OpenLedger - AI Blockchain

OpenLedger

July 1, 2025

Abstract

OpenLedger is the AI Blockchain, unlocking liquidity to monetize data, models and agents. The Blockchain records every contribution to the AI lifecycle on-chain, ensuring transparency, traceability, and accountability. Proof of Attribution ensures that ownership, credit, and economic rewards are accurately assigned to data providers, model developers, and other participants. OpenLedger creates a collaborative, traceable, and auditable environment for building AI that is open, fair, and owned by all. This paper outlines the design of the AI Blockchain, including its attribution mechanism, incentive structure, and the role of community collaboration in powering a decentralized AI economy.

1 Introduction

Artificial intelligence is rapidly evolving, but the infrastructure behind it has not. While large general-purpose models have driven early progress, the future of AI depends on models that are specialized, explainable, and aligned with real-world needs.

To enable specialized models, we need specialized data. But without proper attribution, transparency, and provenance, sourcing and sustaining that data becomes difficult. Contributions are hard to trace, incentives are misaligned, and the value created by data providers and model developers often goes unrecognized. There is no reliable way to prove who contributed what or to ensure that participants are rewarded in proportion to their impact.

OpenLedger introduces the AI Blockchain to solve these problems. It brings data contribution, model refinement, and collaboration on-chain, enabling transparent attribution and economic alignment throughout the AI lifecycle. With Proof of Attribution embedded into the blockchain, OpenLedger makes it possible to build AI in a way that is open, verifiable, and collectively owned.

1.1 Why AI Needs a Blockchain

AI is built on contributions from a wide range of participants, including data providers, model developers, researchers, and application builders. Today's AI infrastructure fails to recognize or reward these roles. Contributions are difficult to trace, attribution is often missing, and centralized platforms control access, credit, and monetization. A blockchain designed specifically for AI changes this by introducing three essential capabilities.

1.1.1 Collaboration and Ownership

Current AI models are primarily trained on large volumes of internet data. While this works for building general-purpose language models, it lacks the depth and domain specificity required for real-world applications. Specialized AI requires access to high-quality, curated datasets. However, today's AI stack does not provide a standard way to collect, attribute, or collaborate on this type of data.

OpenLedger addresses this gap by enabling open collaboration. Anyone can contribute data, models, or insights, and every contribution is permanently linked to its origin. This ensures that ownership is preserved and that contributors receive proper credit for their work.

1.1.2 Monetization and Rewarding Contributions

Many contributors in the AI pipeline, including those who provide data, evaluate models, or improve performance, go unrewarded despite the value they create. OpenLedger's Proof of Attribution ensures that contributors receive economic rewards based on the impact of their work. Anyone can participate in AI development and earn from their contributions without relying on centralized platforms.

1.1.3 Transparency and Traceability

Most AI systems operate as black boxes. Their internal decision logic, training data, and development history are hidden or unclear. This lack of explainability undermines trust and accountability. OpenLedger records every contribution on the blockchain with verifiable metadata. Data sources, model changes, and downstream interactions are fully traceable and auditable. This enables explainable AI across the full lifecycle.

While the need for a blockchain in AI development is clear, not all blockchains are designed to meet this need. Existing general-purpose chains lack the infrastructure to support attribution, coordination, and incentive alignment specific to AI. The next section outlines why these blockchains fall short and how OpenLedger fills the gap.

1.2 Why General-Purpose Blockchains Aren't Enough

Most existing blockchains were designed for financial transactions, asset transfers, or digital collectibles. While these systems have proven valuable for decentralized finance and ownership, they fall short when applied to the needs of artificial intelligence. AI development is not just about transactions or token movement. It requires collaboration, provenance, and ongoing contributions across data, models, and evaluation processes.

General-purpose blockchains lack native support for attribution, version control of models, structured data flows, and fine-grained reward systems. They cannot represent the full lifecycle of an AI system, from raw data to deployed model. Attempts to retrofit these blockchains for AI often result in workarounds that compromise scalability, traceability, and contributor incentives.

