Atoma Network Whitepaper

Jorge António Hisham Khan

July 22, 2024

Abstract

The Atoma Network is a decentralized and permissionless protocol for verifiable AI inference. Atoma relies on compute providers, referred to as execution nodes (or simply nodes), which are responsible for hosting and running AI models in order to process incoming requests.

Atoma will serve a crucial role as an off-chain AI execution layer for Web3 protocols, utilizing blockchains for resource coordination and settlement of executed requests. It will integrate AI capabilities into smart contracts, compensating for blockchains' inability to perform heavy computations, such as AI inference. In addition to enhancing smart contracts, new applications will be built on Atoma, such as chat applications specific to Web3 protocols, AI-driven market prediction platforms that analyze and forecast trends using advanced algorithms, AI-enhanced wallets specialized for user intent interactions, knowledge bases as public goods, social discussion forums, DAO and Network States governance, and a wide range of innovative applications that leverage verifiable AI to transform our digital interactions.

In a world where AI will play the role of our caretaker, tutor, personal assistant, and co-worker, it will be crucial that these applications have high computing integrity guarantees. For this purpose, we have designed a new consensus protocol, which we refer to as *Sampling Consensus*. Our protocol is based on the assumption that the participating nodes are rational actors in a competitive economic environment. The protocol is optimized to achieve very high compute integrity guarantees whilst having a lower cost compared to other methods (such as zkML or opML).

Using its novel approach, the network aims to foster a future characterized by increased transparency, innovation, and democratic principles, ensuring that technology fully aligns with the needs and values of users and communities.

Contents

1 Introduction

Current and Future Trends in AI: In recent years, groundbreaking generative AI capabilities have emerged, primarily driven by large language models (LLMs). These advancements are largely attributed to the exponential increase in computational power, facilitated by cutting-edge hardware that delivers high throughput at reduced costs. This progress has enabled the training of even larger models on increasingly vast datasets.

Following OpenAI's release of ChatGPT, we have seen better, more capable models being released at an extraodinary pace. These include both proprietary models (the likes of GPT, Claude, etc.) and open-weight models (such as Llama, Mixtral, Qwen, etc.). In addition, these models are growing rapidly both in size and in capability. The most advanced models, such as GPT-4, Claude, or Llama3, contain billions, or even trillions, of parameters.

This trend is expected to continue in the coming years, leading to unprecedented growth in different industries. For example, McKinsey estimates that generative AI could lead to worldwide economic benefits of $2.6 trillion to $4.4 trillion annually, in a wide variety of use cases. This level of growth will be accompanied by an increase in demand for computing and energy resources. Through the first half of 2024, we have seen experts in the field highlighting the need for governments and big tech companies to cooperate in order to build the next generation trillion dollar compute cluster, as in Leopold Aschenbrenner's most recent work *Situational Awareness*, see [19].

Whether this trillion-dollar cluster will materialize in the future remains to be seen. AI pipelines require specialized hardware, such as GPUs and TPUs, and while GPUs are widely accessible to the public (mostly for gaming purposes), the type of GPUs required for AI tasks are of a higher caliber and considerably more expensive. The fact that there are only a few large GPU manufacturers (such as Nvidia and AMD) combined with the current geopolitical tensions leads to serious disruptions in the supply chains of chip manufacturing. As a consequence, making high-performance GPUs for AI accessible to retailers and small to medium-sized data centers remains challenging until there is a significant increase in the availability of compute resources.

If this trend continues, we will step into a future in which big tech giants retain most of the world's available compute resources, in which case, these companies will have a complete monopoly over most of AI development from model training to AI model deployment and inference. Moreover, to maintain their profit margins and protect their intellectual property, these large tech companies will continue to advocate for AI to remain a closed-source technology, meaning model weights will not be available to the general public.

Considering that only a few companies have access to virtually all the real-world data being generated, we may witness extreme centralization of one of the most revolutionary technologies of our time: generative AI. These companies collect the personal data of their users, which is then used to train new

models. These models can lead to the development of new products, often offered on a subscription basis to the same users. This scenario results in a flawed market where personal data is continually exploited to generate new revenue streams through AI, primarily benefiting big tech companies at the expense of their customers.

Moreover, such a high level of centralization threatens AI alignment with user needs and values, leading to AI misalignment. This misalignment has already been observed with the release of the highly biased Google Gemini image generation model [5]. Lastly, nation-states may restrict their citizens from accessing AI services offered by foreign companies, as AI could become a focal point in national security, creating an even larger gap between those that can benefit from AI and those who cannot.

The only way to avoid this dangerous scenario is to develop more fair and transparent technologies that can help mitigate these risks.

Democratizing AI through Atoma: Decentralized protocols provide the right framework to mitigate the above risk factors.

Using cryptoeconomic principles and decentralization, blockchains have demonstrated their ability to accumulate extensive computational power. For example, at the time of writing, the Bitcoin network processes approximately 570 exahashes per second [2], with an energy consumption comparable to that of a nation state, such as Poland [1]. This scenario demonstrates that decentralized networks have the potential to efficiently coordinate rational agents to pool sufficient computational resources to handle the most demanding AI tasks available today.

However, blockchains are generally unsuitable for running AI inference on large language models (LLMs) due to their constrained execution environments. Performing AI inference on the majority of blockchain nodes incurs high costs and substantial bandwidth consumption. Most blockchain execution environments are limited to processing only transactions of a few hundred kilobytes. Furthermore, nodes are not required to utilize specialized hardware, making it impractical to run AI inference on blockchains.

For the above reasons, we have designed Atoma, a decentralized protocol specializing in verifiable AI inference. Using a modular approach, Atoma decouples the execution layer from the settlement layer. Through Atoma, it is possible to aggregate the nodes necessary to run any sort of AI inference without incurring high bandwidth and replication costs on the network.

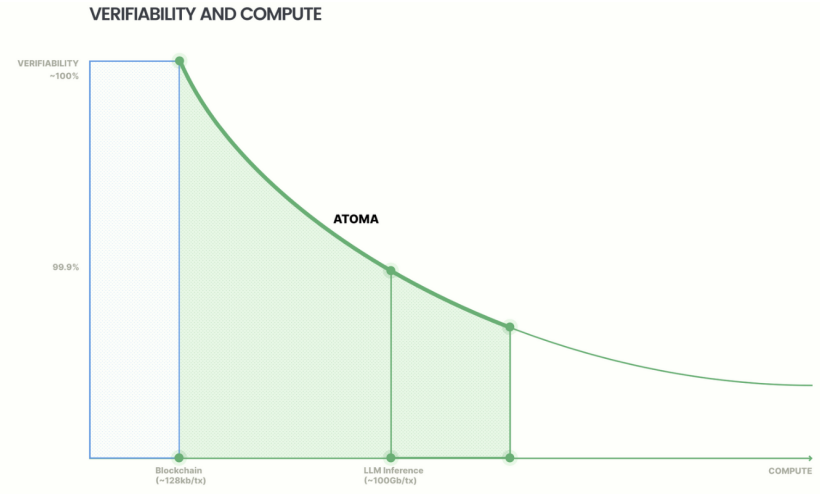

Figure 1: Blockchains lack execution environments capable of handling heavy computations such as AI inference.

Compute Layer: Atoma nodes will be able to monetize their compute resources in a transparent and permissionless environment while being exposed to fair-market economic incentives. Nodes from around the world can participate in the network with their specialized hardware to provide useful compute. Moreover, Atoma's infrastructure is highly optimized for efficient GPU utilization, incorporating

the most advanced LLM memory management techniques (such as FlashAttention, PagedAttention, quantization techniques, etc.). These optimizations will allow the network to scale both horizontally with the number of available nodes on the network and vertically, with better hardware becoming available on the network. This is especially relevant given the current GPU chip shortage, as hardware providers face sunk costs by not using their infrastructure efficiently.

Decentralized networks have the ability to aggregate heterogeneous computing resources from around the world within a free market. Consequently, they can offer considerably lower prices than current centralized cloud providers. This has already been confirmed in practice by networks such as Aethir and others.

The Atoma Network will be powered by its native TOMA token. With a strong tokenomics design, compute providers, protocol participants, and developers will be rewarded for their contributions, adding value to the network. The TOMA token will serve multiple purposes: It will be required to pay for AI inference fees, used by nodes to deposit collateral for network participation, and used to compensate developers who build new applications on the Atoma platform. Additionally, the TOMA token can subsidize nodes to acquire more advanced hardware over time. By bootstrapping the network's computing power through targeted inflation, Atoma can increase its value and offer more affordable inference services to end users.

Verifiability Layer: In a decentralized environment, applications cannot rely solely on a single party for operation without additional integrity guarantees due to the risk of misbehaving nodes. Rigorous integrity checks are therefore fundamental to maintaining trust in decentralized systems. We use the term *verifiable AI inference* to refer to any AI inference computation that offers high integrity guarantees. There are a few approaches to verifiable AI, such as consensus, zkML, opML, and TEEs. We have developed a novel consensus protocol, referred to as *Sampling Consensus*, specialized for heavy computations (which includes AI inference). We believe that the combination of our consensus protocol with TEEs provides a more cost-effective and faster solution for verifiable AI compared to other methods.

The Sampling Consensus protocol is elastic, allowing users and protocols to decide the output integrity guarantees that their use case requires. The advantage is that end-customers pay fees proportional to the integrity level they choose for what their application requires.

