For example, the highest number of communities is in a population of 20, highlighting an issue with the Louvain method on lower populations (Or all the agents are too separate that they form individual communities).

7 Conclusions

This research examined whether scaling multi-agent systems to larger populations naturally yields emergent creative behaviors and stylistic evolution without artificial constraints. The findings show that while larger populations produce more complex and fragmented interaction patterns, stable clique formation did not consistently meet the strict modularity criterion for conclusive evidence of long-term clustering. Nonetheless, increasing the number of agents led to novel interaction structures, including multi-hub topologies and isolated micro-clusters, which differ from small-scale results and indicate partially new dynamics as hypothesized. Moreover, in contrast to H1, high-modularity communities were not conclusively sustained, though larger populations displayed persistent subgroups and bridging agents not present in smaller systems. Regarding H2, scaling effects did reveal distinct interaction modes that were not observed with fewer than 20 agents, suggesting that large-scale modeling captures creative behaviors otherwise absent in small-scale frameworks. Despite these promising indicators, computational overhead limited the range of possible sharing parameters in populations over 500 agents, restricting a full exploration of the system's capabilities and potential emergent properties.

Overall, we tested whether large-scale multi-agent systems could form emergent creative cliques without artificial constraints, finding that scaling to hundreds of agents did uncover new behaviors (most notably multi-hub networks and ephemeral clusters) yet stable, high-modularity groupings never fully materialized. These findings indicate that while scaling fosters increased complexity, fundamental computational constraints and agent-level interaction rules may inhibit the formation of truly persistent creative communities. Future work could explore adaptive social policies, selective sharing, and novel architectural optimizations to determine if stable emergent creativity can be sustained at scale. Ultimately, these results suggest that scaling can unlock new modes of social creativity, provided computational limits are addressed, opening pathways to more nuanced explorations of large-scale creative phenomena.

8 Future work

Future work could enhance the computational architecture by refining data handling and expanding parallelization. Asynchronous data collection, coupled with a more efficient novelty detection framework, may reduce the overhead that limited the scale of current experiments. Upgrading from single-GPU settings to GPU clusters or multi-threaded environments could also enable investigations of larger populations and longer simulation runs, revealing whether more pronounced or stable community structures emerge under heavier computational budgets.

Another research direction lies in broadening the simulation's parameter space to explore more sophisticated social policies. Selective artifact sharing, domain policies that govern which artifacts remain in circulation, and repository mechanisms that induce forgetting could all shift the dynamics of community formation. Novel approaches to agent interactions, such as chained or conditional communication protocols, might highlight whether high-modularity cliques become more persistent under more context-specific decision rules.

Lastly, future implementations could introduce more diverse agent types with specialized roles in the creative process. For instance, some agents could focus on critique, offering more

stringent evaluations, while others emphasize exploration of new artifact forms. Further varying the image generation component and enabling cross-domain or external artifact sharing might provide different perspectives.

References

[1] M. Csikszentmihalyi, "16 implications of a systems perspective for the study of creativity," *Handbook of creativity*, vol. 313, 1999.

[2] G. Fischer, E. Giaccardi, H. Eden, M. Sugimoto, and Y. Ye, "Social creativity: Making all voices heard," *International Journal of Human-Computer Studies*, vol. 63, no. 4-5, pp. 369–387, 2005.

[3] R. K. Sawyer, "Explaining creativity: The science of human innovation," *Creativity Research Journal*, vol. 18, no. 3, pp. 391–394, 2006.

[4] B. A. Hennessey and T. M. Amabile, "Creativity," *Annual Review of Psychology*, vol. 61, pp. 569–598, 2010.

[5] V. P. Glăveanu, "Paradigms in the study of creativity: Introducing the perspective of cultural psychology," *New Ideas in Psychology*, vol. 28, no. 1, pp. 79–93, 2010.

[6] R. Saunders and J. S. Gero, "Curious agents and situated design evaluations," *AI EDAM*, vol. 18, no. 2, pp. 153–161, 2004.

[7] R. Saunders, "Towards autonomous creative systems: A computational approach," *Cognitive Computation*, vol. 4, pp. 216–225, 2012.