OpenLedger is built specifically to address these gaps. It is not a blockchain for general applications that happens to support AI. It is the AI Blockchain, designed from the ground up to handle attribution, model tracking, contribution history, and collaborative ownership.

OpenLedger does not adapt blockchain to AI. It redefines what a blockchain can be by aligning it directly with how AI is created, shared, and sustained.

With a clear understanding of what general-purpose blockchains cannot offer, it becomes important to revisit the current direction of the AI industry itself. The next section examines the limitations of general-purpose AI models and the growing importance of specialized, fine-tuned intelligence.

Feature	**General Blockchain**	**OpenLedger (AI Blockchain)**
Purpose	Built for DeFi, NFTs, etc.	Built only for AI and model workflows
Data Attribution	No native tracking	Every contribution is tracked on-chain
Rewards	For validators and miners	For data and model contributors
Security	Focus on transactions	Protects AI contributions and usage
Provenance	Limited	Full history of models and datasets
Governance	Votes on protocol upgrades	Votes on model quality and improvement rules

Table 1: Comparison of General Blockchains and OpenLedger

1.3 The Shift from General Models to Specialized AI

AI research is shifting from the pursuit of ever-larger, general-purpose models to the development of **highly optimized, domain-specific intelligence**. While foundational models are trained on broad internet data, they often lack applicability in specialized contexts. As a result, the industry now prioritizes **adaptability, efficiency, and application-specific intelligence**, which requires:

- **Fine-tuning models** for specialized applications in sectors like finance, healthcare, legal, and cybersecurity.
- **Reducing computational costs** by leveraging **smaller, optimized models** rather than running expensive, general-purpose LLMs.
- **Enhancing explainability** through **specialized models that provide interpretable, domain-specific justifications**.

The idea is not to replace foundational models, but to coexist and utilize the existing foundational models to make them even

more intelligent. Instead of competing with large-scale AI models, OpenLedger enables **fine-tuned, specialized AI models** to work in tandem with foundational AI, unlocking greater efficiency, accuracy, and real-world applicability.

To support this transition, OpenLedger provides a **framework for model attribution, decentralized fine-tuning, and governance**, ensuring that **AI builders and contributors receive fair recognition and financial incentives** for improving models.

The shift toward specialized AI models signals not just a technical change but a broader economic one. As AI systems become more autonomous and capable, they are redefining how value is created and exchanged in digital environments. The following section explores this economic transition and its implications.

1.4 Economic Shift from the Internet to AI: The Need for AI-Native Platforms

AI is not just a technological shift, it is an **economic transformation**. Traditional internet-based revenue models, such as **advertising, SEO, and centralized data monetization**, are being **disrupted by AI-driven automation**. This shift is causing fundamental changes in how digital economies function:

- **Search engines and SEO-based businesses are losing value** as AI-driven assistants **replace traditional search interactions**.
- **Content creation is increasingly AI-dominated**, reducing traditional monetization opportunities for human creators.
- **The legacy internet economy (advertising, centralized data ownership) is collapsing**, necessitating **a new system for AI-driven economic transactions**.

OpenLedger introduces **AI-native economic infrastructure**, ensuring that AI models and agents operate within a **sustainable, decentralized economy** where **contributors, developers, and liquidity providers are directly incentivized** through **tokenized AI models**.

A robust economic foundation requires clear roles and responsibilities. OpenLedger defines a set of key stakeholders who contribute to and benefit from the AI Blockchain. The next section outlines these roles and how they interact within the ecosystem.

