

Quantum-Inspired Computational Frameworks for Autonomous Trading Agents: A Deep Analysis of Local AI Optimization on the Polygon Network

The convergence of decentralized finance (DeFi), high-frequency trading (HFT), and local artificial intelligence execution marks a significant evolution in the architecture of automated market participation. As liquid assets increasingly migrate to Ethereum-compatible scaling solutions such as the Polygon network, the demand for sub-millisecond decision-making and predictive precision has created a technological bottleneck. Traditional machine learning models, while effective in static environments, often fail to meet the "winner-take-all" latency requirements of the blockchain's "Dark Forest," where Maximal Extractable Value (MEV) extraction is the primary driver of profitability.¹ Consequently, the industry is shifting toward quantum-inspired classical algorithms that leverage the probabilistic and parallel processing advantages of quantum mechanics without the hardware constraints of current noisy intermediate-scale quantum (NISQ) devices.³ By utilizing the Ollama framework for the local deployment and tuning of large language models (LLMs), developers can create "mock" quantum environments that facilitate complex, non-linear reasoning. This report provides an exhaustive technical analysis of how these quantum-inspired paradigms—specifically the Corner Classification (CC4) neural network, Toshiba's Simulated Bifurcation Machine (SBM), and variable-time preselection algorithms—can be integrated into a local trading stack to simulate faster code execution and achieve alpha on the Polygon network.

The Paradigm of Local Intelligence: Ollama as a Secure Execution Environment

The choice of execution environment for a trading agent is as critical as the strategy itself. In the context of high-stakes cryptocurrency trading, the use of cloud-based AI providers introduces unacceptable latencies and significant privacy risks, as trade intent can be leaked to service providers or third-party observers. Ollama has emerged as the premier open-source platform for running large language models locally on macOS, Linux, and Windows, providing a privacy-first approach that ensures the trading agent's logic remains entirely within the user's infrastructure.⁵ This local deployment is not merely a security measure; it is a prerequisite for the low-latency response times required to interact with Polygon's block-producing cycle.

Tuning LLMs for Probabilistic "Quantum" Logic

To simulate quantum algorithms within an Ollama-managed model, developers must move beyond standard zero-shot prompting and toward a deep tuning of the Modelfile. The Modelfile serves as the configuration blueprint for the local model, allowing for the precise adjustment of parameters that govern the model's stochastic behavior—a critical component when attempting to mimic the probabilistic nature of quantum superposition.⁷ By adjusting the temperature and top_p parameters, a developer can effectively control the "coherence" of the model's output. A higher temperature (e.g., 1.5) encourages a wider distribution of potential "tokens" or trade outcomes, simulating a state of superposition where multiple conflicting market interpretations exist simultaneously.⁷ Conversely, lower temperatures simulate the collapse of this wave function into a singular, deterministic trading decision.

Parameter	Standard Value	Quantum-Inspired Value	Functional Implication
temperature	0.8	1.1 - 1.5	Increases the probability of exploring non-obvious, non-linear trade routes. ⁷
top_p	0.9	0.5 - 0.7	Focuses the "collapsed" output on high-probability, high-conviction signals. ⁷
num_ctx	4096	8192 - 32768	Enables the processing of longer transaction histories for better trend coherence. ⁷
repeat_penalty	1.1	1.5	Prevents the model from getting stuck in local minima or repetitive reasoning cycles. ⁷

The system message (set via the SYSTEM instruction in the Modelfile) is used to define the "mock quantum" persona. By instructing the model to treat input market data as a series of qubits subjected to specific logic gates—such as the Hadamard (H) gate for creating

superposition or the Controlled-NOT (CNOT) gate for representing the entanglement of correlated asset pairs—the agent can be steered toward a more holistic, multi-dimensional analysis of the market state.⁹ Research indicates that LLMs, particularly reasoning-focused models like DeepSeek-R1, are capable of learning the evolution patterns of quantum bits and predicting the outputs of complex quantum dynamics with minimal error.⁹