[8] R. Saunders and O. Bown, "Computational social creativity," *Artificial Life*, vol. 21, no. 3, pp. 366–378, 2015.

[9] S. Linkola, T. Takala, and H. Toivonen, "Novelty-seeking multi-agent systems," Association for Computational Creativity, 2016, pp. 1–8.

[10] G. A. Wiggins, "A preliminary framework for description, analysis and comparison of creative systems," *Knowledge-based systems*, vol. 19, no. 7, pp. 449–458, 2006.

[11] R. Saunders and J. S. Gero, "Artificial creativity: A synthetic approach to the study of creative behaviour," *Computational and Cognitive Models of Creative Design V, Key Centre of Design Computing and Cognition, University of Sydney, Sydney*, pp. 113–139, 2001.

[12] J. Wei, Y. Tay, R. Bommasani, *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.

[13] M. A. Boden, *The creative mind: Myths and mechanisms*, 2004.

[14] R. L. Goldstone and T. M. Gureckis, "Collective behavior," *Topics in cognitive science*, vol. 1, no. 3, pp. 412–438, 2009.

[15] J. S. Gero and M. L. Maher, *Modeling creativity and knowledge-based creative design*. Psychology Press, 2013.

[16] E. Watson, "Who or what creates? a conceptual framework for social creativity," *Human Resource Development Review*, vol. 6, no. 4, pp. 419–441, 2007.

[17] M. Csikszentmihalyi, "Motivation and creativity: Toward a synthesis of structural and energistic approaches to cognition," *New Ideas in psychology*, vol. 6, no. 2, pp. 159–176, 1988.

[18] M. Csikszentmihalyi and M. Csikszentmihalyi, *Society, culture, and person: A systems view of creativity*. Springer, 2014.

[19] Y.-T. Liu, "Creativity or novelty?: Cognitive-computational versus social-cultural," *Design Studies*, vol. 21, no. 3, pp. 261–276, 2000.

[20] S. Hanna, "Where creativity comes from," *Proceedings of Human Interaction 2005*, pp. 45–70, 2005.

[21] R. Saunders, "Multi-agent-based models of social creativity," *Computational creativity: The philosophy and engineering of autonomously creative systems*, pp. 305–326, 2019.

[22] G. Ritchie, "Some empirical criteria for attributing creativity to a computer program," *Minds and Machines*, vol. 17, no. 1, pp. 67–99, 2007.

[23] S. Pidhorskyi, R. Almohsen, D. A. Adjeroh, and G. Doretto, "Generative probabilistic novelty detection with adversarial autoencoders," *arXiv preprint arXiv:1807.02588*, 2018.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[25] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, "Adversarially learned one-class classifier for novelty detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3379–3388. DOI: 10.1109/CVPR.2018.00356.

[26] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014. DOI: 10.1016/j.sigpro.2013.12.026.

[27] D. K. Simonton, "Taking the us patent office creativity criteria seriously: A quantitative three-criterion definition and its implications," *Creativity Research Journal*, vol. 24, no. 2-3, pp. 97–106, 2012.

[28] M. A. Boden, "Creativity and artificial intelligence," *Artificial Intelligence*, vol. 103, no. 1-2, pp. 347–356, 1998.

[29] C. G. Langton, "Studying artificial life with cellular automata," *Physica D: Nonlinear Phenomena*, vol. 22, no. 1-3, pp. 120–149, 1986.

[30] M. Csikszentmihalyi, "Implications of a systems perspective for the study of creativity," *Handbook of creativity*, vol. 313, p. 321, 1999.

[31] L. Gabora, "Modeling cultural dynamics," *Proceedings of the Association for the Advancement of Artificial Intelligence*, pp. 1–8, 2010.

[32] H. Sayama and R. Sinatra, "Robust self-organized meta-population dynamics on networks," *Physical Review E*, vol. 82, no. 1, p. 016 111, 2010.

[33] S. Fortunato, "Community detection in graphs," *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[34] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, P10008, 2008.

[35] V. A. Traag, L. Waltman, and N. J. Van Eck, "From louvain to leiden: Guaranteeing well-connected communities," *Scientific reports*, vol. 9, no. 1, pp. 1–12, 2019.

[36] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *Proceedings of the national academy of sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[37] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Physical Review E*, vol. 74, no. 1, p. 016 110, 2006.

