Distributed AI Training Network with Proof of Learning and Privacy Preserving Techniques

Jure Snoj[1], Gal Gantar[1], and Andrew Klein[1]

[1]DKLabs Labs

August 23, 2024

Abstract

The rapid growth of Artificial Intelligence (AI) and Machine Learning (ML) demands unprecedented computational resources, leading to bottlenecks that traditional computing architectures fail to address. DKLabs.AI proposes a decentralized network leveraging blockchain technology to connect AI developers with idle GPU resources, creating a scalable, efficient, and cost-effective solution for AI training. Our system prioritizes security and trust by ensuring computational integrity through Proof of Learning (PoL) and data privacy with Trusted Execution Environments (TEEs). Consumers (AI developers) can construct detailed orders specifying model training requirements, which are then matched with bids from Compute Providers. These providers, organized under Validators, perform the training and generate PoL to verify the accuracy and authenticity of their work. Validators ensure the training's correctness and facilitate secure data handling using advanced cryptographic techniques. By utilizing distributed and federated learning methods, and employing robust data compression techniques for efficient communication between nodes, our network enhances the scalability and reliability of AI training processes. This decentralized approach not only accelerates AI development but also democratizes access to computational power, encouraging innovation while maintaining stringent privacy and security standards.

1 Problem and Motivation

The increasing demand for Artificial Intelligence (AI) and Machine Learning (ML) has led to a significant surge in computational requirements. As AI models become more complex and data-intensive, the need for scalable and efficient computing resources has become a major bottleneck. Traditional computing architectures are no longer sufficient to handle the massive amounts of data and computational power required for AI training.

2 The Limitations of Current Solutions

Moore's Law, which states that computing power doubles approximately every two years, is no longer applicable. The rate of progress in computing power has slowed down, and the industry is facing a significant challenge in keeping up with the demands of AI computing. Domain-specific hardware, such as Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs), has improved performance, but it is not enough to meet the growing demands of AI training.

3 Ensuring Trust and Verification in Distributed Systems

One of the primary challenges in establishing a distributed AI training network is ensuring that computational tasks are executed precisely as intended. Without this assurance, participants might misuse their computing power, underperform, or overcharge, ultimately degrading the system and hindering real adoption due to a lack of trust. This underscores the need for robust mechanisms to verify that participants perform their assigned computations accurately, thereby maintaining the network's integrity and trustworthiness. More on how we achieve this in a later section.

4 Addressing Data Privacy

The importance of privacy in training AI models and using user data cannot be overstated. Ensuring user privacy safeguards sensitive personal information, preventing misuse and unauthorized access. It helps build and maintain trust between users and organizations, which is critical for the widespread adoption of AI technologies. Respecting proprietary models and data preserves intellectual property rights and encourages innovation by protecting the interests of developers and organizations. Moreover, stringent privacy measures are often a legal requirement, ensuring compliance with data protection regulations and avoiding potentially severe legal ramifications. Therefore, any distributed AI training network must prioritize the protection of user data throughout the training process.

5 The Need for Distributed Computing

Distributed computing has emerged as a crucial solution to address the scalability and efficiency challenges in AI training. By harnessing the collective power of multiple computing resources, distributed computing enables the processing of large datasets and

complex AI models. However, current distributed computing solutions are often limited by technical challenges, such as data parallelism, model synchronization, and communication overhead. The Opportunity for a Distributed AI Training Network The increasing demand for AI computing resources has created a significant opportunity for a distributed AI training network. By connecting available GPU compute resources with AI developers who need them, a distributed AI training network can provide a scalable, efficient, and cost-effective solution for AI training. Such a network can accelerate the development of AI applications, enable the training of more complex models, and drive innovation in various industries.

6 The Vision

Our vision is to create a leading distributed AI training network that bridges the gap between available computing resources and AI developers. By providing a seamless and efficient platform for distributed AI training, we aim to accelerate the adoption of AI and drive innovation in various industries. Our network will enable AI developers to focus on developing and training AI models, while we handle the underlying computing infrastructure.

