BFAI: Brain-Forest Adaptive Intelligence

A Biomimetic Cognitive Architecture with Structural Polymorphism and Ramanujan Mathematical Backbone

Rushikesh More
Independent Researcher
rushikeshmore271@gmail.com

January 2026

Abstract

This position paper proposes Brain-Forest Adaptive Intelligence (BFAI), a theoretical framework for energy-efficient AI that combines biomimetic design principles with mathematical structures from Srinivasa Ramanujan's work. The architecture features a dual-module system: a lightweight "Brain" for rapid query classification and a sparse "Forest" of heterogeneous experts for deep processing, connected via biologically-inspired routing. We explore how seven of Ramanujan's mathematical contributions—including Ramanujan Graphs, partition functions, and continued fractions—might provide theoretical grounding for specific architectural components.

Scope and Honesty: This paper presents a *theoretical framework*, not empirical results. Approximately 60% of proposed components build on established research, 30% represent novel combinations requiring validation, and 10% are speculative connections requiring significant formalization. We explicitly acknowledge limitations regarding hardware-software co-design, differentiability of certain components, and the gap between theoretical projections and practical implementation. This work aims to inspire research directions rather than claim proven solutions.

Keywords: Mixture of Experts, Sparse Computation, Ramanujan Graphs, Energy-Efficient AI, Biomimetic Architecture, Cognitive Architecture

1 Glossary of Terms

To ensure clarity, we define key terms as used throughout this paper:

Term	Definition
Brain Module	Lightweight first-stage classifier (∼50M parameters) for rapid query analysis and routing. Inspired by Kahneman's "System 1" fast thinking.
Forest Module	Collection of heterogeneous specialized experts for deep computation. Inspired by "System 2" slow, deliberate thinking.
Cognitive Primitive	One of 108 fundamental query types used to classify incoming requests. Inspired by Sanskrit alphabet organization.
Mycelium Router	Routing layer connecting Brain to Forest, structured as a Ramanujan Graph for optimal connectivity. Named after fungal networks.
Watcher	Output validation module that monitors response quality and triggers refinement if confidence is low.
Oxygen Feedback	Iterative refinement loop where low-confidence outputs are reprocessed with additional computation.
Heterogeneous MoE	Mixture of Experts where different experts have different *architectures*, not just different weights.
Ramanujan Graph	A k-regular graph achieving optimal spectral expansion (Alon-Boppana bound).

Table 1: Glossary of key terms used in BFAI

2 Introduction

The rapid advancement of large language models (LLMs) has created an urgent sustainability challenge. Training GPT-4-scale models requires an estimated 50 GWh of electricity [Patterson et al., 2022], and inference costs scale with user adoption. Current approaches to efficiency—quantization, pruning, distillation—treat the model as a monolithic structure to be compressed, rather than reconsidering the fundamental architecture.

This paper proposes an alternative paradigm: **adaptive, heterogeneous computation** guided by mathematical structures that are provably optimal for their intended purpose. We draw inspiration from two sources:

1. **Biological systems:** The human brain uses approximately 20 watts yet outperforms AI on most cognitive tasks. It achieves this through sparse activation, specialized regions, and adaptive resource allocation.
2. **Ramanujan's mathematics:** The Indian mathematician Srinivasa Ramanujan (1887–1920) discovered structures—graphs, partitions, continued fractions—that are provably optimal for connectivity, distribution, and approximation.

Our central hypothesis is that **mathematically optimal structures may provide better architectural primitives than learned or random structures** for specific components of AI systems. This is not a claim that mathematics can replace learning, but that mathematical optimality might constrain the search space productively.

2.1 Positioning: Beyond Standard MoE

While BFAI utilizes sparse activation common in Mixture-of-Experts (MoE) systems, it departs from the standard MoE paradigm in four fundamental ways:

1. **Structural Polymorphism:** Unlike "size heterogeneity" (making some experts larger), BFAI proposes *architectural heterogeneity*—fundamentally different neural circuits for different cognitive tasks.
2. **Deterministic Mathematical Backbone:** Rather than learned gating via latent embeddings, BFAI routes through provably optimal Ramanujan Graph topology.
3. **Axiomatic Cognitive Basis:** The 108 primitives create a neuro-symbolic layer that classifies query *intent* before token-level processing.
4. **Recurrent Residual Refinement:** The Oxygen Feedback loop transforms static feed-forward inference into adaptive reasoning.