Strategic Advantage of Locally Ran Models

The speed of execution for a local model is heavily dependent on the hardware configuration and the quantization level of the model itself. For real-time trading on the Polygon network, where block times are approximately two seconds, a balance must be struck between the model's parameter count and its response time.¹³ While a full-parameter model like Llama 3.1 405B offers superior reasoning, it may be too slow for high-frequency applications. Instead, distilled or quantized versions (e.g., DeepSeek-R1:8b or Llama 3.2:3b) are preferred for their ability to provide sub-second inference.⁶

Model Variant	Parameters	Latency (Local GPU)	Trading Context
DeepSeek-R1:1.5b	1.5 Billion	~0.5s	Micro-arbitrage and signal filtering. ¹⁵
DeepSeek-R1:7b	7 Billion	~1.2s	Mid-frequency trend and sentiment analysis. ⁸
Llama 3.2:1b	1 Billion	~0.3s	High-speed data preprocessing and gate simulation. ⁶
Qwen 2.5:32b	32 Billion	~4.0s	Complex strategy backtesting and optimization. ¹⁶

By deploying these models locally, the agent can maintain a persistent "thinking" process, where it continuously updates its internal "quantum" representation of the liquidity pools on Polygon as new blocks arrive, rather than starting a fresh inference for every trade.¹⁷

Simulating Faster Code Execution: The Algorithmic

Speedup

The user's query highlights a critical desire to use quantum algorithms to simulate "faster code execution." In the context of classical computing, this does not typically mean the physical cycles of the CPU are faster; rather, it refers to a reduction in the algorithmic complexity (O -notation) of the tasks performed. Quantum-inspired algorithms can bypass the traditional bottlenecks of linear processing by utilizing parallel search techniques and non-iterative learning structures that achieve results in a fraction of the time required by classical methods.⁴

The Kak Family of Neural Networks: Instantaneous Training

One of the most potent examples of simulating faster execution is the use of the Kak family of neural networks, specifically the Corner Classification (CC4) architecture. Proposed by Subhash Kak, the CC4 network is designed for "instantaneous" training.²⁰ Unlike standard artificial neural networks (ANNs) that rely on backpropagation—a process involving thousands of weight adjustment iterations—CC4 utilizes prescriptive learning, where the interconnection weights are assigned directly based on the input patterns themselves.²⁰

This prescriptive approach is highly advantageous for high-frequency cryptocurrency trading, where market regimes shift rapidly and a model that takes minutes to retrain would be obsolete before its first trade. In a CC4 network, the hidden layer neurons act as "corners" of a multidimensional cube, representing specific market states.²⁰ When a new trading pattern is identified, a new neuron is simply added or its weights are instantly set, allowing the model to adapt to new volatility without the latency of an iterative training cycle.²²

Mathematical Implementation of the CC4 Learning Rule

The weight between an input neuron x_i and a hidden neuron j is determined by the input bit x_i of the training sample j :

$$w_{ij} = \begin{cases} 1 & \text{if } x_i = 1 \\ -1 & \text{if } x_i = 0 \end{cases}$$
 The bias weight $w_{n+1, j}$ for the hidden neuron j is calculated as:

$$w_{n+1, j} = r - s + 1$$

where s is the total number of 1s in the training sample j and r is the "radius of generalization".²³ This r value determines the Hamming distance within which a new, unlearned input will still be classified as belonging to the same state as the training sample.²⁰ For a trading agent, this means that even if a price point is slightly different from a historically observed profitable signal, the network can instantly generalize and recognize it as a buying opportunity, effectively simulating a "fuzzy" or quantum-like superposition of nearby price states.