7 Architecture

DKLabs.AI is a purpose-built L1/L2 blockchain network specialized for AI training tasks. It connects AI developers to compute resources and directly rewards the providers of these resources for their contributions. This ensures that AI developers always have access to essential compute and GPU resources, while incentivizing compute providers to utilize their idle GPUs through rewards. Our goal is to build DKLabs.AI into the most scalable and affordable decentralized GPU network, while maintaining the crucial properties of transparency, verifiability, and decentralization.

To achieve this, we need to:

- Ensure that anyone can easily and permissionlessly join the network.
- Enable anyone to verify that the training was actually performed by the compute providers.

We have designed a system that separates actors into three distinct roles: **Consumers**, **Validators**, and **Compute Providers**.

Consumers: These are AI developers looking to leverage available compute resources at a lower cost, while being assured that they are receiving the compute power they paid for.

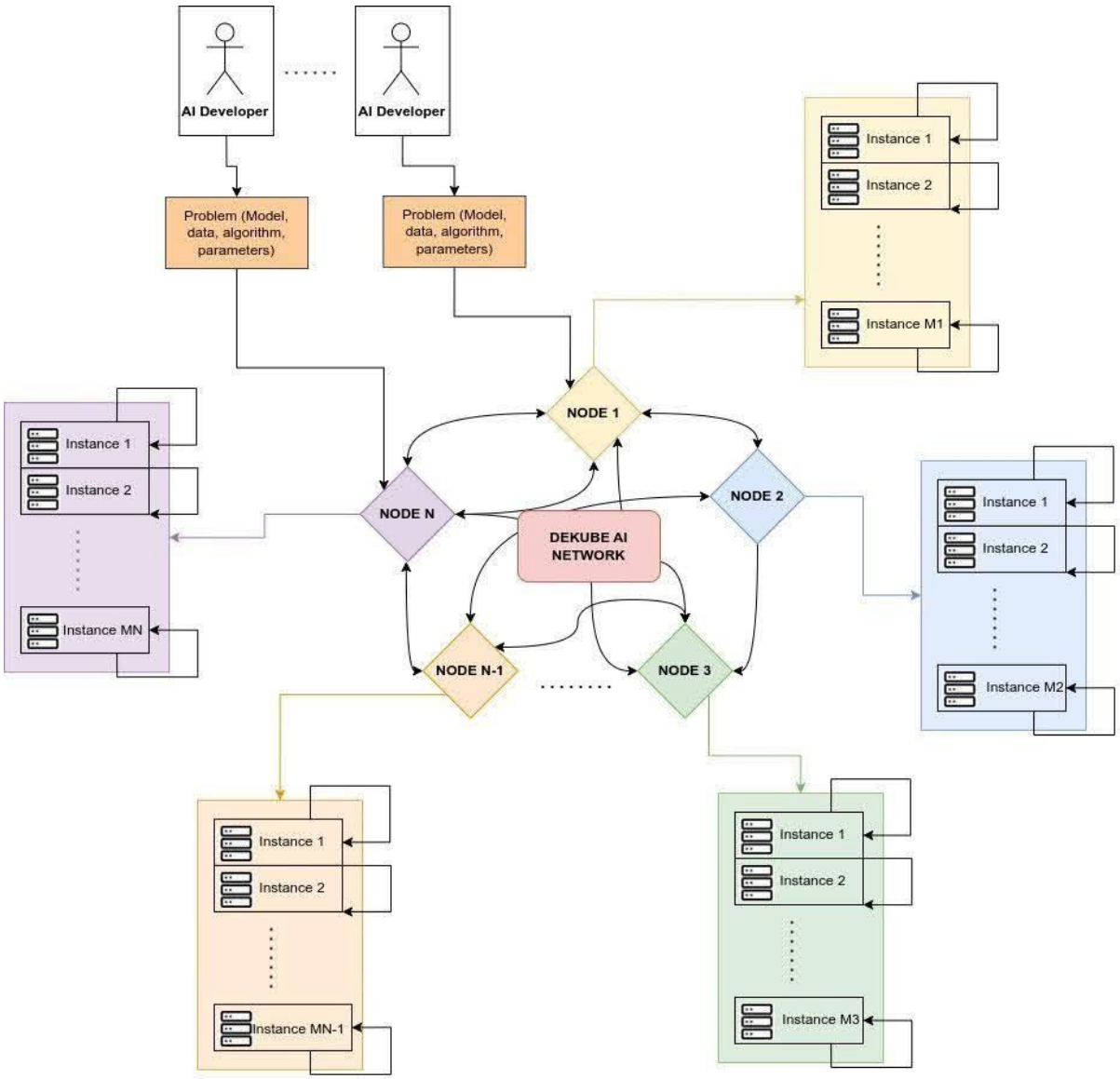

Figure 1: Architecture overview

Consumers can create orders where they define their problem (model, data, algorithm, parameters, etc.) and the conditions of the order, such as the amount of GPU power, training time, the price they are willing to pay, or even more sophisticated conditions related to privacy, geolocation, regulation, etc. **Compute Providers/Workers**: These individuals provide their available spare compute resources in exchange for rewards from the network. This role is distinct because we want to enforce minimal requirements for people to join the network. Instead, we transfer all the more complex tasks to the validators who are more specialized. **Validators**: These nodes participate in consensus and implement all the logic of the network. They run the software to match and execute the orders and prove that everything was done correctly. Like in any blockchain network, the nodes are the entry point to the system and facilitate communication between different parties. Consumers send their orders to the Validators, who then pass them on to the

Compute Providers and ensure everything is done correctly.

7.1 Workflow

To start with a simplified explanation, a typical workflow is as follows: The consumer creates an order for a certain amount of compute, specifically for model training. The nodes bid for the order, and the order gets matched against the most attractive bid. The node that won the bid initiates the distributed/federated learning process with the workers in the validator's cluster. Each of the workers performs the training on the data it has received, generates proof of training, and sends the results to the validator. The validator verifies that the results are valid and iterates steps 3-5 as long as necessary.