This positions BFAI not as an MoE extension, but as a **Biomimetic Cognitive Architecture** that uses MoE as one component. The "Forest" metaphor is technically accurate: standard MoE is a monoculture plantation (same trees, different sizes), while BFAI is a literal forest (oaks for strength, willows for flexibility, vines for connection).

2.2 Scope and Limitations

This is a **position paper** presenting a theoretical framework. We do not provide:

- Empirical benchmarks or trained models
- Formal proofs of all mathematical connections
- Hardware implementations or energy measurements

We do provide:

- A coherent architectural vision with clear components
- Explicit confidence levels for each claim
- Honest assessment of limitations and open questions
- A research roadmap for validation

3 Background and Related Work

3.1 Mixture of Experts

Sparse Mixture of Experts (MoE) architectures activate only a subset of parameters per input, reducing inference cost. Switch Transformer [Fedus et al., 2022] demonstrated that MoE can scale to trillion-parameter models while maintaining efficiency. Mixtral [Jiang et al., 2024] showed that MoE can match dense model quality with 5× fewer active parameters.

Critical Distinction—Size vs Structural Heterogeneity: Recent "Heterogeneous MoE" papers (HMoE, MoHGE, 2024) focus on *size heterogeneity*—making some experts larger than others. BFAI proposes *structural heterogeneity*—fundamentally different neural architectures for different cognitive tasks. This is analogous to the difference between a plantation (same trees, different sizes) and a forest (different species for different functions).

BFAI's departure from MoE: While BFAI uses sparse activation, it is better characterized as a **Cognitive Architecture** that incorporates MoE as one component, rather than an MoE extension. The key differences:

- **Routing:** Learned latent embeddings → Ramanujan Graph topology
- **Experts:** Homogeneous weights → Heterogeneous architectures
- **Inference:** Feed-forward → Recurrent refinement (Oxygen Feedback)
- **Basis:** Token embeddings → Cognitive primitives (neuro-symbolic)

3.2 Adaptive Computation

Adaptive Computation Time (ACT) [Graves, 2016] allows models to vary computation per input via learned halting probabilities. Universal Transformers [Dehghani et al., 2018] apply shared weights iteratively until convergence.

BFAI's difference: Rather than iterating the same computation, we propose routing to *different* experts and *unfolding* precision levels based on confidence. The "Oxygen Feedback" mechanism is conceptually similar to ACT but operates at the expert-selection level rather than layer depth.

3.3 Ramanujan Graphs in Machine Learning

Recent work has applied Ramanujan Graphs to neural network design. Hoory et al. [2006] established the theoretical foundations of expander graphs. SpectralFly (2024) used Ramanujan Graph topology for communication-efficient distributed training. DeepMind's AlphaEvolve (2025) discovered new Ramanujan Graph constructions via AI search, demonstrating renewed interest in these structures.

BFAI's contribution: We extend the application of Ramanujan Graphs from layer connectivity to *routing topology* in MoE systems, and explore connections to other Ramanujan mathematical structures.

3.4 Cognitive Architecture Inspiration

Kahneman's dual-process theory [Kahneman, 2011] distinguishes "System 1" (fast, intuitive) from "System 2" (slow, deliberate) thinking. Several AI architectures have drawn on this distinction, though typically as metaphor rather than structural constraint.

BFAI's interpretation: The Brain Module implements System 1 (rapid classification), while the Forest Module implements System 2 (deep reasoning). The Watcher implements metacognition (knowing when System 2 is needed).

4 Architecture Overview

BFAI processes queries through a six-step pipeline:

Step	Component	Function	Status
1	Brain Module	Classify query into cognitive primitive	**[Established]** Classifier design
2	Clarity Layer	Assess query complexity (1–5 scale)	**[Established]** Confidence scoring
3	Mycelium Router	Route to appropriate expert(s)	**[Novel]** Ramanujan topology
4	Forest Module	Execute specialized computation	**[Novel]** Heterogeneous MoE
5	Watcher	Validate output confidence	**[Established]** Output monitoring
6	Oxygen Feedback	Refine if confidence $<$ threshold	**[Novel]** Adaptive refinement

Table 2: BFAI processing pipeline with component status

5 Brain Module: Rapid Classification

The Brain Module is a lightweight transformer ($\sim$50M parameters) that classifies incoming queries into one of 108 **cognitive primitives**.