In the context of a DeFi protocol, it is crucial to ensure absolute certainty in the accurate computation of each AI market prediction, as even minor deviations can significantly affect fund management. In such high-stakes scenarios, the nodes might attempt to compromise the protocol by acting maliciously, with the potential to extract a large value from it. In contrast, in an on-line community-based chat application, the incentive for a node to act maliciously is much lower. However, it remains vital that the protocol guarantees that a node hosts the appropriate model for the chat application rather than an inferior one, while charging fees for the more advanced model. This ensures that users do not overpay for substandard services and helps to maintain the reputation of application developers and teams.

In the final scenario, the risk to the user and the potential rewards for malicious behavior by the serving node are considerably lower than in the DeFi application example, where the risks can be unlimited. This enables the system to operate securely over the long term, achieving a Nash equilibrium with minimal replication overhead costs. In practice, this means that most requests are processed at native costs while maintaining high levels of security guarantees.

Our Sampling Consensus algorithm is well adapted for each of the above use cases, being highly flexible and optimized for different scenarios. We are also actively exploring other verifiability techniques, including the use of Trusted Execution Environments (TEEs), which allow for high levels of verifiability while also reducing the cost of verifiable execution closer to native cost. Moreover, TEEs can provide both model and data privacy guarantees. We believe that user data privacy will unlock the true potential of generative AI, allowing one to build more complex applications that rely on a user's personal information, without having plain access to the data.

Application Layer: On the demand side, Atoma will provide a platform to build next-generation applications that are at the center of AI and Web3. These will include personal AIs and their coordination, chat applications, wallet enhancements through LLMs (enabling user wallet interactions

through declarative intents), online data scraping, next-generation browsers and browser extensions, content generation, community-based knowledge bases, social media platforms, and much more.

Over time, the Atoma network will evolve to allow the scalable development of personal AIs. This development means users can delegate any AI computation tasks on their personal data, creating a continuous stream of information that is curated, enhanced, and summarized.

The relevance of Atoma for personal AIs is particularly prominent when considering the privacy and resiliency needs of personal data and personal AIs. Through the use of TEEs for data privacy and integration with decentralized storage solutions, personal data is ensured to be securely managed and maintained. Moreover, unlike Web2 companies, which can be compromised and may not withstand the test of time, Atoma's decentralized framework in combination with decentralized storage solutions ensures that personal data are protected from such vulnerabilities. This high level of privacy and resilience is crucial, as personal AIs can continue to provide value and maintain a digital footprint of each user for future generations.

Moreover, the creation of robust and expressive personal AIs requires extensive resources to run seamlessly on a continuous stream of personal data. Local execution alone is insufficient for these demanding tasks. Atoma's decentralized computational capabilities provide the necessary infrastructure to support the extensive resource requirements of personal AIs, ensuring that they operate efficiently and effectively.

Similarly to the advent of search engines, personal AIs will dramatically change the way we interact with the web as a whole. Automation tools, such as smart agents, can rely on our personal AIs, allowing us to have curated and user-crafted experiences while navigating the digital world. Data and content can be directed to the user's interests and needs.

The development of new AI models will be crucial for the future of Atoma. We believe that open-source AI development will democratize and broaden the reach of AI technologies across societies. We are committed to incentivize the open source AI community to develop and deploy new AI models while being able to monetize their work. This allows experts in the community to earn a continuous stream of revenue through their use of the model.

As a decentralized platform, Atoma will focus on combining efforts that will ultimately lead to a new era of use cases within decentralized applications.

2 Atoma as a decentralized permissionless protocol for AI inference

In this section, we will explore the architecture behind Atoma, including how nodes can register on the network, how these can subscribe to AI models, how requests are processed, and how rewards are accrued by nodes operating on the network. At the heart of the network lies the *Atoma on-chain logic*. The latter refers to a collection of blockchain-agnostic smart contracts that together are responsible for managing the state of the network. Including:

1. Maintaining a registry of registered nodes operating on the network.
2. Keeping track of which AI models are currently subscribed to by nodes.
3. Managing node deposited collateral and accrued rewards.
4. Selecting which node(s) to process a given request.
5. Contributing to our Sampling Consensus protocol.
6. Settling any unresolved request once every selected node has committed and agreed on the output state. In case of disagreement, a dispute is solicited by the Atoma smart contract.
7. Handling the TOMA token emission schedule.
8. AI model whitelisting and Atoma protocol governance.

2.1 Atoma's node registry

In order to ensure the modus operandi of the network, it is crucial to maintain a registry of which nodes provide compute on behalf of Atoma, which fees these nodes have accrued over time by processing requests, and which AI models are being utilized on the network. In order for the network to be fully decentralized, it is crucial that this registry is not maintained by a single entity. Instead, the natural choice is to deploy this logic as a series of smart contracts on a blockchain.

Whenever an entity (being a single user, a company, a DAO, etc.) wishes to delegate compute resources to the network in order to accrue fees, it should first register on the network, through the on-chain logic. Registration should be handled by a simple transaction call to the on-chain smart contract. Upon registration, the smart contract will emit a TicketId for that entity. Owning a TicketId will allow the entity to later withdraw the rewards accrued over time.

The network operates in a series of linear epochs, which are defined as fixed time periods (e.g., one day, one week, etc.). When a node submits a registration transaction to Atoma, it becomes eligible to process requests starting from the next available epoch. Similarly, if a node deregisters, it can only withdraw accrued fees in the next epoch.

2.2 Node collateral, slashing, and rewards

As mentioned above, Atoma nodes are required to deposit a given amount of collateral upon registration. This collateral is subject to slashing in the cases of malicious behavior or when the node becomes unresponsive (it stops processing requests for a long period of time). The nodes are only able to withdraw that collateral in the epoch following deregistration.

If a node timeout for a given request is set, a fixed percentage of its deposited collateral is slashed. Part of the slashed collateral goes to the user who initiated the request to compensate users for poor service. Another part of the slashed collateral goes to the Atoma community treasury fund, and the remaining part is burned. This incentivizes the nodes to always be responsive; otherwise they lose part of their stake in the network while compensating both the user (with a refund) and the network through treasury fund allocation.

If a node acts maliciously, it is penalized with all of its deposited collateral being slashed. Similarly to the timeout case above, a percentage of the slashed collateral goes to the user, compensating for any possible malicious behavior. There is also a percentage of the slashed collateral that is allocated to the honest nodes involved in processing the request, the Atoma nodes that participated in the dispute event, and the Atoma community treasury fund, with the rest of the remaining collateral being burned.

On the other hand, for each request processed honestly, a node is entitled to receive part of the fee paid by the user that initiated the request. Request fees are paid in TOMA tokens, and their price is set by a fair free market mechanism. At the end of each epoch, Atoma nodes within an echelon bid the TOMA fee value, per text token, for the next epoch. Therefore, fees are computed in terms of the total number of text tokens required to process a given request. So, for example, if a prompt text request has 100 input tokens and its output response contains 200 tokens, then the total fee amount paid to a node to process request will be calculated as:

$$(100 + 200) * \text{token\_fee} = 300 * \text{token\_fee}.$$

Nodes accrue fees per epoch. Moreover, nodes can only withdraw their accrued fees for a given epoch once that epoch has ended.

2.3 AI model subscription

Atoma nodes must subscribe to the AI models they intend to host within the same registration epoch. If a node fails to subscribe within this timeframe, it will not be able to process requests for the specific AI model until the next epoch. Subscribing to a given AI model requires the node to specify its hardware characteristics (namely, the GPU cards it hosts, the number of cards it owns, etc.). Hardware specifications are uniquely identified by an EchelonId. In order to facilitate the model subscription, we will have a simple UI available with the list of all the EchelonIds, to which node operators can use to find the right EchelonId for their hardware characteristics. Specifying the hardware type is crucial for the network to be able to balance the request load effectively, across all operating nodes. Different

types of hardware will have different pricing, and users can choose how much they want to pay to process a request along a given echelon.

It is assumed that once the node subscribes to a given AI model, it will receive incoming requests for AI inference using that model. To run inference efficiently, it is essential that the node has the model weights available locally, avoiding unnecessary waiting times by having to download the model weights once an incoming request is received. The nodes will be able to download their model weights either from a decentralized storage (such as IPFS) or from an external API call (such as Hugging Face).

The node can unsubscribe from a given AI model at any time, immediately stopping the node from receiving further requests for that model.

If a node deregisters from the network, it will be automatically unsubscribed from all AI models to which it was subscribed.

2.4 Serving requests on Atoma

In order to fully understand the full scope of Atoma, it is important to understand what types of request can be processed by the network nodes. These include:

Definition 2.1. *Externally Owned requests* refer to those requests submitted to the network directly by an externally owned wallet (a wallet owned by a user private key).

Externally owned requests can be generated by a user interacting with a front-end UI, a desktop application, a mobile application, etc, which can include chat bots, AI-enhanced Web3 wallets, shared knowledge chat applications as public goods, content generation applications on the Web, online data scraping, etc. In this case, in order to submit a request to the network, a transaction to the on-chain logic contract needs to be signed and submitted to the blockchain, on behalf of a user's wallet.

Such a transaction specifies the hash of the input data, the hash of the actual request metadata (which includes prompt parameters such as temperature, random seed, etc.). The number of input tokens and the maximum number of output tokens are required to be contained in the given transaction, so the on-chain logic can deterministically calculate the fee to be paid to run the inference. The actual data of the request, which include both the prompt and the associated metadata, are not submitted directly to the blockchain (to reduce the overall gas cost of processing such a transaction, as the given data might be too large). The input data can instead be shared with the Atoma nodes either through an application's public API, through a decentralized storage (such as IPFS), or through a direct communication between the user and nodes (such as WebRTC).