1.5 Key Stakeholders in the OpenLedger Blockchain

The OpenLedger blockchain is built around a **collaborative model**, where multiple participants contribute to AI model creation, validation, and adoption:

- **AI Model Developers** – Build, train, and optimize AI models for deployment.
- **Data Contributors** – Provide domain-specific data with **verifiable attribution**, ensuring transparent model improvements.
- **Validators** – Secure the network, validate AI model performance, and prevent misuse or low-quality contributions.
- **Applications and AI Agents** – Consume AI models for real-world automation, integrating them into decentralized ecosystems.
- **Protocol Governors** – Stake OPEN tokens to earn voting power and guide the future of AI model development. They evaluate proposals, vote on their progression, and ensure that only high-quality models backed by the community advance through the lifecycle.

2 Architecture

The OpenLedger architecture[fig 1] is structured to provide an efficient, verifiable, and economically sustainable framework for decentralized specialized model development. It consists of two primary layers: the blockchain layer and the specialized model layer. Each of these layers plays a distinct role in ensuring that specialized models are secure, interpretable, and capable of interacting with external environments.

2.1 EVM Compatible Blockchain

The foundational layer of OpenLedger is built on an **EVM-compatible Blockchain**. This ensures low-latency transactions while leveraging the robust security and liquidity provided by ethereum. The blockchain layer plays

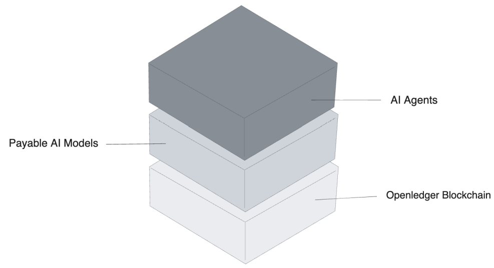

Figure 1: Openledger - AI Blockchain, unlocking liquidity to monetize data, models and agents

a crucial role in maintaining an immutable record of specialized models, ownership, incentives, data points, and proof of attribution. Transactions are optimized using rollups to ensure scalability while maintaining cryptographic integrity.

For a transaction set T, the state transition function σ is defined as:

$$\sigma(s, T) \rightarrow s', \tag{1}$$

where s and s' represent the system state before and after processing T, ensuring verifiable and deterministic execution of smart contract logic for model registration, staking, and governance.

2.2 Proof of Attribution

Ensuring transparent and verifiable **data attribution** is fundamental to Openledger. By leveraging **Proof of attribution techniques**, we enable model creators and data contributors to receive fair compensation while maintaining an auditable record of how data influences specialized models. This section details the full pipeline of data attribution, mathematical formulations, and mechanisms to ensure trust, accountability, and incentives within the OpenLedger ecosystem.

2.2.1 Why Data Attribution is Necessary

As AI models evolve, ensuring proper attribution of **data contributions** is essential for:

- Tracking how individual training data points influence model behavior.

- Rewarding data contributors based on their contribution's impact.
- Reducing bias and misinformation by penalizing low-quality data.
- Enabling verifiable proof-of-attribution for AI-generated outputs.

In Openledger, **proof of attribution** ensures that each data source is **cryptographically linked** to model outputs, providing an **immutable, decentralized ledger of contribution**.

2.2.2 Mathematical Framework for Influence-Based Attribution

To measure the influence of a given data point d_i on the model's output y, we define the **influence function**:

$$I(d_i, y) = \frac{\partial L(y, \theta)}{\partial \theta} \cdot \frac{\partial \theta}{\partial d_i} \tag{2}$$

where:

- $L(y, \theta)$ is the loss function for generating output y.
- θ represents the model parameters.
- $\frac{\partial L}{\partial \theta}$ measures how model parameters affect the loss.
- $\frac{\partial \theta}{\partial d_i}$ measures how training data influences model parameters.