5.1 The 108 Cognitive Primitives

The number 108 is inspired by the organizational structure of the Sanskrit alphabet, which groups sounds by articulatory features. We do not claim linguistic equivalence, but use this as a design heuristic for organizing query types.

Category	Count	Examples
Factual Retrieval	15	Definition, Date, Location, Quantity, Identity
Logical Reasoning	15	Deduction, Induction, Comparison, Causation
Creative Generation	15	Narrative, Poetry, Dialogue, Description
Mathematical	15	Arithmetic, Algebra, Geometry, Statistics
Code & Technical	15	Syntax, Debug, Optimize, Architecture
Agentic Operations	**18**	**Planning, Tool-Use, Search, Execute, Delegate, Verify**
Meta & Clarification	15	Ambiguity, Context, Reformulation, Scope

Table 3: Cognitive primitive categories ($15 \times 6 + 18 = 108$). Note: Agentic Operations added to align with 2025 industry convergence toward autonomous AI agents.

2025 Update—Agentic Primitives: Following industry convergence (OpenAI, Anthropic, DeepMind) toward agentic AI, we include a dedicated "Agentic Operations" category covering: Planning, Tool-Use, Web-Search, Code-Execution, Task-Delegation, and Result-Verification. This positions BFAI as a blueprint for *autonomous agents*, not just conversational assistants.

Implementation note: The 108 primitives are implemented as learned embeddings, not hard-coded rules. The Sanskrit-inspired categorization provides initial structure, but the model can learn to adjust boundaries during training.

Critical clarification—Soft vs Hard Routing: The primitives function as **soft priors**, not hard assignments:

- The Brain Module outputs a *probability distribution* over all 108 primitives
- The Mycelium Router uses this distribution as a *weighted prior* for expert selection
- Experts outside the top primitive can still be activated if the router's learned weights favor them
- This prevents the "Taxonomy Trap" where misclassification cascades to downstream errors

Risk acknowledged: History shows human-designed taxonomies (Cyc, WordNet) are often outperformed by unsupervised latent representations. The soft routing approach mitigates but does not eliminate this risk. Empirical comparison with fully-learned routing is essential future work.

5.2 Clarity Assessment

The Brain Module also produces a **Clarity Score** (1–5) indicating query complexity:

- **Level 1:** Direct lookup ("What is the capital of France?")
- **Level 3:** Multi-step reasoning ("Compare GDP growth rates...")
- **Level 5:** Open-ended synthesis ("Design a sustainable city...")

This score determines how many experts the Forest Module will activate.

6 Mycelium Router: Ramanujan Graph Topology

The routing layer connecting Brain to Forest is structured as a **Ramanujan Graph**—a k-regular graph that achieves optimal spectral expansion.

6.1 Why Ramanujan Graphs?

[Established] Ramanujan Graphs provide maximum connectivity with minimum edges. For a k-regular graph, the Alon-Boppana bound states that the second-largest eigenvalue $\lambda_2 \geq 2\sqrt{k-1} - o(1)$. Ramanujan Graphs achieve $\lambda_2 \leq 2\sqrt{k-1}$, meaning they are optimally expanding.

Practical implication: In a routing network, this means any query can reach any expert through minimal hops, while the network uses far fewer connections than a fully-connected graph.

Topology	Connections	Max Path Length
Fully Connected (64 experts)	2,016	1
Random Sparse	192	4–8 (variable)
Ramanujan Graph (k=6)	192	3 (guaranteed)

Table 4: Routing topology comparison

[Novel] Applying Ramanujan Graphs to MoE routing topology is, to our knowledge, a novel contribution. While prior work has used these graphs for layer connectivity, we propose using them to structure the expert selection network.