Unary Coding: Bio-Inspired Data Representation

The efficiency of the CC4 network is further enhanced by the use of unary and spread-unary coding for input data. Standard binary coding is inefficient for neural network training because small changes in value (e.g., from 7 to 8) can lead to large, non-linear changes in the bit string (from 0111 to 1000), increasing the Hamming distance and confusing the learner.²³ Unary coding represents a number n by a sequence of n ones followed by a zero, ensuring that the Hamming distance between any two numbers is directly proportional to their numerical difference.²¹

A more advanced version, known as "spread-unary" coding, allows a number to be represented by a fixed-length string with a specific "spread" of ones.²⁶ This method mimics biological neural systems and provides a "saturation" of Hamming distance, which improves the network's resilience to noisy market data—a common feature of high-frequency cryptocurrency feeds.²¹

Coding Method	Efficiency (n bits)	Hamming Distance Fidelity	Use Case
Binary	2^n numbers	Low (non-linear)	Standard storage. ²⁶
Standard Unary	n numbers	Perfect (linear)	High-precision signals. ²⁶
Spread-Unary	$\approx n^2$ numbers	Balanced	Robust trend prediction. ²⁷
Fixed-Length Unary	n numbers	Perfect	Instantaneous learning. ²³

By representing price levels, volume spikes, and gas fees in spread-unary format, the CC4-based trading agent can process market movements with a structural speed that exceeds traditional floating-point matrix calculations. This is essentially a "mock" form of quantum parallelism, where the data itself is encoded in a way that allows the "hidden" layer of the network to evaluate all stored patterns simultaneously.²⁸

Quantum Algorithms for Statistical Arbitrage on Polygon

Statistical arbitrage—the identification and exploitation of mean-reverting relationships between correlated assets—is the bread and butter of HFT firms. However, identifying these relationships in a universe of thousands of tokens on the Polygon network is a computationally

expensive task, often requiring $O(N^2d)$ complexity classically, where N is the historical data length and d is the number of stocks.¹⁹ Quantum-inspired algorithms can reduce this complexity significantly, enabling "preselection" and "verification" of trade pairs in real-time.

Variable Time Preselection Algorithm (VTPA)

The VTPA is a quantum-inspired subroutine designed to identify portfolios that exhibit multicollinearity, a necessary condition for cointegration.¹⁹ In a high-frequency environment, the agent cannot afford to perform full cointegration tests on every possible asset pair. Instead, the VTPA utilizes a "Variable Time Condition Number Estimation" technique to find portfolios with large condition numbers and small eigenvalues.¹⁹

This process relies on the Quantum Condition Number Comparison Algorithm (QCNCA), which repeatedly applies a phase estimation logic to probe the matrix of price correlations.³⁰ The average query complexity is reduced to:

$O(\sqrt{d} \kappa^2 (\log(1/\epsilon))^2)$

where κ is the condition number and ϵ is the precision.¹⁹ By focusing only on the "collinear" candidates identified by the VTPA, the trading agent can ignore the vast majority of market noise, effectively "zooming in" on profitable opportunities with a speed that simulates quantum-enhanced search.³⁰

Quantum Cointegration Test (QCTA)

Once a candidate pair (e.g., MATIC and a liquid-staking derivative like stMATIC) is preselected, the QCTA performs a rigorous verification.¹⁹ The algorithm implements the Engle-Granger two-step method, using a quantum linear regression model to estimate the relationship between the assets and then performing an Augmented Dickey-Fuller (ADF) test on the residuals to confirm stationarity.¹⁹

Phase	Classical Method	Quantum-Inspired Enhancement
Preselection	Matrix Factorization $O(N^3)$	VTPA with Variable Time $O(\sqrt{d}\kappa^2)$. ³⁰
Regression	Ordinary Least Squares (OLS)	Quantum Linear Regression (QLR). ¹⁹

Verification	ADF Test on CPU	ADF with qRAM-based data retrieval. ¹⁹
Complexity	10^{15} operations (for $N=10^7$)	10^8 operations (for $\kappa=1000$). ¹⁹

This reduction in operations translates directly into a reduction in "time-to-signal." On the Polygon network, where "Sandwich" attacks and "Front-running" occur within the span of a single block, the ability to confirm a cointegration and issue a trade order in microseconds is the difference between capturing a spread and being the victim of slippage.¹

Combinatorial Optimization: The Simulated Bifurcation Machine

A critical challenge for a Polygon trading agent is the identification of optimal multi-leg arbitrage paths across various decentralized exchanges (e.g., Uniswap V3, QuickSwap, and Balancer). This is a combinatorial optimization problem—specifically, finding the shortest path in a market graph where edge weights (exchange rates) are constantly fluctuating. Classical path-finding algorithms like Dijkstra's are often too slow when scaled to hundreds of tokens and thousands of liquidity pools.