2.2.3 Efficient Influence Computation

Computing full Hessian-based attributions is computationally expensive. Instead, Openledger adopts [1], an efficient approximation for **real-time attribution**:

$$I_{\text{DataInf}}(d_i, y) = \sum_{l=1}^{L} \frac{1}{\lambda_l} \left(\frac{1}{n} \sum_{j=1}^{n} \frac{L_{l,ij}}{\lambda_l + L_{l,ii}} - L_{l,i} \right) \tag{3}$$

where:

- $L_{l,ij} = \nabla_{\theta_l} \ell_i^T \nabla_{\theta_l} \ell_j$ represents pairwise loss impact.
- λ_l is a stabilization parameter.

This allows efficient **batch computation of attribution scores** while ensuring **scalability** [1].

Here's the updated OpenLedger Data Attribution Pipeline section with the correct math, current influence model (inference-based), and fee-reward flow—all formatted to match your system and phrased cleanly:

2.2.4 OpenLedger Data Attribution Pipeline

Step 1: Data Contribution

- Data contributors submit structured, domain-specific datasets intended to power Datanets.
- Each data submission is uniquely **attributed**, enabling traceable and verifiable contributions.

Step 2: Influence Attribution During Inference

When a model serves an inference, we compute how much each data point d_i contributed to the final output y. We define an *inference-level influence score*:

$$I(d_i, y) = \alpha \cdot F(d_i, y)$$

Where:

- α is a constant weight,
- $F(d_i, y)$ measures the direct impact of data point d_i on the model's output y.

Only points with $I(d_i, y) > 0$ qualify for reward distribution.

Step 3: Inference Fee Calculation and Contributor Rewards

Each inference request incurs a fee to cover computation and platform costs:

$$\text{Fee}_{\text{inference}} = \left(\frac{T_{\text{in}}}{1000} \cdot R_{\text{in}}\right) + \left(\frac{T_{\text{out}}}{1000} \cdot R_{\text{out}}\right) + F_{\text{platform}}$$

Where:

- T_{in}: number of input tokens

- T_{out}: number of output tokens
- R_{in}: rate per 1000 input tokens (in OPN)
- R_{out}: rate per 1000 output tokens (in OPN)
- F_{platform}: flat platform fee (in OPN)

If the platform fee is percentage-based:

$$F_{\text{platform}} = \eta \cdot \left(\frac{T_{\text{in}}}{1000} \cdot R_{\text{in}} + \frac{T_{\text{out}}}{1000} \cdot R_{\text{out}} \right)$$

After retaining the platform fee:

$$F_{\text{net}} = \text{Fee}_{\text{inference}} - F_{\text{platform}}$$

This is split into:

$$F_{\text{net}} = F_{\text{model}} + F_{\text{stakers}} + F_{\text{contributors}}$$

With:

$$F_{\text{model}} = \beta \cdot F_{\text{net}}, \quad F_{\text{stakers}} = \gamma \cdot F_{\text{net}}, \quad F_{\text{contributors}} = \delta \cdot F_{\text{net}}$$

Each contributor c_i is rewarded based on their proportional influence:

$$w_i = \frac{I(d_i, y)}{\sum_{j=1}^{n} I(d_j, y)} \quad \Rightarrow \quad \text{Reward}_{c_i} = w_i \cdot F_{\text{contributors}}$$

Example
Say:

- $T_{\text{in}} = 800$, $R_{\text{in}} = 0.2$
- $T_{\text{out}} = 1200$, $R_{\text{out}} = 0.4$
- $F_{\text{platform}} = 0.5$ OPN

$$\text{Fee}_{\text{inference}} = \left(\frac{800}{1000} \cdot 0.2\right) + \left(\frac{1200}{1000} \cdot 0.4\right) + 0.5 = 0.16 + 0.48 + 0.5 = 1.14 \text{ OPN}$$

$$F_{\text{net}} = 1.14 - 0.5 = 0.64 \text{ OPN}$$

If split as:

$$F_{\text{model}} = 0.448, \quad F_{\text{stakers}} = 0.064, \quad F_{\text{contributors}} = 0.128$$

Then, for a contributor with $w_i = 0.25$:

$$\text{Reward}_{c_i} = 0.25 \cdot 0.128 = 0.032 \text{ OPN}$$

This system enables **provable, real-time attribution**, ensuring contributors are rewarded proportionally to their data's influence on model inference. It discourages low-quality inputs and promotes the curation of high-impact, verifiable datasets.