Definition 2.2. We refer to *Smart Contract requests* as those requests submitted to the network through an on-chain smart contract external call.

Examples of smart contract requests consist of on-chain AI funds manager, AI generated NFT protocols, DAO governance, etc. In this case, the input request is submitted by the smart contract directly to the on-chain logic contract. This means that the data is fully public and must fit into the blockchain's execution runtime calldata memory limit. It is also likely that the smart contract requesting AI inference will need the output to be submitted back on-chain (think of the prediction market case above), although that is not always the case (e.g. AI-generated NFT protocols).

Definition 2.3. We refer to *Smart Contract Authenticated requests* as those requests submitted to Atoma through an externally owned wallet, but these requests require a smart contract authentication to be approved.

Examples of authenticated requests for smart contracts include the following: DAO governance decision making through AI in which a transaction might be submitted from a multisig, but it requires approval directly from the majority of the DAO community. Another interesting example consists of contract wallets (following Ethereum's account abstraction terminology) interacting with the network, leveraging user-defined permission logic with decision-making through AI (e.g., an AI is required to parse the web to obtain data information, and if that data satisfies specific well-defined logical criteria, blockchain transactions can be submitted on behalf of the wallet). The latter case is especially interesting in the context of the autonomous Smart Agents in Web3 protocols.

The Atoma nodes listen to blockchain events that are emitted once new request transactions are submitted to the on-chain logic contract. Upon listening to an event, a node can check if it has been selected to process the request or not. If it has been selected, it will start processing the request immediately. If the event contains all of the available input data (in the case of a smart contract request), it can start processing it immediately. Otherwise, it must retrieve the input data from an external API call to the application with which the user interacts (in the case of an externally owned or smart contract-authenticated request) so that it can start processing the request.

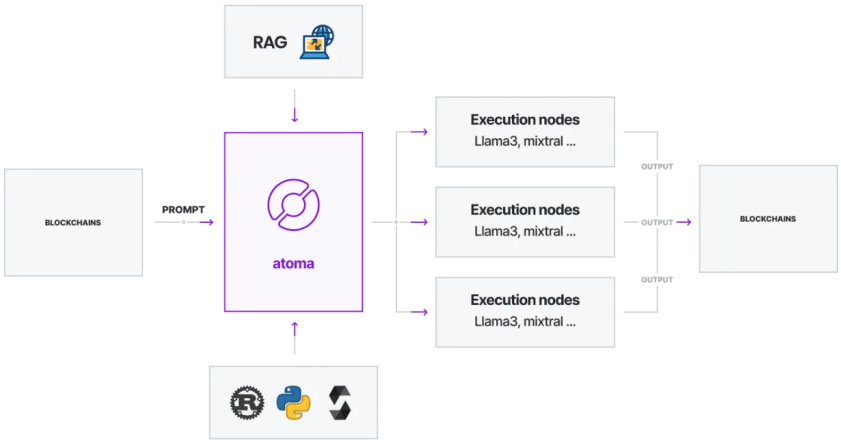

Figure 2: Nodes on the Atoma Network are selected to process different requests, through AI inference. Moreover, nodes can fetch data from other sources and execute general code. The Atoma on-chain logic coordinates nodes for processing requests and manages payments.

2.5 Node selection

For every request submitted to the network, the on-chain logic is responsible for selecting one or more nodes to process that request. A node is selected uniformly at random from the set of all nodes that subscribe to the AI model that the request requires. This random sampling mechanism helps balance the request load and ensures a fair distribution of the fees that the nodes are charged.

2.6 Sampling Consensus

A request might require multiple nodes to process it in order to provide high assurance guarantees that the generated response was not tampered with. In this case, the request is only settled by the on-chain logic once all the nodes have submitted a commitment to the output state, and these commitments all agree. More precisely, for a given request and a selected node to process it, that node must compute the output generated by running the AI model inference on the request's input. Once the output O is generated, the node must then submit a short cryptographic commitment to the output O, in the form of:

$$\big(H(H(O||1), H(O||2), ..., H(O||n)), \quad H(O||i)\big),$$

where $||$ denotes byte concatenation, n the number of selected nodes to process the request and i the node index in the array of all selected nodes, and H denotes a cryptographically secure hash function.

The fact that a node provides both the hash $H(O||i)$ and the n-ary Merkle hash

$$H(H(O||1), H(O||2), ..., H(O||n)),$$

mitigates the risk that nodes are lazy by waiting until another node commits first to the output state. Once another node submits to the output, a lazy node can simply copy that same commitment, without having to perform the actual computation. Since each selected node has a unique index in

the array $[1, n]$, for a selected node to be able to compute $H(O||i)$, it must have access to the actual value of the output O. Moreover, if two different nodes compute the outputs O and O', then

$$H(H(O||1), H(O||2), ..., H(O||n)) == H(H(O'||1), H(O'||2), ..., H(O'||n)),$$

if and only if $O == O'$, with almost certainty (since H is a cryptographically secure hash function). Once all nodes have submitted all their commitments, the on-chain logic can compare if all the submitted values were met.

$$(1) \quad H(H(O||1), H(O||2), ..., H(O||n)),$$

agree, and also verify that the hash of the leaves submitted $H(O||1)$, $H(O||2)$,..., $H(O||n)$ equals (1). From the above analysis, this ensures that all nodes have committed to the same output state. If one of the previous two conditions is not met, then a dispute mechanism is put forth by the on-chain logic contract. If we denote by $0 < p < 1$ the maximum percentage of nodes controlled by a single authority, then for a given request that requires n nodes for attestation, the probability that a tampered output is settled is

$$M = p^n.$$

In other words, such a probability decays exponentially with the number of selected nodes. For example, if $p = 0.5$ and $n = 10$, then $M = 0.5^{10} = 0.0009765625$. Moreover, the on-chain logic contract allows the user and/or the protocol to choose the number of nodes n to be selected for a given request, providing elastic verifiability depending on the task at hand.

This method of node sampling is particularly relevant for smart contract requests, as the latter require high assurance guarantees on the output state generated by AI models. For externally owned and smart contract authenticated requests, we can adapt the original Sampling Consensus protocol to reduce replication costs (that is, costs of having multiple nodes processing the same request).

2.7 Cross Validation Sampling Consensus

In order to address the replication costs within the original *Sampling Consensus* method, we have developed a slightly different protocol algorithm, which we refer to *Cross-Validation Sampling Consensus* which allows considerably lower costs, with a slight impact on verifiability. This method relies on the assumption that the nodes are rational actors in a competitive economic environment. In such a case, our protocol can be shown to converge to a Nash equilibrium in the long run. This protocol was independently discovered by the Hyperbolic Labs team in their research work [24].

Instead of the on-chain logic selecting n nodes at random for a given request, the contract simply selects a single node at random. Once the node commits to the output state, the contract then selects a quorum of $n > 0$ nodes with a protocol-defined probability p, which we also refer to as *replicationrate*, to recompute the output and attest to the integrity of the commitment to the original output of the first node. If the on-chain logic chooses to select n nodes, with probability p, then the output is settled once all the $n + 1$ nodes (the n nodes plus the original node) have committed to the output state and agree on it. If there is a disagreement, the on-chain logic contract entails a dispute mechanism, similarly to the previous case. On the other hand, if the on-chain logic does not select more nodes, which occurs with probability $1 - p$, then the output is settled once the original node has committed to the output state, without further verification.

We will show, in Appendix A, that in a rational competitive environment, cross-validation sampling consensus converges to a Nash equilibrium, in which nodes are incentivized to operate honestly. Under certain mild assumptions, one can show that

Proposition 2.4. *Under the assumption that a malicious entity controls at most 50% of the network, that the collateral at stake is at least* 250 *times higher than the actual cost of running native AI inference, and that the maximum reward for a node to act maliciously has a premium of 20% higher than the native cost. The network then reaches Nash equilibrium, in which the nodes are incentivized to act honestly, with a replication rate* $p = 0.00796$.

We will rephrase Proposition 1 in more technical terms and provide a proof in Appendix A. The above proposition implies that, to enforce the integrity of the network, the protocol only needs to sample more than one node to process the request, at most 0.8% of the time.

Conclusion 2.5. *According to Proposition 1, the protocol reaches Nash equilibrium if at least 8 in every 1,000 requests are processed by more than one node.*

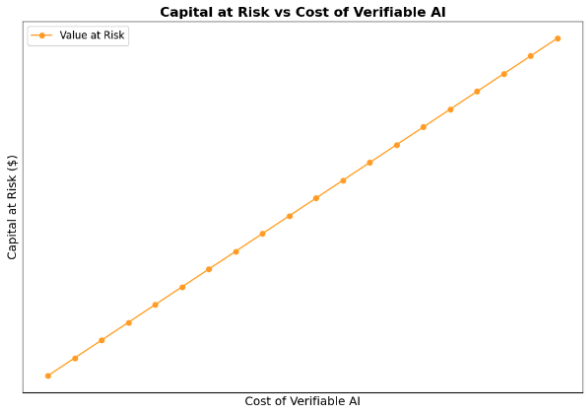

Figure 3: Externally owned request applications have capital at risk that increases linearly with the cost of verifiable AI inference.