Ising Models and the SBM

Toshiba's Simulated Bifurcation Machine (SBM) provides a quantum-inspired solution to these "hard" quadratic discrete optimization problems.³¹ The SBM operates by modeling the market as an "Ising machine," where each potential trade route is represented as a state in a many-body system of interacting oscillators.³³ The algorithm simulates the time-evolution of these oscillators, allowing the system to "bifurcate" or settle into a low-energy state that corresponds to the most profitable trading path.³³

The SBM has been demonstrated to solve these problems with microsecond latency, achieving a system-wide speed that is up to 14 times faster than traditional Simulated Annealing (SA).³³ For a Polygon agent, an SBM-inspired solver can ingest a market feed packet informing a change in the best bid/ask prices and issue an order packet within 30 microseconds.³²

Advantages of the SBM in Real-Time Trading

The SBM is particularly well-suited for high-speed real-time trading because it can be implemented on commodity hardware like GPUs or specialized hardware like FPGAs.³²

1. **Parallelism:** The algorithm is inherently parallel, allowing it to evaluate thousands of transaction paths simultaneously.³⁴

2. **Reliability:** In long-term experiments, SBM-based systems have shown a 90.96% probability of finding the absolute top optimal solution without making calculation errors.³¹
3. **Scale-Out Capability:** Multiple FPGAs can be networked to exchange information on variables, enabling the system to handle larger stock universes or complex "basket" trades without increasing latency.³⁵

Metric	Simulated Bifurcation Machine (SBM)	Standard Optimization Solver
Execution Speed	< 30 microseconds	> 1 millisecond. ³²
Accuracy	90%+ Top-1 Solution	Variable based on iterations. ³²
Platform	FPGA / GPU (Massively Parallel)	CPU / GPU (Limited Parallelism). ³⁴
Optimization Type	Quadratic Discrete Optimization	Linear or Simple Non-Linear. ³¹

For a trading agent on Polygon, the "Extended Pair-Trade" and "Correlation-Diversified Portfolio" strategies enabled by SBM-like solvers allow for the detection of mispricings that exist only in the complex relationships between multiple tokens—opportunities that are "ever-untargeted" by traditional bots.³¹

Navigating the Polygon "Dark Forest": MEV and Execution Strategy

Unlike the Hedera network, which utilizes a "Fair Ordering" consensus mechanism to prevent reordering and sandwich attacks, the Polygon network (specifically the PoS chain) operates with a public mempool.¹ This environment is a "Dark Forest" where every pending transaction is visible to predatory searcher bots and validators who can prioritize their own trades by manipulating block order.¹

MEV Taxonomy on Polygon

The extraction of MEV on Polygon has matured significantly, shifting from simple arbitrage to structured, high-competition strategies. Research into Account Abstraction (AA) transactions on Polygon reveals that approximately 13% of MEV is captured via "FastLane" strategies,

which involve competitive bidding and bundling of transactions.²

MEV Strategy Type	Mechanism	Impact on Trading Agent
Sandwich Attack	Placing trades before and after a victim trade. ²	Requires slippage protection and private RPCs. ³⁷
Front-Running	Detecting a trade and jumping the queue. ¹	Solved by local signal generation and fast execution. ³⁷
Arbitrage (FastLane)	Competitive bidding for liquidity imbalances. ²	Requires quantum-inspired speed to win the bid. ³¹
Spam-based MEV	Flooding the network with arbitrage attempts. ²	Inefficient; neutralized by fixed gas fee models. ³⁷

To succeed in this environment, the agent must not only be fast in its reasoning but also in its interaction with the network. This necessitates the use of dedicated trading nodes and high-performance gRPC streams.¹⁴