2.3 Openledger AI Studio

The Openledger AI Studio is responsible for hosting, training, and evaluating explainable, domain-specific specialized models. Unlike generic foundation models, specialized models in OpenLedger are optimized for niche domains while ensuring complete transparency in their attribution and training processes.

2.3.1 Datanets

Specialized models rely on high-quality, domain-specific data to function optimally. The **Datanets** acts as an on-chain data aggregation and attribution mechanism, ensuring that every contributor is appropriately rewarded. Data points D submitted to the reservoir are assigned credibility scores based on staking weights w_i:

$$C(D) = \sum_{i=1}^{n} w_i \cdot f(x_i, y_i), \tag{4}$$

where x_i and y_i represent input-label pairs, and $f(x_i, y_i)$ is a function determining data quality and reliability.

2.3.2 Supervised Fine-Tuning

Once a sufficient dataset has been aggregated, specialized models undergo **supervised fine-tuning**. The model parameters θ are optimized via **Empirical Risk Minimization (ERM)**:

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)), \tag{5}$$

where L represents the loss function, and f_θ denotes the specialized model function.

Once a sufficient dataset has been aggregated, specialized models undergo **supervised fine-tuning**. The model parameters θ are optimized via **Empirical Risk Minimization (ERM)**:

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)), \tag{6}$$

where L represents the loss function, and f_θ denotes the specialized model function.

To improve efficiency, Openledger employs **fine-tuning**, where the specialized data collected is turned into intelligence, which can inturn power up the more purposeful and powerful AI Agents.

2.3.3 Reinforcement Learning with Human Feedback(RLHF)

To enhance model reasoning and adaptability, Openledger integrates **reinforcement learning (RL)** with human feedback. The reward function $R(\theta)$ for model updates is defined as:

$$R(\theta) = \sum_{i=1}^{n} w_i \cdot (V(y_i, f_\theta(x_i)) - \alpha \cdot L(y_i, f_\theta(x_i))), \tag{7}$$

where $V(y_i, f_\theta(x_i))$ represents validator-assigned scores based on correctness and interpretability, and α is a regularization parameter to prevent overfitting.

Users who provide high-quality feedback are **rewarded with stake incentives**, while those who attempt to manipulate the system face stake slashing.

2.3.4 Model Evaluation and Deployment: OpenLoRA

OpenLoRA is a **multi-tenant LoRA model serving framework** that enables scalable, low-latency inference for specialized models in Openledger. It optimizes GPU utilization and inference latency through:

2.3.5 Multi-Tenant GPU Infrastructure for LoRA Model Serving

OpenLoRA allows multiple LoRA models to share a single pre-trained backbone model. This reduces GPU memory overhead and optimizes computational efficiency.

2.3.6 Segmented Gather Matrix-Vector Multiplication (SGMV) for LoRA Model Execution

SGMV enables batched execution of LoRA adapters while maintaining efficient memory access patterns. Given input tensor $\mathbf{x}$, LoRA transformations are computed as:

$$\mathbf{y} = \mathbf{x}W + \mathbf{x}AB, \tag{8}$$

where W is the pre-trained model weight, and A, B are LoRA-specific adapters.

2.3.7 On-Demand Model Loading and Fast Switching Between LoRA Models

OpenLoRA dynamically loads LoRA adapters, reducing cold-start times by keeping backbone models in GPU memory and swapping in only the adapter weights.