[38] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[39] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, 2016.

[40] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2017.

[41] R. K. Sawyer, "Explaining creativity: The science of human innovation," *Oxford university press*, 2012.

[42] E. B. Kennedy, "Modeling collective creativity using evolutionary algorithms," *Neural Computing and Applications*, vol. 30, no. 2, pp. 385–394, 2018.

[43] A. J. Cropley, *Creativity in education and learning: A guide for teachers and educators*. Routledge, 2018.

[44] M. Csikszentmihalyi, "The systems model of creativity and its applications," *The Wiley handbook of genius*, pp. 533–545, 2014.

[45] V. P. Glăveanu, "Distributed creativity: Thinking outside the box of the creative individual," *Springer*, 2014.

[46] D. Centola, "The spread of behavior in an online social network experiment," *Science*, vol. 329, no. 5996, pp. 1194–1197, 2010.

[47] B. Uzzi and J. Spiro, "Collaboration and creativity: The small world problem," *American Journal of Sociology*, vol. 111, no. 2, pp. 447–504, 2005.

[48] J. Henrich, "Cultural transmission and the diffusion of innovations: Adoption dynamics indicate that biased cultural transmission is the predominate force in behavioral change," *American Anthropologist*, vol. 103, no. 4, pp. 992–1013, 2001.

[49] M. Muthukrishna, B. W. Shulman, V. Vasilescu, and J. Henrich, "Sociality influences cultural complexity," *Proceedings of the Royal Society B: Biological Sciences*, vol. 281, no. 1774, p. 20 132 511, 2014.

[50] M. Derex, M.-P. Beugin, B. Godelle, and M. Raymond, "Population size affects cultural evolution in nonintuitive ways," *Proceedings of the Royal Society B: Biological Sciences*, vol. 280, no. 1767, p. 20 123 072, 2013.

[51] E. M. Rogers, *Diffusion of innovations*. Simon and Schuster, 2010.

[52] S. Perry, "Social learning of foraging techniques in wild white-faced capuchin monkeys (cebus capucinus)," *Animal Cognition*, vol. 6, no. 2, pp. 77–87, 2003.

[53] M. S. Granovetter, "The strength of weak ties," *American Journal of Sociology*, vol. 78, no. 6, pp. 1360–1380, 1973.

9 Appendices

Category	Operation	Description
Constants	Q_IDENTITY	Quaternion constant representing (1, 0, 0, 0).
	Q_I	Quaternion constant representing (0, 1, 0, 0).
	Q_J	Quaternion constant representing (0, 0, 1, 0).
	Q_K	Quaternion constant representing (0, 0, 0, 1).
Basic Arithmetic	Addition	Component-wise addition of two quaternions.
	Subtraction	Component-wise subtraction.
	Multiplication	Hamilton product of two quaternions.
	Division	Division of quaternions using the conjugate.
	Power	Raises a quaternion to a random real power.
	Cube	Computes the cube of a quaternion.
Trigonometric and Hyperbolic	sin	Quaternion sine function.
	cos	Quaternion cosine function.
	tan	Quaternion tangent function.
	sinh	Quaternion hyperbolic sine function.
	cosh	Quaternion hyperbolic cosine function.
Special	exp	Quaternion exponential function.
	log	Quaternion natural logarithm.
	sqrt	Quaternion square root function.
	abs	Computes the absolute value.
	mod2	Computes the modulo operation with 2.0.
	floor	Rounds down to the nearest integer.
Transformations	normalize	Scales the quaternion to unit length.
	conjugate	Computes the conjugate of the quaternion.
	rotate45	Rotates a quaternion by 45 degrees.
	ripple	Generates ripple patterns based on radial distance.
	swirl	Generates swirl patterns based on radial and angular coordinates.
	blend	Smoothly blends between two quaternions.
Coordinate and Logical Operations	coord	Converts x, y coordinates into a quaternion.
	imin	Selects the quaternion with the smaller norm.
	imax	Selects the quaternion with the larger norm.
	ilog	Logarithm with emphasis on the imaginary components.
	isin	Sine function with emphasis on the imaginary components.

Table 3: Quaternion Operations Used in Artifact Generation