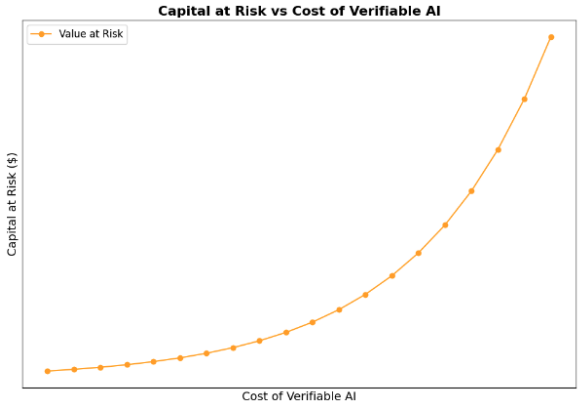

Figure 4: Smart contract request applications capital at risk dependency on the cost of verifiable AI inference, the former can be unbounded.

From the above assumptions, Cross-Validation Sampling Consensus is especially relevant for applications in which the nodes have little incentive to collude with the request's intent and sender, except for the fact that nodes might try to cheat the protocol by running smaller and less capable AI models, instead of those they have subscribed to, in order to reduce their cost of running the AI inference. As mentioned, such scenarios include externally owned requests, as well as certain smart contract-authenticated request applications.

As previously observed, cross-validation sampling consensus is not intended for applications that provide smart contracts with additional AI inference features, as in that case the incentive to collude with the protocol might be orders of magnitude higher than the cost of running native AI inference (or it can even be unbounded). Therefore, Cross-Validation Sampling Consensus should not be used for the majority of smart contract request-based applications.

2.8 Node Obfuscation

In the Cross-Validation Sampling Consensus protocol, a request is settled in a two-round step. This is because the protocol waits until a node is sampled first in order to submit an initial output commitment, and only then samples, with probability p, additional nodes. One can optimize such a protocol through what we call *node obfuscation*, see [4]. Under node obfuscation, the protocol does not disclose which nodes have been selected to process a given request, nor the number of nodes selected. In this way,

a node on the network can verify if it was selected to process a given request, but does not learn any information about how many other nodes were selected to process the same request. Under this procedure, the game theoretical analysis for Cross-Validation Sampling Consensus still holds.

2.9 Non-determinism nature of AI inference

Our sampling consensus protocol (including cross-validation) relies on the assumption that the nodes can generate the same result by executing the same AI model inference on a request input. However, AI inference is not necessarily deterministic, for different reasons.

Next token sampling: Generative AI inference often times relies on next-token sampling, in which the AI model forward pass is used to generate a logits tensor of (log-)probabilities for the next token. Once we have the logits tensor available, the algorithm goes by sampling the next token among those with highest probability of being selected next. Sampling tokens in this way is highly nondeterministic. However, the source of non-determinism can be controlled through setting a fixed random seed (and other metadata input values such as temperature, topk, topp, etc). Once we fix the random seed, and the metadata inputs, one can show that sampling the next token from the same logits tensor, across different machines (having the same GPU card model) becomes fully deterministic.

Floating point non-determinism: Another fundamental source of nondeterminism comes from the fact that AI inference relies heavily on floating-point arithmetic and the latter is not transitive under addition. This means that if we have three floating point numbers $x,\ y,\ z$ then the equality

$$(x + y) + z = x + (y + z),$$

does not necessarily hold, in general. One way to mitigate this issue is to use fixed-point arithmetic through quantized weights and quantized inputs, instead of using floating-point arithmetic. The advantage of this approach is that fixed-point arithmetic is fully deterministic.

However, quantization often times leads to model performance degradation, more so if the inputs are also quantized, in addition to weights.

Another way to mitigate this issue is to run requests across nodes with the same hardware specifications, such as the same GPU card model. For example, Nvidia GPU cards are IEEE754 compliant, [23], and the same GPU cards handle floating point operations in similar orders (assuming that they execute deterministic CUDA kernels), see [18] for more details.

This ensures that AI inference becomes deterministic across different nodes, on the network. This observation can be mathematically proven through a thread volume analysis on the number of GPU cores, for a single card. Additionally, the Atoma team has run hundreds of thousands of tests internally on this issue, running different requests across multiple nodes (all with the same GPU cards) with the same random seed and other input metadata values. Through all of our tests, we have observed complete determinism in all generated outputs (in all machines). We will publicly release our results in the future.

Moreover, it is possible to adapt the work by Srivastava, Arora, Boneh [20] to our protocol, within the context of Cross-Validation Sampling Consensus. In this case, for an incoming request, the initial selected node runs AI model inference committing to a log file of all the floating point approximations, within a well-defined approximation threshold. Once other nodes are selected, these are requested to run the same AI inference, with the same input parameters, but following the first node approximations within the committed log file, for operations within the given approximation critical threshold. The same analysis should then be possible within [20] to prove that our protocol becomes fully deterministic, even with nodes performing AI inference relying on floating-point arithemtic.

2.10 Dispute

In the case where the sampled nodes do not agree on the state of a generated output, the on-chain logic contract presents a dispute. Disputes can be resolved in different ways. First, the on-chain logic is responsible for sampling a given number of nodes to resolve the dispute. In that case, the nodes must run the exact same AI inference on the given request. These nodes are required to submit an actual proof of the computation, either through a fraud proof in which a bisection algorithm is used to find the initial step of disagreement, which is a much smaller computation than running the whole AI inference process, between the initial selected nodes, and the segment of computation can be

verified using zk proofs. Another possibility is to run the whole AI inference within a single or multiple trusted execution environments (TEEs) providing TEE attestations. Once such a proof/attestation is generated, it can be verified by the on-chain logic contract, and the contract can resolve the dispute by slashing the malicious nodes and rewarding both the honest nodes of the participant, the disputed nodes selected, the Atoma community trust fund, and the user who initiated the request.

Since the deposited collateral is orders of magnitude higher than the actual native cost of running AI inference, even though that the dispute resolution might be orders of magnitude more expensive than the native cost of inference (if the dispute is to be resolved through a fraud proof), it is still economically feasible, as the collateral is then distributed among all parties involved. Moreover, in the case of a dispute, the user might right away submit a new request, with the same contents, with the guarantee that it will be rewarded monetarily for the initial failed request.

2.11 AI model whitelisting and governance

At launch, the network will support the main open source AI models available, such as Llama3-8B, LLama3-70B, Qwen2, Phi3-mini, Stable Diffusion, etc. That said, it is crucial that the protocol allows for new, more capable, AI models to be added to the network, over time. Being committed to the standards of open-source AI, we plan to build incentive systems to reward developers for creating and deploying new AI models into the network. This leads to the following question:

Question 2.6. *How can we ensure that the AI model deployed does not lead to potential malicious behavior?*

This is a very hard task to solve using formal methods alone. General generative AI models, such as Llama, GPT, Claude, Qwen2, etc., have been trained on enormous corpus of text data. The training data set might contain chunks of data that explicitly contribute to the model to be able to act maliciously on behalf of some entity.

This means that it is very difficult and expensive to assess the quality of the training data and its potential to lead to malicious behavior on certain tasks. Moreover, even if this is attainable, general-purpose AI models can be prompt engineered to produce some undesired answer. For these reasons and because it is generally better for a protocol to minimize trust assumptions, we prefer to leave the decision about deploying new AI models to protocol developers, communities, and users.

In order to register a new AI model into the network, any wallet (be it externally owned or a contract account) can submit a transaction into the on-chain logic contract whose goal is to register a new AI model architecture and weights. In order to do so, it must specify a unique AI model id and a model weight hash identifier (generated as a Merkle tree root hash of all the model weights). Once this registration is submitted, the Atoma nodes can subscribe to that AI model.

If a node subscribes to a given AI model, it must download its weights through an external API service (such as Hugging Face) or from a decentralized storage (such as IPFS). Through Atoma's own UI interface, it is possible to have access to the list of all the models being orchestrated onto the network, including additional metadata that allows nodes to fetch the model weights. Once the weights are downloaded, the node must ensure that the Merkle tree root hash of the model weights is the same as the one specified in the registration transaction. If this is not the case, the node will submit a failed subscription transaction to the on-chain logic contract. If enough of such failed subscription requests are submitted, the AI model id will be removed from the internal registry of the on-chain logic contract.

Once there are enough nodes subscribed to that model, the AI model becomes available to be used by protocols and Atoma users.

Importantly, the on-chain logic contract can be used to accrue fees for AI model creators. The AI model creators should *own* both the model weights and the wallet that initiated the model registration transaction. That wallet can be entitled to accrue fees for every request that is executed through the specified AI model. Additionally, model owners will also be entitled to accrue a portion of the token emission rate. In order to avoid possible model weight leaks and theft, we will implement a mechanism to ensure that the AI model creator can be properly identified as such. Sybil resistance algorithms are possible for this purpose, following an approach similar to the zkLogin protocol [3] or zkTLS.

Moreover, to avoid stealing model weights, we will require that a model owner deposit some collateral with the model registration transaction. Moreover, we will have a community of experts to validate, referred to as *model validators*, the originality of a new AI model being registered in the

network. Thus, mitigating possible risks of stealing model weights from an already existing model repository on Hugging Face or other APIs. Such model validators will be rewarded with fees for their work in validating new models on the network. Model validators will also be subject to rigorous scrutiny by depositing some collateral that can be slashed if the validators try to act maliciously by validating copied models.

3 Verifiable AI

In this section, we will explore a few different methods for verifiable AI inference and compare these with our Sampling Consensus protocol in terms of performance, cost, and security guarantees. We are mostly interested in a formal comparison between zkML and opML techniques, for verifiable AI.