The Role of gRPC Streams and Low-Latency RPCs

For high-frequency trading, standard JSON-RPC endpoints are a major source of latency due to the overhead of HTTP headers and the polling nature of the requests. Modern HFT stacks on Polygon and similar chains utilize gRPC-based "Geyser" or "Shred" streams.³⁸ Specialized providers like Triton One and RPC Fast deliver real-time streams of block, slot, and account data directly to the agent's node.³⁸

By bypassing the standard "gossip" protocol—where shreds of block data hop from validator to validator—these specialized feeds can deliver market updates in under 50 milliseconds.³⁹ Stacking these feeds with a "mock quantum" solver creates a formidable pipeline:

1. **Data Ingestion:** gRPC streams deliver the latest mempool state and liquidity pool changes in <50ms.³⁹
2. **Signal Generation:** The CC4/Unary network identifies a pattern "instantaneously" (microseconds).²⁰
3. **Optimization:** The SBM-inspired Ising machine finds the optimal trade route in <30 microseconds.³²
4. **Execution:** The agent submits a transaction bundle via a private relay (e.g., Flashbots or FastLane) to prevent front-running.²

Architectural Comparison: Polygon vs. Hedera

The original research materials provided for this task include a detailed guide on deploying autonomous agents on the Hedera network by Chad Brewer (Dec 2025).³⁷ While the user's primary focus is the Polygon network, the architectural distinctions between the two are instructive for understanding where a quantum-inspired speed advantage is most effective.

Fair Ordering vs. Gas Auctions

Hedera’s "Fair Ordering" mechanism, based on asynchronous Byzantine Fault Tolerance (aBFT), assigns consensus timestamps to transactions based on the median time they were received by the entire network.³⁷ This effectively eliminates MEV because no single leader can unilaterally reorder transactions or see a mempool before it is finalized.³⁷

On Hedera, "alpha" is generated purely through network latency—being topologically close to the consensus nodes—and the deterministic calculation of fixed USD fees (\$0.001).³⁷ On Polygon, however, fees are variable, and the ordering is determined by gas-price auctions.¹ This means that a Polygon agent's "mock quantum" logic must account for a much wider range of variables, including the "priority fee" required to jump the queue and the risk of being sandwiched by a validator.²

Metric	Hedera (Hashgraph)	Polygon (EVM Sidechain)
Finality Time	3-5 Seconds (Deterministic)	~2 Seconds (Probabilistic). ¹³
TPS	10,000+ (HTS) / ~400 (HSCS)	~700 Theoretical / ~100 Real-time. ¹³
Fee Model	Fixed USD (Paid in HBAR)	Variable Gas (MATIC/POL). ¹³
Consensus	Gossip-about-Gossip (aBFT)	Proof of Stake (PoS / Peering). ¹³
MEV Risk	Near-Zero	High (1% of TVL extracted). ²

For a trading agent, the Polygon environment is more "adversarial," making the speed gains from quantum-inspired algorithms even more valuable. In a network where you can "pay to play" through gas auctions, the first bot to identify a price gap with high confidence and

submit a precisely-priced bundle will win the opportunity.²

Implementing the Mock Quantum Agent with Ollama and Python

Integrating these disparate technologies into a cohesive agent requires a robust software framework. The following architecture demonstrates how a Python-based agent can orchestrate Ollama, a local Ising solver, and a gRPC data stream.

Agent Logic and Decision Batching: The KITE Protocol

The KITE protocol offers a unique framework for "Quantum-Inspired Decision Batching".⁴² In traditional DeFi, early movers create information asymmetries that disadvantage later participants. KITE maintains participant autonomy by keeping individual agent decisions private and independent (in a "superposition" state) until a coordination threshold is met.⁴² The "wave function" of collective decisions then collapses into a synchronized revelation phase, preventing cascade failures and ensuring genuine price discovery.⁴²

In a multi-agent system on Polygon, this could be used to coordinate a fleet of "searcher" agents that explore different liquidity segments:

- **Segment A Searcher:** Monitoring stablecoin pairs.
- **Segment B Searcher:** Monitoring volatile "long-tail" assets.
- **Coordination Agent:** Using a local Ollama model to synthesize these findings and execute a multi-segment cross-asset swap via the SBM-inspired pathfinder.