6.2 Self-Optimizing Topology via AI Discovery

While we propose Ramanujan Graph topology, the *specific* adjacency matrix of the Mycelium Router need not be static. Recent advances in AI-driven mathematical discovery suggest a powerful extension:

AlphaEvolve Integration: Google DeepMind's AlphaEvolve framework (2025) demonstrated AI agents discovering novel Ramanujan Graph constructions, including a 163-node graph with spectral properties previously unknown. We propose that:

- The initial Mycelium Router uses a known Ramanujan construction (e.g., LPS graphs)
- An evolutionary discovery agent can optimize the specific topology for a given task distribution
- The objective: minimize spectral gap while maximizing routing efficiency for the observed primitive distribution

This adds a layer of **"Self-Optimizing Architecture"** to BFAI—the routing topology itself can evolve to match the workload.

7 Forest Module: Heterogeneous Experts

Core Innovation: Unlike standard MoE (Switch Transformer, Mixtral, DeepSeek) where all experts share identical architecture with different weights, BFAI proposes **heterogeneous experts**—structurally different neural networks optimized for fundamentally different cognitive tasks.

This is arguably **BFAI's most novel and defensible contribution**. While Ramanujan mathematics provides theoretical grounding, the biomimetic insight that "different tasks require different computational structures" is the architectural innovation with the clearest path to empirical validation.

Expert Type	Architecture	Primitives Served	Parameters
Factual	Retrieval-augmented	Lookup, Definition	200M
Logical	Chain-of-thought	Deduction, Comparison	500M
Creative	High-temperature sampling	Narrative, Poetry	300M
Mathematical	Symbolic + Neural	Arithmetic, Proof	400M
Code	Syntax-aware transformer	Debug, Generate	400M
Meta	Reflection architecture	Clarification, Scope	200M

Table 5: Heterogeneous expert specifications

Sparsity: Following Ramanujan Graph structure, each expert maintains only 5% of fully-connected weights, achieving 95% sparsity while preserving information flow.

Hardware caveat: [Future Work] Current GPUs are optimized for dense tensor operations. Sparse matrix multiplication often achieves lower throughput due to irregular memory access. True efficiency gains may require sparse-optimized hardware (neuromorphic chips, FPGAs) or future GPU architectures with better sparse support.

8 Watcher and Oxygen Feedback

8.1 The Watcher Module

The Watcher monitors output quality through multiple signals:

- **Confidence score:** Softmax probability of top prediction
- **Coherence check:** Internal consistency of generated response
- **Primitive alignment:** Does output match expected primitive type?

If confidence falls below threshold (default: 0.7), the Watcher triggers refinement.

8.2 Oxygen Feedback Loop

The refinement mechanism is named "Oxygen Feedback" by analogy to biological systems that increase oxygen delivery under stress.

1. **Level 1 refinement:** Activate additional experts ($1 \rightarrow 3$)
2. **Level 2 refinement:** Increase computation precision (see Section 10.2.2)
3. **Level 3 refinement:** Engage full Forest Module (fallback to dense)

[Novel] The Oxygen Feedback mechanism combined with heterogeneous MoE is our primary architectural contribution. This design choice is independently supported by recent work from Liu et al. [2025], which demonstrates that hallucinations arise from inaccurate internal world modeling—validating BFAI's approach of re-routing low-confidence outputs through additional expert consultation as a principled mitigation strategy. The specific implementation details require further formalization.

9 Training Strategy

A critical gap in many MoE proposals is the **differentiability problem**: how do you train a router that makes discrete expert selections?

9.1 The Routing Gradient Problem

Standard backpropagation requires smooth, differentiable operations. But "selecting Expert A over Expert B" is a discrete, non-differentiable decision. Without a training strategy, BFAI would be a static architecture, not a learning system.

9.2 Proposed Solutions

We propose two complementary approaches:

1. Gumbel-Softmax Relaxation [Jang et al., 2016]

During training, replace hard expert selection with a continuous relaxation:

$$y_i = \frac{\exp((\log \pi_i + g_i)/\tau)}{\sum_j \exp((\log \pi_j + g_j)/\tau)} \tag{1}$$

where g_i are Gumbel noise samples and τ is a temperature parameter. As $\tau \to 0$, this approaches hard selection. This allows gradients to flow through the Mycelium Router.

2. Reinforcement Learning (REINFORCE / PPO)

Treat expert selection as a policy decision:

- **State:** Query embedding + primitive distribution
- **Action:** Which expert(s) to activate
- **Reward:** Output quality (confidence) minus energy cost

This is particularly suitable for the Oxygen Feedback loop, where the decision to "refine or accept" can be learned via policy gradient.