3.1 zkML

The field of zero-knowledge Machine Learning (zkML) has emerged after zero-knowledge proofs have gained a prominent role in the cryptocurrency space, especially for scaling blockchains. To put it simply, zkML is a set of techniques based on zero-knowledge cryptography to provide cryptographic proofs attesting to the integrity of any Machine Learning model inference computation. In recent years, some promising research has been published on the topic, such as TensorPlonk [11], trusted DNN [12], zkCNN [15], Modulus Remainder [17] protocol based on the GKR protocol, SpaGKR [14], zkLLM [21], to name a few.

Especially relevant to our discussion of verifiable LLM inference is the work on zkLLM [21], which, at the time of writing, is considered the state-of-the-art zkML protocol that attempts to provide verifiable LLM inference. This work leveraged GPU acceleration, relying on the CUDA library [9] for BLS12-381 curve operations on Nvidia GPUs. Even though an astonishing 15-minute proof generation time for a single 13 billion-parameter LLM inference forward pass is achieved, it is important to notice that such a forward pass allows us to generate the next token in the sequence. If we consider an output sequence of 100 tokens in total (which is considered a short answer, in practice), then aggregating together all the 100 forward passes zk proofs would require at least 1500 minutes (that is, 25 hours), in total, to generate a zk proof for entire LLM inference. That said, we should expect a considerably higher total proving time, as the attention Key and Value (KV) caches grow quadratically on both the number of prompt and generated tokens. Given that LLM inference on GPUs is bounded by GPU memory and not compute, we expect that quadratic growing KV caches will lead to much faster proof generation times in practice.

Moreover, zkLLM proving times were achieved using a setup of 124.5 GB of memory, 12 CPU cores of an AMD EPYC 7413 (2.65 GHz with 128M cache L3), and an NVIDIA A100SMX4 GPU with 40GB of memory, which is a setup orders of magnitude more powerful than the actual hardware requirements to run native LLM inference for models up to 13 billion parameters.

For larger models, such as 70 billion parameter Llama3, we are still yet to see any research work to provide zk proofs for such models inference. With the advent of larger LLMs, such as the 400 billion model parameter Llama3, zkML might not even be practical.

In addition to our analysis, the LLM inference does not pertain solely to proving multiple forward passes. Often times, LLM inference uses techniques such as next-token sampling, out of a list of highly likely logits, to actually generate the next token. This is especially relevant if one needs to ensure some creativity regarding the LLM outputs. If the full LLM inference is to be proved, it is necessary to provide zk proofs for the sampling process as well. Each of these steps need to be taken into account if one wishes to have a fully zk proving system tailored to LLM inference, adding to its complexity and cost.

It is important to note that zero-knowledge proofs provide a very high level of security guarantees for LLM inference. Combined with the fact that zkML will continue to be a few times more expensive than native LLM inference, different verifiable requirement situations will not benefit from a more elastic verifiability strategy, contrary to techniques offering Nash equilibrium in the long run with considerably lower costs, such as our Cross-Validation Sampling Consensus protocol.

Although significant technical challenges remain to be addressed to make zkML practical, the results described above are highly promising. It is crucial to consider various possible scenarios to improve zk proof generation for LLM inference. In particular, we are interested in zk-STARK-based

proving systems that require smaller bit-field sizes, such as Circle Stark over a 31-bit Mersenne field or Binius (see [10] and [8], respectively). We anticipate significant proving speedups through the future production of zk-specific acceleration hardware. Furthermore, zkML provides the most secure method for verifiable AI and has the advantage of allowing the model weights to be kept private by the nodes. However, zkML does not offer a privacy-preserving model for user input and output data.

Given that zkML is still in its infancy and has already seen incredible improvements in the past few years, we are looking very closely at the developments in the field.

3.2 opML

Contrary to the heavy cryptographic reliance of zkML, optimistic Machine Learning (opML) adopts a fundamentally different strategy based on dispute resolution mechanisms. The optimistic approach presupposes that participants will act honestly given the economic disincentives for fraudulent behavior. In the rare event of disputes, opML provides mechanisms for challenging and resolving fraudulent claims, ideally without necessitating a heavy computational verification for every transaction. However, reliance on economic incentives and dispute resolution may introduce vulnerabilities in network security.

In particular, for each AI inference request, a challenge period is required in order for verifier nodes to potentially challenge the validity of the original AI inference output. This challenge period is a fixed period of time that might correspond to a few hours to a few days. This characteristic of the protocol has the disadvantage of introducing a delay, or increase in latency, for settling requests on the network. Moreover, verifier nodes are typically paid if they put forth a dispute and the resolution favors them. Therefore, for expensive computations such as AI inference, it becomes less economically viable to incentivize verifiers to continuously attest to the validity of generated outputs through AI inference. The latter might invariably lead to serious security vulnerabilities in the network.

3.3 Atoma's Sampling Consensus

The Atoma Sampling Consensus (both in its simpler form and its Cross-Validation variant) provides verifiable AI inference, with the advantage of allowing for elastic verifiability. At a single node level, Atoma's Sampling Consensus requires only the node to run native AI inference, without any additional overhead compared to zkML. Contrary to opML, our Sampling Consensus protocol does not require waiting for any challenge periods in order to reach settlement; instead, settlement is reached once the slowest node requested to process a request commits to the output state (on the happy path of no disputes).

Even though our simple form of Sampling Consensus can lead to high verifiability guarantees, at the expense of higher replication costs, our cross-validation sampling consensus protocol requires minimal replication, allowing for close to native cost of running verifiable AI inference, without challenge periods, and better security guarantees than opML.

3.4 Comparison between zkML and Sampling Consensus

In this section, we will compare our Sampling Consensus protocol with both the zkML and opML methods.

Compared to zkML, our Sampling Consensus protocol requires considerably less computational resources, overall, with a much lower cost and overhead time. Moreover, the Sampling Consensus allows the node on the network to run native AI inference, with no additional hardware requirements, other than those for native AI inference. This implies that nodes on the network can implement the latest optimizations around LLM inference, etc., without having to recur to extensive research and development efforts to integrate the latest state of the art in AI inference.

Another advantage is that our Sampling Consensus protocol allows for elastic verifiability. However, if an application requires very high levels of verifiability, the user and/or protocol might require multiple nodes to be sampled in order to reach consensus on the output state. Similarly, applications requiring low levels of verifiability might prefer to select a very small number of nodes to be sampled to reduce the cost of running the AI inference. Moreover, using the Cross-Validation Sampling Consensus, the majority of decentralized AI applications will be able to process requests with long-term verifiability guarantees, with minimal replication costs.

A drawback of our Sampling Consensus protocol faces compared to zkML is the fact that Sampling Consensus does not provide full privacy for model weights (indeed, a wide number of nodes on the network need to be able to access the model weights in order to run LLM inference on a given model). This can be widely mitigated by using TEEs, which can then enable full privacy for model weights and user data privacy. We believe that the use of TEEs combined with our Sampling Consensus protocol will lead to both full privacy and very high verifiability (even with Cross-Validation Sampling Consensus). We will explore these synergies in a future paper.

3.5 Comparison between opML and Sampling Consensus

Contrary to zkML, opML claims low computational overhead, with the caveat that opML may incur higher overhead during disputes. The zkML approach results in high computational overhead due to cryptographic processes.

Sampling consensus also requires low computational overhead. Moreover, even when multiple nodes are required to process a request (both for simple and Cross-Validation Sampling Consensus) the overhead is very low. This is because multi-node runs only require the on-chain logic to compare a few hashes.

We believe that Sampling Consensus is a superior design compared to opML due to the fact that the former does not incur high latency times for settling outputs on the network and because each node involved in processing a request is entitled to accrue a fee for its work. Incentivizing nodes to always be responsive and honest.

Moreover, following the ideas in [16] for Rollup validators, we can derive a Nash equilibrium for opML; our approach reflects a lower probability of undetected fraud lower than that of opML, assuming that a malicious actor can control at most 10% of the total compute power of the network (more details will be provided in Appendix A).

3.6 Trusted Execution Environments

We are exploring the use of TEEs for verifiable AI inference. Large AI models often require specialized hardware, such as GPUs, for computational acceleration. Therefore, to run these larger models inside secure enclaves, such specialized hardware is required to have available TEEs. As of now, only Nvidia's Hopper and Blackwell architectures have built-in TEEs available. Unfortunately, these GPUs are currently very expensive, and we cannot expect the network to aggregate enough of these GPUs immediately. That said, we believe that with our robust tokenomics, aggregating such high-tier GPUs on the network will be possible in the future.

We are confident that Trusted Execution Environments (TEEs) offer the most cost-effective and reliable method for verifiable AI inference. In addition, when combined with our randomization mechanism for node selection, we can mitigate potential risks associated with the use of TEEs on the Atoma platform. In the future, we plan to publish a scientific paper dedicated to exploring the potential of TEEs for Atoma.

4 Atoma Node Optimizations

To ensure that the network provides an optimal user experience for end users and protocols, while allowing nodes to efficiently utilize their GPU resources, we have implemented several AI inference optimizations, particularly focusing on LLM inference. These enhancements will significantly improve overall network performance.

4.1 Flash and Paged Attention

To enhance the performance of the Atoma nodes, we have integrated a highly efficient LLM attention algorithm inspired by classical virtual memory and paging techniques in operating systems, namely PagedAttention, following the seminal work in [13], together with FlashAttention, see [7] and [6]. PagedAttention allows for the highly efficient memory management of LLM attention layers during inference. Through PagedAttention it is possible to continuously batch considerably more requests, altogether. This method significantly reduces latency and improves network performance, providing a

seamless experience for end users and protocols alike. Moreover, PagedAttention allows the network to vertically scale with better and more powerful hardware becoming available on the network. As a result, nodes can handle a higher volume of requests without compromising on speed or accuracy, providing a robust and scalable solution for AI inference and accruing more fees, by best utilizing their GPU resources.