2.3.8 GPU Resource Scheduling and Load Balancing

Requests are dynamically assigned to GPUs based on batch size constraints and memory availability. A new request R_{new} is assigned to GPU k^* where:

$$k^* = \arg\max_k \{R_k \mid R_k < B_{\max}, \mathcal{M}_k > \mathcal{M}_{\text{required}}\} \tag{9}$$

This ensures optimal throughput and load balancing.

2.3.9 Request Migration and Memory Optimization for KvCache Management

KvCache storage is dynamically managed to prevent memory overflow, with request migration occurring when a GPU reaches capacity. Requests are reassigned while preserving previous inference states to maintain seamless execution.

OpenLedger will host models and integrate with leading agent frameworks, ensuring seamless availability while establishing an infrastructure to develop more **purposeful and specialized AI agents**.

3 Ecosystem Tools

3.1 Model Factory

ModelFactory is an advanced fine-tuning platform for Large Language Models (LLMs) within the OpenLedger ecosystem. Unlike traditional fine-tuning approaches that require extensive command-line expertise or API integrations, ModelFactory offers a purely GUI-based solution that integrates secure dataset access and automated fine-tuning workflows. By leveraging OpenLedger's decentralized dataset repository, it ensures that fine-tuning processes are verifiable and secure.

3.1.1 System Architecture

The architecture of ModelFactory is modular and designed for scalability. It consists of the following key components:

- **User Management Module:** Manages authentication and dataset access permissions.
- **Dataset Access Control:** Enforces permissioned access to OpenLedger's dataset repository.
- **Fine-Tuning Engine:** Supports advanced fine-tuning optimizations such as LoRA and QLoRA.
- **Chat Interface Module:** Enables real-time interaction with fine-tuned models.

- **RAG Attribution Module:** Integrates retrieval-augmented generation (RAG) techniques for source attribution.
- **Evaluation & Deployment Module:** Provides benchmarking, validation, and deployment functionalities.

3.1.2 Fine-Tuning Process

ModelFactory employs a structured workflow that begins with dataset request and approval, followed by model selection, training, evaluation, and deployment.

Given a dataset D consisting of input-output pairs (x_i, y_i), fine-tuning follows an optimization function:

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) \tag{10}$$

where L is the loss function and θ are the model parameters. The system supports both full-model fine-tuning and parameter-efficient approaches like LoRA, where only adapter weights A and B are updated:

$$\mathbf{y} = \mathbf{x}W + \mathbf{x}AB \tag{11}$$

3.1.3 Benchmarking and Performance Optimization

To evaluate fine-tuned models, ModelFactory provides automated benchmarking based on:

- Perplexity and loss reduction on test datasets.
- Rouge and BLEU scores for text-based applications.
- GPU memory efficiency analysis, particularly for QLoRA models.

3.2 OpenLoRA

Open LoRA is a highly efficient framework designed for serving thousands of fine-tuned LoRA (Low-Rank Adaptation) models on a single GPU. It optimizes resource utilization by enabling dynamic adapter loading, reducing memory overhead, and ensuring high throughput with low latency. Open

LoRA is particularly beneficial for applications that require rapid model switching and efficient inference without deploying separate instances for each fine-tuned model.

3.2.1 Key Features

- **Dynamic Adapter Loading:** Just-in-time (JIT) loading of LoRA adapters from OpenLedger
- **Efficient Memory Utilization:** Supports merging adapters per request for ensemble inference without preloading all models into memory.
- **Optimized Inference:** Uses advanced optimizations like tensor parallelism, flash-attention, paged attention, and quantization for improved efficiency.
- **Scalability:** Supports serving thousands of fine-tuned LoRA models on a single GPU.
- **Cost Reduction:** Reduces serving costs while maintaining low latency and high throughput.
- **Streaming & Quantization:** Implements token streaming and quantization for optimized inference .

4 Model Lifecycle in OpenLedger

The OpenLedger model lifecycle is a structured process that ensures AI models are collaboratively developed, economically sustained, and seamlessly integrated into AI-driven applications. Each phase plays a crucial role in building specialized, high-performance AI models while maintaining decentralized governance and financial incentives.