9.3 Training the Heterogeneous Experts

Since experts have different architectures, we propose:

1. **Stage 1:** Pre-train each expert independently on domain-specific data
2. **Stage 2:** Freeze experts, train only the Brain Module and Router
3. **Stage 3:** Fine-tune end-to-end with low learning rate on experts

This staged approach prevents the router from "collapsing" to always select one expert.

10 Ramanujan Mathematical Framework

We explore seven of Ramanujan's mathematical contributions as potential architectural primitives. We explicitly tier these by confidence level:

Formula	Application	Confidence	Notes
Ramanujan Graphs	Router topology	HIGH	Published validation in ML
Alon-Boppana Bound	Optimality proof	HIGH	Proven mathematical theorem
Partition Function $p(n)$	Memory allocation	MEDIUM	Novel application, needs testing
Continued Fractions	Precision refinement	MEDIUM	Conceptually strong, needs formalization
Infinite Series	Convergence logic	MEDIUM	Applies principle, not formula directly
Ramanujan Sums $c_n(k)$	Pattern detection	LOW	Signal $\to$ NLP transfer is hypothetical
Mock Theta Functions	Hallucination filter	SPECULATIVE	Beautiful metaphor, no formal connection

Table 6: Ramanujan mathematical components with confidence levels

10.1 High-Confidence Components

10.1.1 Ramanujan Graphs for Routing

The Mycelium Router uses Ramanujan Graph topology as described in Section 6. This is **[Established]** well-established in graph theory and has been validated in ML contexts.

10.1.2 Alon-Boppana Bound for Optimality

For a k-regular graph with n vertices, the second eigenvalue satisfies:

$$\lambda_2 \geq 2\sqrt{k-1} - o(1) \tag{2}$$

Ramanujan Graphs achieve this bound exactly, making them **[Established]** provably optimal for the routing connectivity problem.

10.2 Medium-Confidence Components

10.2.1 Partition Functions for Memory

Ramanujan's work on the partition function $p(n)$ and its congruences:

$$p(5n+4) \equiv 0 \pmod 5 \tag{3}$$
$$p(7n+5) \equiv 0 \pmod 7 \tag{4}$$
$$p(11n+6) \equiv 0 \pmod{11} \tag{5}$$

[Novel] We propose using these congruences to structure memory allocation in the Forest Module, ensuring balanced distribution across memory banks. This is a novel application without empirical validation.

10.2.2 Continued Fractions for Refinement

Ramanujan's continued fraction representations provide optimal rational approximations with progressive accuracy:

$$x = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}} \tag{6}$$

[Novel] We propose "unfolding" continued fractions as a refinement mechanism:

- Level 0: Read first 2 coefficients (∼low precision, fast)
- Level 1: Read first 4 coefficients (∼medium precision)
- Level 2: Read all coefficients (∼full precision)

Formalization gap: This concept requires tensor-level definition. How coefficients map to weight precision, and how gradients flow through variable-depth fractions, are open questions.

10.3 Low-Confidence / Speculative Components

10.3.1 Ramanujan Sums

Ramanujan Sums $c_n(k) = \sum_{(j,n)=1} e^{2\pi i j k/n}$ are proven for periodic signal detection in signal processing.

[Speculative] Applying these to NLP assumes linguistic patterns have periodicity properties, which is not established. Language is hierarchical and stochastic, not periodic in the Fourier sense. We include this as a research direction, not a validated component.

10.3.2 Mock Theta Functions

Ramanujan's mock theta functions have deep properties related to modularity and "near-misses" of symmetric structures.

[Speculative] Using these for hallucination detection is currently a metaphor without mathematical formalization. The connection is aesthetic rather than functional. Making this differentiable for end-to-end training is an open research problem.