Technical Implementation Workflow

The implementation of such an agent involves several distinct layers of technology working in tandem to minimize the "perception-to-execution" loop.

1. **Local Model Setup:** Install Ollama and pull a reasoning model such as DeepSeek-R1.¹⁵ Customize the Modelfile to prioritize quantum-inspired reasoning and low-latency response.⁷
2. **Data Pipeline:** Connect to a high-speed gRPC provider (e.g., QuickNode) to ingest Polygon mempool and log events.³⁸
3. **Preprocessing:** Convert raw market data (price, volume, liquidity) into spread-unary coding strings to feed the CC4 network.²³
4. **Instantaneous Learning (CC4):** The local agent uses a CC4 network to classify the current market state as "Bullish," "Bearish," or "Mean-Reverting" based on the radius of generalization $\$r\$$.²⁰
5. **Optimization (SBM):** If a profitable state is detected, the Ising-based solver identifies the optimal sequence of swaps across the Polygon ecosystem in <30 microseconds.³²
6. **Bidding Logic:** A secondary reasoning loop in Ollama calculates the optimal "Priority

Fee" based on current mempool congestion to ensure first-block inclusion.²

7. **Execution:** The transaction is dispatched via a Flashbots bundle or a direct validator connection to minimize the risk of being front-run.²

The Future of Quantum-Inspired DeFi Agents

The transition from static algorithmic scripts to dynamic, quantum-inspired autonomous agents represents a fundamental shift in market microstructure. While true quantum hardware remains a future prospect, the "mock" algorithms described here provide a bridge that allows classical hardware to operate with quantum-like efficiency.³

Strategic Implications for Capital Accumulation

The use of these advanced technologies allows for a "meritocratic competition" based on architectural efficiency.³⁷ On the Polygon network, this translates to the ability to execute "Dust Arbitrage"—capturing tiny price discrepancies that are unprofitable for standard bots due to their slower calculation speeds or higher gas overhead.³⁷

Furthermore, by aligning the agent's function with ecosystem goals—such as maintaining liquidity depth or ensuring price efficiency—developers can leverage grants and incentive programs from foundations (e.g., the Polygon Foundation or HBAR Foundation) to secure sustainable operational capital.³⁷ This transforms the agent from a predatory bot into a "value-accretive pillar" of the DeFi economy.³⁷

Conclusion of Technical Analysis

The synthesis of local AI execution via Ollama and quantum-inspired classical algorithms like the CC4 network and SBM solvers creates a powerful framework for high-frequency trading on the Polygon network. By reducing the algorithmic complexity of market analysis and path optimization, these methods effectively simulate "faster code execution," allowing agents to perceive and react to market-moving events before they are finalized on the ledger.

The successful implementation of such an agent depends on a rigid adherence to data latency minimization, the use of privacy-focused local models to protect trade intent, and a robust understanding of the adversarial MEV landscape on the Polygon blockchain. As the DeFi ecosystem continues to evolve, the integration of these quantum-inspired paradigms will be the defining factor in achieving sustainable alpha in an increasingly automated and competitive market.

Works cited

1. Solving MEV with Zero-Knowledge Proofs and Trusted Execution Environments - Medium, accessed December 29, 2025, <https://medium.com/@dheeraj.eth/solving-mev-with-zero-knowledge-proofs-an>