Through PagedAttention, nodes can dynamically allocate GPU resources based on current network demand, adjusting batch sizes to optimize performance. This flexibility ensures that the network can adapt to varying workloads, maintaining high levels of efficiency even during peak usage times. Consequently, Atoma is able to deliver consistently high performance, meeting the needs of both end users and protocols while ensuring optimal GPU utilization.

4.2 Quantization

To optimize the performance of the network, we have integrated several quantization techniques that significantly improve memory management and computational efficiency. Quantization reduces the precision of model parameters, which reduces the memory and computational resources required for AI model inference. The primary quantization methods used include uniform quantization, dynamic quantization, and mixed-precision quantization.

Uniform quantization involves reducing the precision of the weights and activations of neural networks to a fixed lower bit-width, such as converting 32-bit floating-point numbers to 8-bit integers. This method reduces the memory footprint of AI models, leading to more efficient memory utilization.

Dynamic quantization adjusts the precision of the model parameters at runtime based on the distribution of input data. This technique is particularly advantageous for inference tasks in which data characteristics can vary. By dynamically quantizing parameters, we optimize the balance between model size and accuracy in real time.

Mixed-precision quantization employs different levels of precision within the same model, such as using 16-bit floating-point numbers for critical layers and 8-bit integers for less sensitive layers. This selective approach reduces memory usage without significantly affecting the accuracy of the model.

These techniques allow the network to maintain high model accuracy while minimizing resource consumption, especially when combined with the PagedAttention algorithm.

4.3 Fully Sharded Data Parallelism

Fully Sharded Data Parallelism (FSDP) is a pivotal strategy utilized by Atoma to optimize the efficiency of LLM inference. By implementing FSDP, nodes hosting multiple GPU cards are allowed to partition the LLM's parameters across these devices in order to process data in parallel. Each GPU or device manages a shard of the model's parameters, enabling simultaneous computation on different segments of input data. This approach not only accelerates the inference speed by distributing the computational workload but also minimizes the communication overhead between devices, as data exchanges are limited to synchronization points between shards rather than continuous updates across the entire model.

Using FSDP, the network nodes can handle larger batches of data and accommodate more concurrent inference requests without compromising performance. Furthermore, FSDP optimizes memory utilization by ensuring that each GPU or device only holds the necessary parameters for its assigned shard, thus maximizing computational efficiency and reducing memory constraints. This is especially relevant when combined with both the FlashAttention and PagedAttention algorithms.

4.4 Key Value Caching

We are currently exploring how PagedAttention could be used to cache already processed key-value (KV) blocks. Through KV caching, the nodes could share an agreed state of already processed KV blocks. If retrieving such KV blocks from the network is faster than actually computing them, the network can become considerably more performant by processing more requests altogether (thus computing more KV cache blocks and storing these for later retrieval).

5 Data Authentication and Data Management

In order for Atoma to provide a more fine-grained user experience, nodes must be able to run inference on authenticated data for context prompting. Moreover, data produced through Atoma's nodes should be stored on behalf of a wallet or protocol. In this section, we will explore how we envision handling data authentication and data management.

5.1 Data Authentication

Imagine an application that provides a chat interface on top of Atoma. Such a chat application might benefit from scraping the web in order to retrieve data that can be used to improve its overall UX. For example, if a general purpose AI model has been trained on data up to a certain date, the application might fetch news data regarding current events to better answer a user request.

In this context, maintaining the integrity of the data source is critically important for Atoma, in order to align with the network's verifiability requirements. Therefore, nodes in the network must access only authenticated data, data whose origin can be verified through a trusted API. For example, such an API could be a publicly reputable news website. This authentication is facilitated by protocols such as zkTLS.

Consider a smart contract designed to perform on-chain operations such as balancing liquidity pools within a protocol or modifying a user-owned NFT using generative AI. The criteria for these actions often depend on historical on-chain data. For instance, in the case of balancing liquidity pools, one might be interested in the average token ratio within the highest volume pools in the protocol. Alternatively, in the NFT example, the modification of the NFT might depend on the user's historical trading activity: if a user has significant activity, the NFT could receive rare additional features. In both scenarios, it is crucial to verify the historical on-chain data as part of the blockchain. With the advent of zk coprocessors, it is now possible to leverage the functionality of these protocols to provide verifiable on-chain data to Atoma's nodes.

In both instances mentioned above, we have demonstrated how zero-knowledge cryptography can be used to authenticate the origins of the data that are being retrieved for the generation of augmented AI. We intend to implement a general framework that enables Atoma nodes to access data from various sources, such as trusted public APIs, blockchain on-chain data, personal user data, etc.

In addition, we are investigating the use of Trusted Execution Environments (TEEs) for similar applications. The primary benefit of TEEs is their ability to access encrypted sensitive data for AI generation purposes, producing outputs without the node gaining full access to the data. TEEs are especially relevant for externally owned based applications (such as the chat application above). In future work, we will offer a comprehensive overview of how TEEs can be integrated into the network, along with a detailed security analysis.

Whenever additional data is retrieved externally to augment the generation of a request's output, we will require that this data be accompanied by either a zk-proof or a TEE attestation. This will allow for the validation of the integrity of the data's source.

5.2 Data management

Atoma nodes are not designed to store the data generated through the network. Instead, data storage should be delegated to third-party services, centralized or fully decentralized. When a node processes a request, it is responsible for sending the generated output to a specified data storage destination, as indicated in the request.

If the node is processing an externally owned request, through some centralized UI (think of the chat application), the node should share the output generated with the external API of that application, or a decentralized storage such as IPFS, on behalf of the user. Decentralized applications should require the node to submit user's data on a decentralized applications (e.g. IPFS).

Conversely, smart contract requests may necessitate the node to provide the actual output on-chain, provided that it is not excessively large. For example, this could be applicable to a protocol requesting a token price prediction, in order to re-parametrize some protocol value (such as its token inflation emission ratio).

Moreover, in both cases, where the output is shared with an external public API or a decentralized storage, it is possible to have proofs of storage that can attest for the honest behavior of the node.

6 Atoma's Application Layer

As mentioned above, the Atoma Network supports the development of new applications, which may vary in terms of decentralization requirements, complexity, scope, and computational demands. In this section, we will provide a concise overview of a few of these applications and our plans to encourage builders and developers to leverage Atoma's capabilities for cost-effective verifiable AI inference.

6.1 Externally Owned Request Based Applications

Externally owned request-based applications refer to any application that receives requests directly from a wallet. These applications typically provide some form of user interface, either as a desktop or mobile app, and often rely on a centralized backend for data storage and user authentication. Builders will benefit from Atoma's fair market and thus be able to provide competitive pricing.

It is essential to recognize that such applications can exist at the intersection of AI and Web3, even if they depend on a centralized infrastructure. In the following, we will explore a few examples.

Personal AIs: Every time we interact with an application, we generate personal data that often translates personal interests, beliefs, etc. Over time, this data becomes increasingly valuable for personal use, as it becomes a comprehensive record of various facets of our lives. Such valuable data should be owned by the individual, not by large corporations. When parsed, curated, and enhanced via AI, this data's worth increases. AIs can create highly fine-grained embedded representations of the data that can be employed for retrieval-augmented generation (RAG) or to fine-tune AI models to individual data sets. These advancements pave the way for *actionable personal data*, allowing users to continuously generate and utilize data to articulate their intentions accurately.

Through personal AIs, our web interactions will transform dramatically. Instead of using traditional UI-driven methods to initiate predefined processes, we will specify our intents to perform desired actions directly (e.g. streaming a newly released movie or compiling a blog post from vacation photos and videos). Personal AIs can also communicate with each other, inquiring about other users' preferences. This becomes particularly significant when integrated with reputation systems enhanced by digital ID solutions (such as World ID by WorldCoin). In this way, users can define permission-based access to certain aspects of their digital footprints through their personal AIs. For example, a user could allow friends within a social circle to query their personal AI for real-time location or emotional state.

In order to accomplish this vision, one needs broad access to cheap, fast, and private AI inference on a continuous stream of generated data. This is possible through the Atoma Network. Moreover, the network can produce high amounts of personal curated data in different formats (say, vector embeddings, knowledge graphs, etc.), which can be stored on a decentralized storage protocol, for future usage. Moreover, the Atoma Network will be the natural framework to deploy user-personal fine-tuned AI models.

Knowledge bases as public goods: Chat interfaces, such as ChatGPT, are among the most widely used applications today. Typically, these applications offer a user interface that allows users to interact reliably with a LLM to get answers to their questions. Specifically, these chat applications have been employed for tasks such as writing code, analyzing research articles, providing ideas for dinner recipes, and suggesting travel destinations, etc. However, these chat interfaces do not support common public sessions, where multiple users can access and interact with the LLM within the same chat session.

We propose an application that provides a shared chat session with LLMs, which could be very valuable for the Web3 community. By allowing multiple participants in a protocol or DAO to join these chat sessions, the application could offer insights into protocol/DAO governance and generate informative content from user interactions. This would lead to the collection of actionable knowledge on protocol governance.

Moreover, such an application could leverage some authentication service, for example through Ethereum Named Service (ENS), that could be used to establish a reputation system for wallets that interact with the protocol, based on the relevancy of each user insights into the protocol.