7.2 Detailed Design

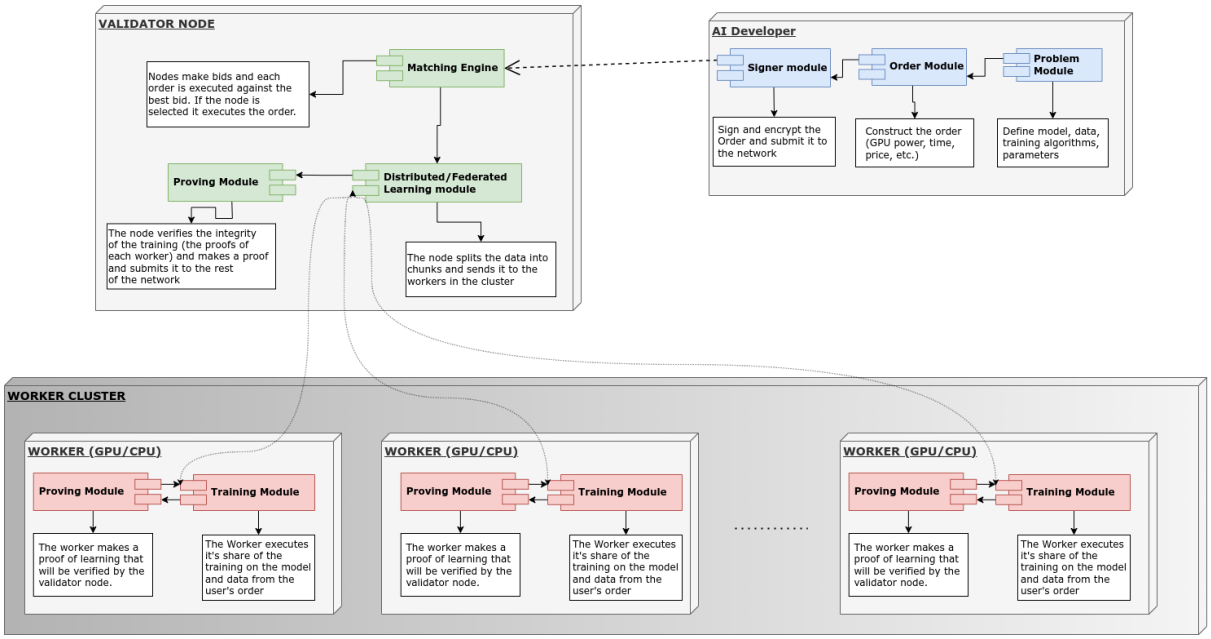

Figure 2: Design schema

1. **Order Construction** The consumer (AI developer) can construct an order for a certain amount of model training. The order includes the Problem Definition and the Conditions. Problem Definition: This includes the model to train, the initial weights, the data to train on (which can be private or public), the algorithms to use, the parameters of the training, etc. Conditions: These describe the amount of GPU power, training time, the price they are willing to pay, and even more sophisticated conditions related to privacy, geolocation, regulation, etc. The consumer then signs the order and submits it to the network along with the payment, which is locked until the job is completed.

2. **Matching the Order** All the nodes (validators) collectively run a decentralized matching engine that is used to give consumers the best possible execution price. The order book includes bids from different nodes offering a certain amount of compute at a specific price. This design ensures that the nodes cannot extract excess value from the consumer but instead have to compete to offer the best price. When the consumer submits the order, it is matched against the best bid that satisfies all the conditions of the order and can be executed by the validator who made the bid.

3. **Initiating Distributed/Federated Learning** Each verifier in the network manages a cluster of compute providers. These providers are assigned to verifiers in a way that optimizes overall latency and training speed, using a clustering algorithm that considers factors such as geolocation, internet speed, and hardware similarity. The amount of compute that can be allocated to a node is determined by the amount staked and can be reduced if the node misbehaves or goes offline for an extended period. The verifier also keeps track of the size and capability of each worker in its cluster to appropriately distribute the work. When the order is executed by the verifier, it splits the work into chunks of different sizes based on the size of the workers in the cluster and sends each chunk to the appropriate worker. The technical details of this process are explained in the "Technical Challenges" section below.

4. **Training the Model** This step involves the actual training of the model within the verifier's cluster that won the bid to execute the order. The training work, having been split into chunks, is distributed to the workers in the cluster. The workers execute the training based on the Problem Definition from the consumer's order. During each step of the training, workers store auxiliary information that will be used to prove that the training was executed correctly. This proof is crucial as it prevents system abuse, ensuring that compute providers are rewarded proportionally to the amount of work they do and that consumers receive the training they paid for. Once the training is complete, the workers send the results along with the proofs to the verifier.

5. **Verification** Once the training is finished, the verifier aggregates all the updates received from the workers and verifies each update against the proof. The technical details of this verification process are explained in the "Technical Challenges" section below. The verifier only accepts valid updates, meaning that if a proof is invalid or if a result is not submitted, the worker will not be rewarded. The verifier then creates a proof of execution and submits it to the network. After this, there is a challenge period designed to prevent validators from intentionally censoring or not including an update. If this occurs, a challenge can be submitted during

this period to force the validator to include the proof, or else the validator will be penalized. After the challenge period, the job is registered as completed, and the results of the training are exchanged for payment through a decentralized escrow system using zero-knowledge (ZK) proofs, eliminating the need for a trusted third party. The technical details of this process are also explained in the "Technical Challenges" section below.