4.1 Model Proposal

Developers initiate the process by submitting a model proposal that outlines the model's purpose, architecture, and intended use case. To ensure commitment and prevent spam submissions, proposers may be required to stake a minimum amount of tokens.

4.2 Model Governance

Model proposals are reviewed and selected through a governance process led by Protocol Governors. Voting power is determined by the number of gOPEN tokens each participant holds. Proposals that reach the required support threshold advance to the next phase of development. This ensures that model progression reflects collective interest and aligns with the incentives of active participants.

4.3 Specialized Data Collection

High-quality, domain-specific data is essential for training specialized AI models. OpenLedger facilitates decentralized data collection, ensuring that contributors are rewarded based on the quality and relevance of their data. Data is attributed using cryptographic mechanisms, providing transparency and verifiability while preventing low-quality or adversarial submissions.

4.4 Model Fine-Tuning

After data collection, the model undergoes fine-tuning to enhance its performance for specific applications. This process ensures that the model is more efficient, accurate, and well-suited for real-world deployment.

4.5 Model Optimization and Alignment using RLHF

To further refine model behavior, reinforcement learning with human feedback (RLHF) is applied. Human validators provide feedback on model outputs, helping to align the model with desired ethical, logical, and functional standards. Contributors who provide valuable feedback are rewarded, while low-quality contributions may face penalties.

4.6 API and Integrations with Agent Frameworks

The final stage involves integrating the trained model into AI-driven applications and intelligent agent frameworks. OpenLedger provides APIs that allow developers to connect models and also provide integrations with agent frameworks. These integrations enable AI models to function as decision-making engines in decentralized applications, expanding their usability.

5 OPEN Token

The OpenLedger Token (OPEN) serves as the fundamental unit of value within the OpenLedger ecosystem. It drives the economic incentives, ensuring that all participants, from data contributors to model creators, are fairly rewarded.

5.1 Tokenomics

Category	**Allocation(in percentage)**
Community	51.71%
Investors	18.29%
Team	15%
Liquidity	5%
Ecosystem	10%

Table 2: Tokenomics of OPEN Token

5.2 Key Utilities of the Token

5.2.1 Proposal and Platform Fees

- Model Creators use tokens to propose new AI models.
- A platform fee is paid to the OpenLedger Treasury to ensure long-term sustainability.
- The Treasury allocates attribution rewards to contributors.

5.2.2 Data Contribution and Attribution Rewards

- Users, Subject Matter Experts (SMEs) and Enterprises contribute data for AI model training.
- Contributors earn attribution rewards in tokens based on their data's impact.

5.2.3 Model Creation and Public Hosting

- Once enough data is collected and the conditions for the bonding curve are reached, the AI model is created and optimized.
- Successfully deployed models generate revenue, which is shared among stakeholders.

5.2.4 Model Inference Payments

- Every time an AI model is used for inference, the computation is paid for using tokens.
- Fine-tuning occurs via supervised learning and reinforcement learning using human feedback.

6 Making a Self-Sustainable Decentralized AI Ecosystem

OpenLedger employs a unified growth flywheel model to drive the sustainable and scalable development of its AI and blockchain ecosystems. This integrated approach ensures that both ecosystems operate in synergy, reinforcing each other within a self-perpetuating loop.

6.1 AI Ecosystem Flywheel

The growth of the AI ecosystem in OpenLedger begins with **model creators** and **developers** who propose, collect specialized data, fine-tune, and deploy specialized AI models. These creators leverage OpenLedger's **Datanets**, gaining access to high-quality datasets and secure fine-tuning tools such as ModelFactory and OpenLoRA.

As these AI models evolve, they contribute to the creation of a **self-sustaining economy**, where **attribution rewards** incentivize further development and data contributions. Increased model usage ensures that **data contributors and model builders are fairly rewarded**, reinforcing the ecosystem's continuous expansion and innovation.