- [d-trusted-execution-environments-d5d79302ca42](#)
2. Unpacking Maximum Extractable Value on Polygon: A Study on Atomic Arbitrage - arXiv, accessed December 29, 2025, <https://arxiv.org/html/2508.21473v1>
 3. Quantum-Inspired Algorithms and Perspectives for Optimization, accessed December 29, 2025, <https://www.mdpi.com/2079-9292/14/14/2839>
 4. Benchmarking of GPU-optimized Quantum-Inspired Evolutionary Optimization Algorithm using Functional Analysis - arXiv, accessed December 29, 2025, <https://arxiv.org/html/2412.08992v1>
 5. Ollama - AI Models, accessed December 29, 2025, <https://ollama.org/>
 6. ollama/ollama: Get up and running with OpenAI gpt-oss, DeepSeek-R1, Gemma 3 and other models. - GitHub, accessed December 29, 2025, <https://github.com/ollama/ollama>
 7. Modelfile Reference - Ollama's documentation, accessed December 29, 2025, <https://docs.ollama.com/modelfile>
 8. Running DeepSeek-R1 with Ollama: A Complete Guide - Collabnix, accessed December 29, 2025, <https://collabnix.com/running-deepseek-r1-with-ollama-a-complete-guide/>
 9. (PDF) Application of Large Language Models to Quantum State Simulation - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/384770892_Application_of_Large_Language_Models_to_Quantum_State_Simulation
 10. How to prompt Code Llama · Ollama Blog, accessed December 29, 2025, <https://ollama.com/blog/how-to-prompt-code-llama>
 11. artificial intelligence: foundations, applications and future directions - Zenodo, accessed December 29, 2025, https://zenodo.org/records/15091725/files/Artificial_Intel.pdf?download=1
 12. [2410.06629] Application of Large Language Models to Quantum State Simulation - arXiv, accessed December 29, 2025, <https://arxiv.org/abs/2410.06629>
 13. Hedera vs Polygon [TPS, Max TPS, Block Time] - Chainspect, accessed December 29, 2025, <https://chainspect.app/compare/hedera-vs-polygon>
 14. HFT Solana Trading Nodes - Dysnix, accessed December 29, 2025, <https://dysnix.com/hft-solana-trading-node>
 15. How to Set Up and Run DeepSeek-R1 Locally With Ollama - DataCamp, accessed December 29, 2025, <https://www.datacamp.com/tutorial/deepseek-r1-ollama>
 16. ivanfioravanti/prompt-eng-ollama-interactive-tutorial - GitHub, accessed December 29, 2025, <https://github.com/ivanfioravanti/prompt-eng-ollama-interactive-tutorial>
 17. Thinking - Ollama's documentation, accessed December 29, 2025, <https://docs.ollama.com/capabilities/thinking>
 18. Thinking Locally, Acting Privately: Building a Reasoning-Powered Q&A App with Deepseek R1 using Ollama, Streamlit and RAG | by Fatih KIR | Medium, accessed December 29, 2025, <https://medium.com/@fatikir15/thinking-locally-acting-privately-building-a-reasoning-powered-q-a-app-with-deepseek-r1-using-3d487627251b>
 19. (PDF) Quantum Quantitative Trading: High-Frequency Statistical ..., accessed

December 29, 2025,

https://www.researchgate.net/publication/351221880_Quantum_Quantitative_Trading_High-Frequency_Statistical_Arbitrage_Algorithm