8 Summary

Our design achieves all the desired properties: decentralization, full verifiability, and is completely permissionless for anyone to join. It can be built upon mature, battle-tested technology stacks such as the Cosmos SDK, Polkadot Substrate, or Ethereum L2 CDKs, allowing us to concentrate on the unique aspects of our application. By separating the roles of verifier and worker nodes, we impose minimal requirements on compute providers. The sophisticated tasks—such as validating order execution and participating in consensus—are handled by the validators. This inclusivity means that anyone, regardless of their specific hardware setup, can join the network. This includes GPU farms, small home rigs, and even consumer devices like laptops and mobile phones. Future developments could enable the secure use of private data from these diverse devices through federated learning, opening up new, valuable, and underexplored use cases.

9 Future Expansion and Opportunities

While our primary focus is on delivering a best-in-class distributed AI training network, we recognize the vast demand and opportunities in this space. To address these opportunities, we have identified several areas for future expansion, which align with our mission and leverage our existing strengths.

1. **Model Hosting** Building on our current design, we plan to expand our platform to support model hosting, tapping into the growing demand for large language models (LLMs) and the associated costs of running them. Many models today are fine-tuned on large open-source LLMs, sharing a significant portion of weights in the early layers. This commonality allows us to batch operations efficiently, optimizing performance and making these models economically viable for a broader range of use cases.

2. **Live Context Retrieval** To further advance AI development, we aim to integrate live context retrieval, enabling models to learn from fresh information and inform decisions in real-time. This capability is currently limited to a few large companies, such as Google and Microsoft, which have built extensive knowledge graphs. We

are exploring partnerships with DePIN providers building such knowledge graphs, like Grass Network, to accelerate their progress and leverage their graphs to deliver relevant context to our users.

3. **Model and Data Sovereignty** Our platform's design facilitates partnerships with other platforms focused on model and data marketplaces, vaults, and payment mechanisms for AI agents. By integrating these services, we can provide a more comprehensive and streamlined experience for our users, while attracting new users from other platforms.

4. **Federated Learning** We are dedicated to exploring the promising field of federated learning, which allows users to train models locally on their private data without compromising its integrity. Our platform is ideally suited to support this approach, offering a secure environment for local model training. In this process, users execute the model on their devices, such as smartphones, make necessary updates, and return the updated gradients with added noise to enhance privacy. This ensures robust data protection while maintaining model accuracy.

5. **Quantum GPU cooling** We are engaged in discussions with a quantum computing lab regarding a potential partnership to offer proprietary GPU cooling systems, which would significantly enhance the efficiency of smaller graphics cards or laptops. By white-labeling these products, we can incentivize users to join our network and increase their earning potential.

These future expansion areas align with our mission to deliver a scalable, efficient, and decentralized AI training network, while opening up new opportunities for growth and innovation.

10 Technical Details

10.1 Privacy

Privacy in AI model training is crucial for safeguarding personal information, building trust, protecting intellectual property, fostering innovation, and complying with legal regulations. At DKlabs.AI, we prioritize privacy as a central concern. To achieve this, we employ a combination of end-to-end encryption, hashing, zero-knowledge proofs (ZK proofs), and Trusted Execution Environments (TEEs). Crucially, the model and user data are never shared publicly across the entire network but are privately communicated to the verifier node executing the order. This data is then split into smaller chunks, with each worker only accessing its assigned piece.

While this setup offers significant privacy protections, we can enhance it further. In our system, all verifiers run Trusted Execution Environments (TEEs), which are common in newer hardware. This allows us to ensure that the order is filled correctly while also maintaining privacy. Consequently, no entity throughout the entire training process sees more than a tiny portion of the data. For even greater privacy, these data chunks can be further divided and processed by multiple verifiers.

Substantial research is focused on efficiently training models on encrypted data using techniques like fully homomorphic encryption, secure multi-party computation, and differential privacy. These advanced methods can be integrated into our platform in the future or offered by specialized validator clusters, in which all workers would leverage these technologies, thereby providing even stronger privacy guarantees.

10.2 Distributed learning

Distributed learning uses multiple devices to collectively train a machine learning model, improving efficiency and scalability. In a system where each consensus node manages a cluster of validator nodes, these nodes contribute to distributed learning through data and model parallelism. Data parallelism lets each validator node train the same model on different data subsets, then aggregate the results. Model parallelism splits the model into parts, with each validator node handling a segment. This allows for larger models and more complex computations. Integrating distributed learning into this system enhances efficiency and robustness by utilizing the computational power of validator nodes for faster convergence and higher-quality models.