11 Energy Analysis

11.1 Theoretical Projections

Under idealized conditions (sparse-optimized hardware, perfect routing), BFAI targets significant efficiency improvements:

Metric	Dense Baseline	BFAI Target
Active Parameters	100%	5–15%
FLOPs per Inference	1.0×	0.1–0.3×
Theoretical Energy	1.0×	0.2–0.4×

Table 7: Theoretical efficiency targets (idealized conditions)

11.2 Hardware Reality Check

Critical caveat: These projections assume ideal sparse computation. Current GPUs achieve poor utilization for sparse operations due to:

- Irregular memory access patterns
- Thread divergence in warp execution
- Tensor cores optimized for dense GEMM

Honest estimate: On current hardware (NVIDIA H100), sparse MoE typically achieves 2–3× efficiency gains, not 5–10×. The larger gains require:

- Sparse-optimized accelerators (Cerebras, Graphcore)
- Neuromorphic hardware
- Future GPU architectures with native sparse support

11.3 The IO-Bound Reality

Critical insight: Modern LLM inference is often *memory-bandwidth bound*, not compute-bound. Even activating only 5% of weights requires loading the router and expert weights into GPU cache.

The Problem: If the Ramanujan Graph topology causes frequent cache misses (because "neighboring" experts in the graph are distant in physical memory), we might actually *increase* energy consumption due to:

- Tail latency from cache misses
- Idle power while waiting for memory transfers
- VRAM fragmentation from non-contiguous expert placement

Proposed Fix—Memory Co-location Strategy:

1. **Graph-aware VRAM Layout:** Store experts such that neighbors in the Ramanujan Graph are contiguous in physical memory

2. **Expert Clustering:** Group frequently co-activated experts (based on training statistics) into shared memory pages
3. **Prefetch Hints:** Use the Brain Module's primitive distribution to prefetch likely experts before routing completes

This is a critical implementation detail that determines whether theoretical sparsity translates to real energy savings.

11.4 Recommended Metric

Rather than percentage reduction, we recommend measuring **Inferences Per Watt (IPW)**:

$$\text{IPW} = \frac{\text{Successful inferences}}{\text{Energy consumed (Wh)}} \tag{7}$$

This metric captures real-world efficiency including routing overhead, memory access, and refinement loops.

12 Limitations and Open Questions

We explicitly acknowledge the following limitations:

12.1 Hardware-Software Gap

Sparse models often run slower than dense models on current hardware. The "95% sparse = 95% energy savings" equation does not hold without hardware co-design. This is a fundamental constraint, not a minor caveat.

12.2 Formalization Gaps

Several components lack tensor-level definitions:

- **Continued Fraction refinement:** How gradients flow through variable-depth computation
- **Mock Theta filter:** Currently non-differentiable; would function as post-processing
- **Partition-based memory:** Implementation details unspecified

12.3 Representation Risk

The 108 cognitive primitives may impose a "representation bottleneck" that harms performance on queries outside the predefined schema. Modern LLMs succeed by learning latent representations; fixed ontologies may be a step backward.

12.4 Mathematical Transfer

Several Ramanujan formulas are proven in specific mathematical contexts (number theory, signal processing) but their transfer to neural network optimization is hypothetical. The aesthetic appeal of these structures does not guarantee practical benefit.

12.5 Lack of Empirical Validation

This paper provides no trained models, benchmarks, or ablation studies. All claims are theoretical projections requiring experimental validation.

13 Future Work

The following directions emerged from expert feedback and represent promising extensions beyond this initial framework:

13.1 Near-Term Formalizations

1. **Highly Composite Number (HCN) Tensor Shapes:** Use Ramanujan's HCN sequences (12, 24, 360, 5040...) to constrain layer dimensions, ensuring sparse sub-matrices align with GPU cache hierarchies.
2. **INT8 Token Mixing:** Reframe Ramanujan Sums as integer-only transforms for the Brain Module, enabling low-power INT8 computation instead of FP32 attention.
3. **Continued Fraction Quantization:** Store weights as continued fraction coefficients, enabling progressive precision "unfolding" without re-computation.

13.2 Advanced Extensions

1. **Master Theorem Differentiability:** Use Ramanujan's Master Theorem to create differentiable sparsity loss, allowing the model to learn optimal sparsity levels.
2. **Rogers-Ramanujan State Space:** Structure long-context memory using Rogers-Ramanujan partition identities for $O(L \log L)$ memory scaling.
3. **Ramanujan Prime Initialization:** Initialize sparse connections using Ramanujan Prime sequences to mathematically guarantee information flow.