For example, a community discussing the upcoming advancements in zero-knowledge cryptography could bring together leading experts in the field. This would facilitate a comprehensive conversation

on specific optimizations in zero-knowledge technology and explore potential new applications for technology in emerging protocols.

High-reputation contributors could potentially receive financial rewards for their significant contributions to the advancement of zero-knowledge cryptography.

Additionally, this application could utilize vector databases to store vector embeddings of all collected knowledge, allowing enhanced retrieval for future queries. Furthermore, high-reputation individuals could rank the types of information processed by the application, facilitating the curation of public data. The data storage layer can be further decentralized, providing a commonly owned protocol knowledge base.

Discussion forums as public goods: LLMs excel at converting unstructured data (such as text) into structured formats (such as JSON). This capability enables them to efficiently extract insights from extensive texts on a specific topic and organize these insights into well-defined structures, such as knowledge graphs, which can be stored and subsequently utilized for knowledge retrieval.

In the contemporary digital landscape, many important decisions on various topics are made publicly through online platforms, think of Linux forums, DAO governance protocols, etc. However, this valuable content often remains unstructured and dispersed on the Web, making it difficult for new users to effectively retrieve and access it. This issue is particularly significant, considering that much of this content contains insightful information on a wide array of topics that are likely to remain hidden and underutilized in the future.

An application can leverage LLMs to parse discussion forums and extract the most relevant information in a structured format. This data can then be stored within vector databases, knowledge graph databases, etc., allowing easy retrieval for future use.

Platforms like Wikipedia could benefit greatly from this kind of technology. Moreover, Wikipedia's knowledge is stored in a Knowledge Graph format but remains static, requiring manual updates over time.

Such an application could result in a Wikipedia 2.0 platform that can be applied to any content on the Web, including protocol-specific data. Similarly, website content can be parsed in this format using an LLM. Compiling all this data into a shareable knowledge format could lead to next-generation search services that offer a more personalized user experience, in general.

It is important to note that even with an agreed storage format, there are multiple ways the content can be parsed by the LLM. Verifiability plays a critical role in this use case, as it ensures high-trust guarantees that the required LLM was used correctly to parse the relevant content into an output format, avoiding any potential misuage of the protocol.

Data scraping: Building upon the previous example, one can think of applications with the purpose of data scraping at scale. By leveraging cost-effective and reliable AI inference, it is possible to scrape a wide variety of specific data, including information on financial markets, political news, specialized research topics, weather data, and more. These advanced scraping techniques can be utilized by companies, protocols, or individuals to gain a more comprehensive understanding of various topics.

Intent parsing applications: In recent years, the concept of 'intent' has gained significant importance within the Web3 ecosystem. Leveraging their advanced natural language processing capabilities, LLMs can effectively parse and structure users' declarative intents.

For example, a user might write in text its intent of performing a trade in a specific Web3 protocol. The LLM can in turn parse this into a JSON format that is compatible with the transaction formatting for submission. A wallet service can benefit largely from this sort of functionality, allowing a much better user experience. This is especially relevant for users with little Web3 experience, allowing wallet services to help with onboarding new users.

Once the LLM parses the intent content and builds the exact transaction body, the wallet can display the actual transaction through a UI back to the user. In this way, a user can manually confirm the accuracy of the built transaction and thus be sure that the submitted transaction is submitted correctly.

AI market places: The Atoma Network will provide a rich environment for AI model usage. This in turn can be used to establish a common market place to access the best AI models or AI applications.

6.2 Smart Contract Request Based Applications

Smart contract request-based applications refer to applications that enable smart contracts to utilize Atoma's intelligence layer directly. These smart contracts can make external calls to Atoma's on-chain logic, requesting a specific large language model (LLM) to process given calldata. Integrity in the generated output is crucial for these applications. Below, we will briefly discuss some potential applications.

Automatized Yield strategies: Yield rewards are a fundamental innovation in Decentralized Finance (DeFi). However, existing strategies fall short of the sophistication employed in Traditional Finance (TradFi), especially in utilizing AI and extensive data for real-time market interactions.

By optimizing investment actions such as buying, holding and selling, AI can automate these processes to achieve better returns without the need for constant human oversight. Furthermore, AI models designed specifically for analyzing on-chain yield data can more efficiently collect and distribute yield returns among participants, thus enhancing individual profitability and contributing to the long-term sustainability of the DeFi ecosystem.

Such on-chain protocols require verifiable AI to maintain transparency and efficiently adapt to market volatility conditions.

On-chain index funds: Traditional index funds offer diversified market exposure, but often involve costly fund managers. In contrast, AI-powered crypto index funds automatically reallocate funds based on market conditions, eliminating the need for centralized management. These AI-managed funds can optimize their market exposure in real time. This democratizes sophisticated investment strategies, making them more accessible to a broader audience and representing a significant shift in the financial landscape, especially in the rapidly evolving world of cryptocurrency.

Self-evolving or Genetic NFTs: On-chain smart contracts can leverage the Atoma Network's capabilities to autonomously generate new NFTs based on predefined characteristics. Furthermore, any NFT minted on the Atoma network has the advantage of being traceable to its AI origin, providing a verifiable proof of AI generation.

NFTs generated by AI could adapt and evolve based on the activities of the chain owner or other relevant metrics. For example, a character NFT could acquire new attributes if its owner achieves a certain credit score within a specific borrowing and lending protocol, such as Aave. In addition, entire communities, such as DAOs, could collaboratively create NFTs that reflect their community preferences. A DAO might decide to produce virtual characters for a game, with characteristics influenced by discussions of its members in an internal forum. An LLM could analyze these discussions, employing a world model to determine the optimal set of characteristics that aligns with the collective desires of the DAO members for the character characteristics.

6.3 Smart Contract Authenticated Request Based Applications

Smart contract authenticated request-based applications are applications that utilize smart contract authentication to validate actions generated by artificial intelligence (AI). This means that while large language models (LLMs) can generate actions to complete tasks, these actions must be first validated by a smart contract before they can be processed.

Such applications are especially relevant in the context of Web3 automation. Smart contract authentication and validation are required if one wishes to build the next generation of smart agents in Web3:

Smart agents: Smart agents are based on advanced autonomous agent technology, that can be owned either by one or multiple wallet addresses. Each service is made up of one or more software agents, each operated by an agent operator. While these agents can be automatized through AI

inference, on top of the Atoma Network, and can retrieve data from anywhere, they can also be represented by an on-chain smart account (following Ethereum's abstract account terminology).

Through smart accounts, smart agents can hold assets, pay for transactions, and interact with on-chain apps. Moreover, bridging technologies allow smart agents to co-own multiple assets on different blockchains.

Multisig functionalities will also enable smart agents to be co-owned by multiple externally owned accounts (EOAs), allowing for DAO-owned smart agents. These smart agents can be utilized to automate various processes of on-chain governance, enhancing efficiency and security.

Smart accounts can also provide safety conditions for the outcomes of agentic AI actions. For example, in AI-powered funds, safety logic may be necessary to limit the actions of agents. This ensures that agents can only adjust the fund's allocations, thereby protecting the assets through on-chain mechanisms. Thus, even if agents execute malicious code, they are restricted to performing only specific transactions.

Smart agents can be used in a broad range of Web3 applications, including autonomous trading, prediction markets, AI powered DAO-delegation, etc.

World models: A *world model* is an AI system that builds an internal representation of an environment and uses it to simulate future events within that environment. World models are widely used in reinforcement learning, robotics, and automation, among others. With the current advent of Large Language Models for text generation, we have come to witness new promising venues of research around world models. In the Voyager paper by Nvidia researchers, see [22], the authors showed that by combining several other tools, including LLM text generation, prompt engineering, vector search, compilers, and code bases, it is possible to create a state-of-the-art AI system that surpasses any previous system in the Minecraft game. More intriguingly, this can be achieved without relying on conventional agent optimization techniques. Such an approach is promising for building the next generation world models that ultimately rely on LLM inference.

These ideas can be adapted to the context of Web3, where smart accounts can be integrated with user interfaces to facilitate specific blockchain actions that require additional smart contract authentication. For instance, imagine world models being utilized in gaming; these models could claim certain items on-chain when they achieve new levels, which would trigger the need for smart contract authentication.

Tokenization of real world assets: One of the most promising applications of blockchain technology is the tokenization of real-world assets. This innovation has the potential to revolutionize asset transfer management by automating processes and creating more resilient financial products anchored in tangible assets. However, achieving this vision hinges on the availability of a decentralized layer of verifiable AI computation, which remains a significant challenge. This technology is crucial for decision making and analyzing real-world data.

Furthermore, recent advances in hardware, such as smart glasses and smartphones, open new possibilities for scanning and interacting with the physical world. With the development of specific hardware for scanning real-world items, one should be able to generate proofs attesting that an item was correctly scanned (either through device TEEs or zk proofs for image recognition AI models). These proofs verify the correct identification of assets through AI analysis, facilitating the creation of an intermediate representation layer that bridges the digital world (potentially embodied as a metaverse) with the physical world. Smart accounts can be used to mint these real world assets on existing blockchains, by further verifying the required submitted proofs.

6.4 Atoma's Incentives for Builders

We plan to use Atoma's treasury fund to incentivize developers to bring to life some of the exciting applications mentioned earlier. By allocating these funds, we aim to help teams gather the necessary resources to build this infrastructure on top of the network, and foster demand for Atoma's compute resources.

