Building an Oracle Network for Ethereum to Leverage Large Language Models

Madhukara S. Holla (Team 2)

1 Introduction

Blockchain technology has revolutionized decentralized applications by providing a trustless and immutable framework. Smart contracts, a cornerstone of this innovation, enable automated execution of agreements without intermediaries. However, smart contracts are inherently limited to deterministic on-chain data, making it challenging to incorporate complex off-chain data. Large Language Models (LLMs), powered by advances in artificial intelligence, offer nuanced, context-aware insights. These capabilities are crucial for evolving blockchain use cases, including decentralized finance (DeFi), supply chain management, and autonomous organizations. Despite their potential, integrating LLMs into smart contracts presents challenges due to their non-deterministic outputs, security concerns, and lack of verifiable interactions. This report discusses the design and implementation of a decentralized oracle network that addresses these issues, enabling Ethereum smart contracts to securely leverage LLMs.

2 Problem Definition

Smart contracts are deterministic by design to ensure consensus across blockchain nodes. This determinism limits their ability to interact with external, dynamic, and non-deterministic systems like LLMs. The key challenges include:

- **Achieving Consensus:** LLMs generate probabilistic outputs, making it difficult for blockchain nodes to agree on a single result.
- **Ensuring Verifiability:** Blockchain systems rely on transparent and immutable data. Integrating LLMs requires mechanisms to verify and log their outputs in a trustless environment.

These challenges underscore the need for a robust oracle network capable of securely bridging the gap between blockchain systems and LLMs.

3 Proposed Solution

The proposed solution involves a decentralized oracle network tailored to address the integration of LLMs with Ethereum smart contracts. The system is designed to ensure security, consensus, and cost-efficiency while maintaining verifiability.

3.1 Core Architecture

- **Decentralized Oracle Network:** A network of 7 nodes operates under a Practical Byzantine Fault Tolerance (PBFT) consensus mechanism to handle non-deterministic LLM outputs.
- **Smart Contract Interface:** Solidity-based contracts deployed on the Polygon ZKEVM testnet facilitate the submission of queries to the oracle network and retrieval of responses.
- **Decentralized Storage:** The InterPlanetary File System (IPFS) stores interaction logs, with corresponding hash references maintained on-chain for verifiability.
- **Response Validation:** Semantic similarity scoring algorithms, implemented using cosine similarity, assess the consistency of LLM responses across oracle nodes.

3.2 Implementation Details

- Each oracle node runs as an HTTP server in a Docker container.
- Nodes interact with the ChatGPT-4.0 model to process queries and generate responses.
- Semantic similarity is calculated using cosine similarity to compare responses among nodes.
- PBFT is used to ensure consensus across the oracle network, even with potential Byzantine faults.
- The leader node aggregates responses, selects the consensus response, and uploads it to IPFS.
- The IPFS CID is returned to the listener component.
- The listener component detects "request received" events from the smart contract, forwards the query to the oracle network, and logs the final CID back to the blockchain using the "submitResponse" function.
- Solidity smart contracts handle query management, response storage, and auditing features.
- Error handling mechanisms ensure continued operation during partial node failures.

4 Results

The implementation of the decentralized oracle network achieved the following:

- **Consensus Achievement:** The oracle network successfully achieved consensus with up to $f = \lfloor (n-1)/3 \rfloor$ Byzantine nodes, even with non-deterministic LLM outputs.
- **Efficient Storage:** Using IPFS for off-chain storage minimized on-chain data footprint, reducing gas costs while maintaining verifiability through hash references.

- **Verifiable Logs:** All LLM interactions were logged in an immutable and verifiable manner, ensuring transparency and auditability.
- **Performance Metrics:** The system demonstrated a sub-30-second response time for LLM queries and effectively handled multiple concurrent requests using its Dockerized architecture.
- **Blockchain Integration:** The deployed smart contracts on the Polygon ZKEVM testnet reliably logged IPFS CIDs, enabling end-to-end traceability of oracle responses.

5 Limitations

The current implementation has the following limitations:

- **No Recovery Mechanism for PBFT Nodes:** The system does not implement recovery protocols for failed or disconnected nodes within the PBFT network.
- **No Peer Discovery:** Nodes must be manually configured as the system lacks automated peer discovery mechanisms.
- **No Staking by the Nodes:** Nodes do not stake any assets, resulting in a lack of economic incentives or "skin in the game" for honest participation.

6 Conclusion

This project successfully addresses the challenges of integrating non-deterministic LLM outputs into Ethereum smart contracts. By implementing a decentralized oracle network with robust PBFT consensus, cost-efficient storage using IPFS, and verifiable logging on the Polygon ZKEVM testnet, the system enables blockchain applications to harness the potential of AI-driven insights. Future work includes extending this framework to multiple blockchain platforms, developing novel consensus protocols, and conducting comprehensive performance and security analyses to further enhance its applicability and robustness.