In the context of training larger models such as fine tuning LLMs, stable diffusion etc. both data parallelism and model parallelism can benefit the training process.

10.3 Ray network

We will use the Ray network, an open-source scaling framework, to achieve efficient distributed learning and verification of proofs in our architecture. Ray will help us manage the distribution of tasks across consensus and validator nodes, optimizing both data and model parallelism. By leveraging Ray's ability to handle large-scale computations and seamless task scheduling, we can ensure that each validator node processes its portion of the data or model efficiently. Additionally, Ray's robust framework will facilitate the verification of proofs by coordinating and aggregating results from multiple nodes, ensuring the integrity and accuracy of the computations.

10.4 Data sharing between distributed training agents

In distributed SGD, most gradient exchanges are redundant, and the Deep Gradient Compression (DGC) algorithm significantly reduces communication bandwidth by compressing gradient data, leading to more efficient training. AQ-SGD compresses activation changes across epochs, allowing decentralized networks with slower connectivity to perform nearly as well as high-speed networks. ZeRO++ uses quantization and data remapping to further cut communication volume while preserving model quality. These methods can reduce the data exchanged between GPUs for AI training to a fraction of the original amount. We aim to develop a similar system that achieves NVLink-like performance, which uses high-speed wire-based connections, but without the need for physical cables. Our goal is to replicate NVLink's efficient data transfer in a wireless setup.

10.5 Proof of model training

To implement Proof of Learning (PoL) in our network, we will integrate the PoL mechanism to ensure that each worker node is genuinely contributing to the model training process rather than performing arbitrary computations. This involves each worker node generating a PoL that includes model-specific information, data point details, signatures, and auxiliary information for every training step. The PoL encapsulates all necessary elements to recreate and verify the training process. By publishing an encrypted variant of model weights and ensuring that the recreation of weight values without knowledge of the entire dataset is computationally challenging, we can deter adversaries from faking their PoL. This approach guarantees that only authentic training efforts are recognized, as any discrepancies in the PoL will be detected with high probability, ensuring correctness, security, and verification efficiency. Thus, integrating PoL into our distributed learning framework enhances trust and integrity, verifying that each worker node is genuinely advancing the model training. [2, 3, 1, 4]

10.6 Trusted Execution Environments (TEEs)

Trusted Execution Environments (TEEs) are secure areas within a processor that ensure the integrity and confidentiality of code and data loaded inside them. They are particularly useful for running sensitive computations because they protect against both software and hardware attacks, providing a secure enclave for operations. TEEs work by isolating applications from the rest of the system, using cryptographic techniques to verify the integrity of the software running in the environment. As security concerns grow, TEEs are becoming increasingly common in modern hardware, from smartphones to cloud servers. This widespread adoption makes TEEs a crucial tool for enhancing privacy and security in various applications, including privacy-preserving machine learning.

References

[1] Kasra Abbaszadeh, Christodoulos Pappas, Jonathan Katz, and Dimitrios Papadopoulos. Zero-knowledge proofs of training for deep neural networks. Cryptology ePrint Archive, Paper 2024/162, 2024. https://eprint.iacr.org/2024/162.

[2] Hengrui Jia, Mohammad Yaghini, Christopher A. Choquette-Choo, Natalie Dullerud, Anvith Thudi, Varun Chandrasekaran, and Nicolas Papernot. Proof-of-learning: Definitions and practice, 2021.

[3] Hengrui Jia, Mohammad Yaghini, Christopher A. Choquette-Choo, Natalie Dullerud, Anvith Thudi, Varun Chandrasekaran, and Nicolas Papernot. Proof-of-learning, 2024. Accessed: 2024-08-03.

[4] zkPoTs. Kaizen, 2024. Accessed: 2024-08-03.