13.3 Application Domains

BFAI may be most applicable to:

- **Edge AI / On-Device:** Where energy constraints are absolute
- **Specialized Domains:** Where heterogeneous experts map naturally to distinct task types
- **Low-Resource Environments:** Where adaptive computation provides meaningful savings

For high-throughput data center inference on current GPUs, routing overhead may negate efficiency gains.

14 Honest Assessment Summary

Claim	Status	Notes
Ramanujan Graphs are optimal	Established	Mathematical theorem
MoE reduces computation	Established	Validated in Switch, Mixtral
Heterogeneous experts help	Novel	Untested hypothesis
108 primitives are useful	Novel	May help or harm; empirical question
Oxygen Feedback improves quality	Novel	Similar to ACT; needs validation
Energy savings of 60–80%	Projected	Requires ideal hardware
Mock Theta detects hallucination	Speculative	No formal connection
Ramanujan Sums for NLP	Speculative	Transfer not established

Table 8: Honest assessment of key claims

15 Conclusion

BFAI presents a theoretical framework for energy-efficient AI that combines biomimetic design with Ramanujan's mathematical structures. The core contributions, in order of novelty and defensibility, are:

1. **Heterogeneous MoE (Primary Innovation):** Experts with structurally different architectures for different task types—not just different weights, but different computational structures. This biomimetic specialization is the most novel and empirically testable contribution.
2. **Oxygen Feedback:** Adaptive refinement based on output confidence, allowing the system to "think harder" when uncertain.
3. **Brain-Forest Architecture:** A dual-module system implementing fast classification (System 1) and slow deliberation (System 2).
4. **Ramanujan Mathematical Grounding:** Exploration of optimal mathematical structures for routing topology, with varying confidence levels across different applications.

We are explicit about what this paper is and is not. It is **not** a validated architecture with empirical results. It **is** a coherent vision with clear research directions, honest assessment of uncertainties, and an invitation for collaboration.

The integration of Indian mathematical heritage—Ramanujan's work and Sanskrit-inspired organization—with modern AI efficiency challenges represents a unique approach. Whether this approach yields practical benefits is an empirical question we hope the research community will help answer.

Acknowledgments

I am grateful to Claude (Anthropic) for assistance with literature review, mathematical exposition, and document preparation throughout this research. Special thanks to Steven Y. Feng for his generous endorsement support, which made this arXiv submission possible.

References

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Kaiser, Ł. (2018). Universal Transformers. *arXiv preprint arXiv:1807.03819*.

Fedus, W., Zoph, B., and Shazeer, N. (2022). Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Graves, A. (2016). Adaptive Computation Time for Recurrent Neural Networks. *arXiv preprint arXiv:1603.08983*.

Hoory, S., Linial, N., and Wigderson, A. (2006). Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561.

Jang, E., Gu, S., and Poole, B. (2016). Categorical Reparameterization with Gumbel-Softmax. *arXiv preprint arXiv:1611.01144*.

Google DeepMind (2025). AlphaEvolve: A coding agent for scientific and algorithmic discovery. *Google DeepMind Technical Report*.

Liu, E., Gangal, V., Zou, C., Huang, X., Yu, M., Chang, A., Tao, Z., Kumar, S., and Feng, S. Y. (2025). A Unified Definition of Hallucination, Or: It's the World Model, Stupid. *arXiv preprint arXiv:2512.21577*.

Jiang, A. Q., Sablayrolles, A., Roux, A., et al. (2024). Mixtral of Experts. *arXiv preprint arXiv:2401.04088*.

Kahneman, D. (2011). *Thinking, Fast and Slow*. Farrar, Straus and Giroux.

Lubotzky, A., Phillips, R., and Sarnak, P. (1988). Ramanujan graphs. *Combinatorica*, 8(3):261–277.

Patterson, D., Gonzalez, J., Hölzle, U., et al. (2022). The Carbon Footprint of Machine Learning Training Will Plateau, Then Shrink. *IEEE Computer*, 55(7):18–28.

Ramanujan, S. (1919). Some properties of $p(n)$, the number of partitions of n. *Proceedings of the Cambridge Philosophical Society*, 19:207–210.

Ramanujan, S. (1927). *Collected Papers of Srinivasa Ramanujan*. Cambridge University Press.