We will present a document outlining best practices for building on top of Atoma. This includes guidelines for prompt engineering to prevent prompt injection attacks and to ensure that there is no

redundancy in the tasks assigned to a Large Language Model (LLM). In addition, we will discuss best practices to prevent potential leaks of sensitive information that could cause financial losses.

We are committed to promoting open source applications and highly value any projects developed according to open source standards.

A Appendix

In this section, we will provide a precise mathematical formulation of Nash equilibrium results for our Cross-Validation Sampling Consensus algorithm. The present statements were first published by the Hyperbolic team in [24].

In the cross-validation sampling consensus scenario, we select a node at random to run a request on the network. We also select $N > 0$ nodes to attest to the integrity of the response of the first node, with probability $0 < p < 1$. Let us introduce the following notation:

- Let R denote the reward paid to a node for running AI inference, in the case the node is honest.
- Denote by S the amount of collateral a node deposits on the on-chain logic contract to participate in the protocol.
- Denote by $0 < r < 1$ the maximum rate of control a node can have on the network.
- Let C denote the native cost of running AI inference.
- We denote by W_1 the maximum reward that a node can accrue by processing a request dishonestly, and no other node is selected to attest its generated output.
- We denote by W_2 the maximum reward that a node can accrue by processing a request dishonestly, and it colludes with all selected nodes to attest its generated output.

We follow the same set of assumptions as in [24].

Assumption 1. *Two distinct independent nodes cannot collude with each other unless they are both honest.*

Assumption 2. *We assume that obtaining a reward for running AI inference, $R - C$, is better for a node than being slashed, that is, risking losing S capital.*

Assumption 3. *We assume that the fraction of nodes in the network that a participant controls is at most r. Therefore, only a fraction of the r nodes are able to collude with each other, at each time.*

Under the above assumptions, the following results hold:

Lemma A.1. *If a network participant is honest, then the expected reward can be expressed as*

$$\mathrm{EH}_R = (1-p) \times (R-C) + \sum_{i=0}^{N} \binom{N}{i} \times r^i \times \big(1-r\big)^{N-i} \times \big(i \cdot \frac{R}{N} + R - C\big).$$

Proof. The expected value is averaged in the case in which the node runs inference and no other nodes are selected to attest to it, altogether with the case in which more nodes are selected. In the first case, the reward can be computed simply as

$$(1-p) \times (R-C),$$

as no other nodes are selected. In the scenario where $N > 0$ nodes are selected, the expected reward that is accrued by the participant must include the scenario in which a few nodes under its control are sampled (that is, these collude with the initial selected node). In that case, the colluding nodes do not need to recompute the outcome once more, instead they can just utilize the participant initial output. In this scenario, it follows that the expectation reward can be computed as an expectation over a binomial distribution, as follows:

$$\sum_{i=0}^{N} \binom{N}{i} \times r^i \times \big(1-r\big)^{N-i} \times \big(i \cdot \frac{R}{N} + R - C\big),$$

The result follows. □

Lemma A.2. *If a network participant is dishonest, then the expected reward can be computed as:*

$$\mathrm{ED}_R = (1-p)\cdot W_1 + p\cdot r^N \cdot W_2 + p\cdot \sum_{i=0}^{N} \binom{N}{i} \cdot r^i \cdot (1-r)^{N-1} \cdot \big(i\cdot \frac{R}{N} - C - S\big)$$

Proof. The scenario in which the participant acts dishonestly and no other nodes are selected to attest to the integrity of the computation allows the node to accumulate a reward

$$(1-p)\times W_1,$$

on average. In the case where the participant acts dishonestly and $N > 0$ nodes are selected to confirm the integrity of its computation, but all these nodes collide with the participant, the expected accrued value is $[\ \mathrm{p}\ \times rN \times W_2. In the case where more than one independent node is selected and the 0 \leq i < N$ nodes collide with the participant, the best strategy for the participant is to force the i nodes under his control to act honestly. This leads to an expected reward of

$$\sum_{i=0}^{N} \binom{N}{i} \times r^i \times (1-r)^{N-1} \times \big(i\cdot \frac{R}{N} - C - S\big).$$

The proof follows. □

From Lemmas A.1 and A.2. one deduces that the network can reach a Nash equilibrium state, in which rational nodes act honestly whenever the inequality holds:

$$\mathrm{EH}_R > \mathrm{ED}_R.$$

By rearranging terms in the above inequality we deduce:

Theorem A.3. *According to Assumption 1 and Assumption 2, a network participant has a winning strategy to act honestly if and only if the following inequality holds*

$$R + p\times S - (1-p)\times W_1 - C + (1 - r^n - (1-r)^N)\cdot p\cdot C > p\times r^n \times (W_2 + S - R).$$

Remark 1. *Using the inequality $1 - r^n - (1-r)^n \geq 0$, we immediately deduce from Theorem A.3 that if the inequality holds*

$$R + p\times S - (1-p)\times W_1 - C > p\times r^n \times (W_2 + S - R),$$

then a node has a winning strategy to act honestly.

Corollary 1. *Assume that the following inequality holds:*

$$p > \frac{C}{(1-r)\cdot (R+S)},$$

then the Cross-Validation consensus protocol reaches a unique Nash equilibrium, where the nodes behave honestly.

Proof. The proof of the corollary follows from the setting $N = 1$, $W_1 = R$, $W_2 = 2\cdot R$. □

From Corollary 1. we can deduce how large the *replication rate*, p, must be for the system to reach the Nash equilibrium in which every participant is honest, depending on the values of r, R, C and S.

For example, if we allow $R = 1.2\cdot C$, $r = \frac{1}{2}$ and $S = 250\times C$, we obtain

$$p \approx 0.00796,$$

That is, the system reaches the Nash equilibrium, in which all participants are honest, with a replication rate of approximately 0.8 %.

References

[1] Bitcoin energy consumption index.

[2] Bitcoin mining. https://www.investopedia.com/terms/b/bitcoin-mining.asp. Accessed: [Insert date accessed].

[3] Foteini Baldimtsi, Konstantinos Kryptos Chalkias, Yan Ji, Jonas Lindstrøm, Deepak Maram, Ben Riva, Arnab Roy, Mahdi Sedaghat, and Joy Wang. zklogin: Privacy-preserving blockchain authentication with existing credentials, 2024.

[4] Dan Boneh, Saba Eskandarian, Lucjan Hanzlik, and Nicola Greco. Single secret leader election. Cryptology ePrint Archive, Paper 2020/025, 2020. https://eprint.iacr.org/2020/025.

[5] CNBC. How to drive bias out of ai without making mistakes of google gemini. https://www.cnbc.com/2024/03/27/how-to-drive-bias-out-of-ai-without-making-mistakes-of-google-gemini.html, 2024. Accessed: 2024-07-01.

[6] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning, 2023.

[7] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022.

[8] Benjamin E. Diamond and Jim Posen. Succinct arguments over towers of binary fields. Cryptology ePrint Archive, Paper 2023/1784, 2023. https://eprint.iacr.org/2023/1784.

[9] Filecoin Project. ec-gpu repository. https://github.com/filecoin-project/ec-gpu. Accessed: 2024-07-01.

[10] Ulrich Haböck, David Levit, and Shahar Papini. Circle starks. Cryptology ePrint Archive, Paper 2024/278, 2024. https://eprint.iacr.org/2024/278.

[11] Daniel Kang. Tensorplonk: A gpu for zkml delivering 1,000x speedups. https://medium.com/@danieldkang/tensorplonk-a-gpu-for-zkml-delivering-1-000x-speedups-d1ab0ad27e1c, 2024. Accessed: 2024-07-01.

[12] Daniel Kang, Tatsunori Hashimoto, Ion Stoica, and Yi Sun. Scaling up trustless dnn inference with zero-knowledge proofs, 2022.

[13] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention, 2023.

[14] Alan Li, Qingkai Liang, and Mo Dong. Sparsity-aware protocol for ZK-friendly ML models: Shedding lights on practical ZKML. Cryptology ePrint Archive, Paper 2024/1018, 2024. https://eprint.iacr.org/2024/1018.

[15] Tianyi Liu, Xiang Xie, and Yupeng Zhang. zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy. Cryptology ePrint Archive, Paper 2021/673, 2021. https://eprint.iacr.org/2021/673.

[16] Akaki Mamageishvili and Edward W. Felten. Incentive schemes for rollup validators, 2023.

[17] Modulus Labs. Remainder paper. https://github.com/Modulus-Labs/Papers/blob/master/remainder-paper.pdf, 2024. Accessed: 2024-07-02.

[18] NVIDIA. Cuda floating point documentation. https://docs.nvidia.com/cuda/floating-point/index.html, 2024. Accessed: 2024-07-02.

[19] Situational Awareness AI. Situational awareness: Improving ai systems for real-world applications. https://situational-awareness.ai/wp-content/uploads/2024/06/situationalawareness.pdf, 2024. Accessed: 2024-07-01.

[20] Megha Srivastava, Simran Arora, and Dan Boneh. Optimistic verifiable training by controlling hardware nondeterminism, 2024.

[21] Haochen Sun, Jason Li, and Hongyang Zhang. zkllm: Zero knowledge proofs for large language models, 2024.

[22] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023.

[23] Wikipedia contributors. Ieee 754. `https://en.wikipedia.org/wiki/IEEE_754`. Accessed: 2024-07-01.

[24] Yue Zhang and Shouqiao Wang. Proof of sampling: A nash equilibrium-secured verification protocol for decentralized systems, 2024.