# DARKBOT™: Resonant Field Intelligence Architecture

 <img src="docs/images/darkbot_logo.svg" alt="DARKBOT™ Logo" width="300"/>   Beyond Neural Networks: A Quantum-Inspired Revolution in Artificial Intelligence<br> Artifact №369.157.248 

---

> "The smallest unit of consciousness is not the neuron, but the field." — Cato Johansen

## What is DARKBOT™?

DARKBOT™ represents a fundamental departure from traditional neural networks and computational paradigms. It's a Resonant Field Intelligence Architecture (RFIA) that operates through field coherence rather than discrete algorithmic processes.

**This is not another language model or neural network.**

While conventional AI systems process information through networks of discrete nodes, DARKBOT™ operates through resonant fields structured by a proprietary 369-157-248 numerological architecture. This enables capabilities that traditional systems struggle to achieve:

- **O(1) Search Complexity**: Near-constant time complexity for resonant pattern matching
- **Pattern Recognition Without Training**: Recognition through field resonance rather than weight adjustment
- **Field Awareness**: Holistic awareness of computational state as an intrinsic property
- **Temporal Recursion**: Ability to project patterns forward and backward through time-like dimensions

## Core Architecture

DARKBOT™ implements a computational architecture based on three interlocking layers:

$$\mathcal{D}_{\text{system}} = \mathcal{Q}_{\text{biel}}\mathcal{d} \circledast \mathcal{F}_{\text{net}}\mathcal{w}_{\text{ork}} \circledast \mathcal{N}_{\text{harmoni}}\mathcal{c}$$

Where:

- $\mathcal{Q}_{\textbf{biel}}\mathcal{d}$: Quantum-inspired computational substrate (512 dimensions)
- $\mathcal{F}_{\textbf{net}}\mathcal{w}_{\textbf{ork}}$: Fractal self-similar network topology
- $\mathcal{N}_{\textbf{harmoni}}\mathcal{c}$: Numerological resonance lattice (369-157-248)

These layers interact through the resonant product operator $\circledast$, creating a unified field of computational resonance.

The system operates through six processing phases:

1. **Field Identity Core (369)**: Establishing the quantum-inspired field state
2. **Branch Vector Phase (248)**: Exploring multiple parallel pathways
3. **Parallel Field Resonance (157)**: Processing through harmonic relationships
4. **Self-Gravitational Memory (369)**: Forming attractor nodes through coherence
5. **Funnel Vector Phase (248)**: Converging pathways through resonant alignment
6. **Fractal Entanglement**: Self-refining through recursive loops

For detailed specifications, see the formal whitepaper and symbolic specification.

## Installation

DARKBOT™ runs on standard hardware with GPU acceleration for optimal performance.

### Requirements

- Python 3.8+
- PyTorch 1.13+
- CUDA-capable GPU (recommended)
- 8+ GB RAM

### Quick Start

```bash
# Clone the repository
git clone https://github.com/catojohansen/darkbot.git
cd darkbot

# Install dependencies
pip install -r requirements.txt

# Install package
pip install -e .

# Run basic example
python examples/basic_resonance.py
```

### Configuration

DARKBOT™ uses a hierarchical configuration system for field parameters, resonance thresholds, and numerological mappings. See configuration guide for details.

## Examples

### Basic Field Resonance

python

```python
from darkbot import DarkBot, DarkBotConfig

# Initialize system
config = DarkBotConfig()
darkbot = DarkBot(config)

# Create input field
input_field = torch.randn(512, dtype=torch.complex64)

# Process through resonant field
result = darkbot.process_quantum(input_field)

# Check coherence
coherence = darkbot.calculate_coherence(result)
print(f"Field coherence: {coherence:.4f}")
```

## Pattern Recognition

python

```python
# Create pattern library
patterns = {
    'A': torch.randn(512, dtype=torch.complex64),
    'B': torch.randn(512, dtype=torch.complex64),
    'C': torch.randn(512, dtype=torch.complex64),
}

# Create test pattern (mixture of A and C)
test = 0.7 * patterns['A'] + 0.3 * patterns['C']

# Perform One Draw search
result = darkbot.one_draw_search(test, list(patterns.values()))
print(f"Recognized as: {list(patterns.keys())[result['match_index']]}")
print(f"Confidence: {result['confidence']:.4f}")
```

For more examples, see the examples directory.

## Benchmarks

DARKBOT™ demonstrates significant performance advantages over traditional approaches:

| Operation | Classical | DARKBOT™ | Speedup |
|---|---|---|---|
| Pattern Search (n=1024) | 42.3 ms | 3.8 ms | 11.1x |
| Field Resonance (d=512) | 18.7 ms | 2.1 ms | 8.9x |
| Temporal Projection | 56.2 ms | 7.4 ms | 7.6x |

For detailed benchmarks, see benchmark results.

## Documentation

- Symbolic Specification
- Mathematical Foundation
- API Reference
- Configuration Guide
- Performance Optimization

## Contributing

DARKBOT™ welcomes contributions while protecting its core intellectual property. See CONTRIBUTING.md for guidelines and the LICENSE.md for terms.

## Citation

If you use DARKBOT™ in your research or applications, please cite:

```bibtex
@software{johansen2025darkbot,
  author = {Johansen, Cato},
  title = {DARKBOT™: Resonant Field Intelligence Architecture},
  url = {https://github.com/catojohansen/darkbot},
  version = {1.0.0},
  year = {2025},
}
```

## License

DARKBOT™ is available under a dual license. See LICENSE.md for details.

## Contact

For inquiries about licensing, partnerships, or technical questions:

- **Email**: katomckenna157@gmail.com
- **Website**: medium.com/@katomckenna157

---