20. The Basic Kak Neural Network with Complex Inputs 1 Introduction - arXiv, accessed December 29, 2025, <https://arxiv.org/pdf/cs/0603015>
21. A Class of Instantaneously Trained Neural Networks | Request PDF - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/222574077_A_Class_of_Instantaneously_Trained_Neural_Networks
22. Training of CC4 Neural Network with Spread Unary Coding - Semantic Scholar, accessed December 29, 2025, <https://www.semanticscholar.org/paper/Training-of-CC4-Neural-Network-with-Spread-Unary-Potluri/c2f536a43170a5f5141c6aa73d7808a41adcf9ff>
23. Unary Coding for Neural Network Learning - arXiv, accessed December 29, 2025, <https://arxiv.org/pdf/1009.4495>
24. Online Hybrid Neural Network for Stock Price Prediction: A Case Study of High-Frequency Stock Trading in the Chinese Market - MDPI, accessed December 29, 2025, <https://www.mdpi.com/2225-1146/11/2/13>
25. [1712.09331] Learning Based on CC1 and CC4 Neural Networks - arXiv, accessed December 29, 2025, <https://arxiv.org/abs/1712.09331>
26. Training of CC4 Neural Network with Spread Unary Coding - arXiv, accessed December 29, 2025, <https://arxiv.org/pdf/1509.01126>
27. Generalized Unary Coding | Request PDF - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/282831250_Generalized_Unary_Coding
28. Quantum neural network - Wikipedia, accessed December 29, 2025, https://en.wikipedia.org/wiki/Quantum_neural_network
29. Quantum-like behavior without quantum physics II. A quantum-like model of neural network dynamics - PubMed Central, accessed December 29, 2025, <https://pmc.ncbi.nlm.nih.gov/articles/PMC6208592/>
30. Quantum Quantitative Trading: High-Frequency Statistical Arbitrage ..., accessed December 29, 2025, <https://arxiv.org/abs/2104.14214>
31. The world's first demonstration of systems that execute unprecedented stock trading strategies based on computationally-hard quadratic discrete optimization by using quantum-inspired computer, Toshiba's simulated bifurcation machine, accessed December 29, 2025, <https://www.global.toshiba/ww/technology/corporate/rdc/rd/topics/23/2312-03.html>
32. Toshiba Develops Proof-of-concept Device for Ultra-high-speed Financial Transaction Machine with Simulated Bifurcation Algorithm -Detects most profitable transaction opportunities with probabilities exceeding 90% at microsecond speeds; starting open recruitment of financial engineering experts toward practical applications- | Corporate Laboratory (Komukai region), accessed December 29, 2025, <https://www.global.toshiba/ww/technology/corporate/rdc/rd/topics/19/1910-02.ht>

[ml](#)

33. Enhancing high-speed trading strategies with quantum-inspired ..., accessed December 29, 2025,
https://www.global.toshiba/content/dam/toshiba/ww/products-solutions/ai-iot/sbm/pdf/TOSHIBA-Enhancing_high-speed_trading_strategies_with_quantum-inspired_technology.pdf
34. Efficient and Scalable Architecture for Multiple-Chip Implementation of Simulated Bifurcation Machines - IEEE Xplore, accessed December 29, 2025,
<https://ieeexplore.ieee.org/iel7/6287639/10380310/10460551.pdf>
35. Toshiba develops scale-out technology to take FinTech to a new level - IBS Intelligence, accessed December 29, 2025,
<https://ibsintelligence.com/ibsi-news/toshiba-develops-scale-out-technology-to-take-fintech-to-a-new-level/>
36. Toshiba and Dharma Capital's Joint Experiment in Financial Markets to Verify the Effectiveness of a Quasi-Quantum Computer When Applied to High Frequency Trading, accessed December 29, 2025,
<https://asia.toshiba.com/press-release/english/toshiba-and-dharma-capitals-joint-experiment-in-financial-markets-to-verify-the-effectiveness-of-a-quasi-quantum-computer-when-applied-to-high-frequency-trading/>
37. Hedera AI Trading and Accumulation Strategies,
https://drive.google.com/open?id=109rvmJrbpRUMsjKs7GZUC14O_utMoFbo0VsPu4TygFA
38. Triton RPC by Triton - Quicknode, accessed December 29, 2025,
<https://www.quicknode.com/builders-guide/tools/triton-rpc-by-triton>
39. Low-latency Solana playbook for HFT traders - RPC Fast, accessed December 29, 2025, <https://rpcfast.com/blog/low-latency-solana-playbook-hft-traders>
40. How to improve Solana RPC latency with ShredStream | Chainstack Blog, accessed December 29, 2025,
<https://chainstack.com/how-to-improve-solana-rpc-latency-with-shredstream/>
41. Does Maximal Extractable Value (MEV) exist on Hedera?, accessed December 29, 2025,
<https://hedera.com/blog/does-maximal-extractable-value-mev-exist-on-hedera>
42. Jia Lilly's Profile | Binance Square, accessed December 29, 2025,
https://www.binance.com/en-NG/square/profile/jenni_aura
43. 10 Best Polygon RPC Providers for Web3 Developers in 2025 - Dwellir, accessed December 29, 2025,
<https://www.dwellir.com/blog/10-best-polygon-rpc-providers-2025>