6.2 Blockchain Ecosystem Flywheel

The blockchain ecosystem in OpenLedger complements and amplifies the AI ecosystem. Built on a decentralized infrastructure, it provides the trust, transparency, and incentive mechanisms required to sustain the platform.

1. **Enabling Transparency and Verifiability** – OpenLedger transforms AI models from black-box systems into transparent, attributable, and verifiable entities, ensuring all contributions and model evolutions can be publicly audited.

2. **Governance for Model Advancement** – Governance ensure that only the most valuable and community-backed models progress, preventing unnecessary development and aligning economic incentives with real-world demand.

3. **Model Utilization and Transactions:** As models are deployed, users interact with them through the blockchain. These interactions generate transactions recorded immutably on the blockchain, ensuring accountability and data provenance.

4. **Increased Validator Engagement:** With higher transaction volumes and revenue, validators are incentivized to maintain the blockchain infrastructure. This improves network stability and scalability, further attracting developers to build on OpenLedger.

5. **Enhanced Ecosystem Trust:** The stable and decentralized blockchain fosters trust among participants, encouraging further development, innovation, and ecosystem expansion.

6.3 Synergy Between AI and Blockchain Ecosystems

The synergy between the AI and blockchain ecosystems forms the core of OpenLedger's growth flywheel. Developers serve as the linchpin, driving innovation in both ecosystems:

- **From Blockchain to AI:** Transparent community enabled model development process ensure that data and AI models are built on a foundation of trust and accountability. The decentralized nature of the blockchain enables a fair and secure platform for AI development.

- **From AI to Blockchain:** As AI models become more sophisticated and widely used, they generate increased revenue and transactions on the blockchain. This strengthens the infrastructure and incentivizes further validator participation, creating a virtuous cycle.
- **Mutual Reinforcement:** Revenue generated from blockchain transactions funds further AI innovation, while advancements in AI drive greater adoption and engagement with the blockchain. Together, these ecosystems create a powerful loop of mutual reinforcement.

7 Glossary

- **Openledger:** A decentralized next-gen Blockchain network for AI that provides trust infrastructure for specialized AI models.
- **OPEN Token:** The fundamental token of the OpenLedger ecosystem, used for staking, governance, and transactions.
- **Specialized Models:** AI models fine-tuned for niche tasks using high-quality datasets from OpenLedger's Datanets.
- **ModelFactory:** A GUI-based platform for secure and efficient fine-tuning of AI models.
- **OpenLoRA:** A multi-tenant system for serving fine-tuned LoRA models with minimal overhead.
- **Datanets:** A secure repository for collecting and managing datasets used for training specialized models.
- **Retrieval-Augmented Generation (RAG):** A technique for generating AI outputs with citations and source attribution.
- **Proof of Attribution:** A system for tracking and verifying the contributions of data providers and model creators.
- **Flywheel Effect:** A self-reinforcing cycle of growth where increased participation and innovation drive further adoption and development.
- **Multi-Tenant GPU Infrastructure:** A system that enables efficient sharing of GPU resources across multiple AI models.

- **Segmented Gather Matrix-Vector Multiplication (SGMV):** An optimized CUDA kernel for accelerating LoRA model execution.
- **Decentralized Governance:** A decision-making framework where stakeholders participate in proposals and voting.
- **Tokenomics:** The economic structure of the OPL token, including supply, distribution, and utility.
- **DataInf:** An efficient method for computing data influence in AI models.

8 References

References

[1] Zhao, B., et al. (2024). "DataInf: Efficiently Estimating Data Influence in LoRA-Tuned LLMs and Diffusion Models." ICLR 2024. https://arxiv.org/abs/2310.00902

[2] Chen, L., et al. (2023). "Punica: Multi-Tenant LoRA Serving." arXiv:2310.18547. https://arxiv.org/abs/2310.18547