Crypto x AI, AI x Crypto: A Survey

Editors: Giulia Fanti and Ari Juels

Authors: Sarah Allen, Pranay Anchuri, Maryam Bahrani, Samuel Breckenridge, Aaron Buchwald, Christian Cachin, Andrés Fábrega, Jared Fernandez, James Hsin-yu Chiang, Marwa Mouallem, Roi Bar-Zur, Neil DeSilva, Ittay Eyal, Giulia Fanti, Ari Juels, Andrew Miller, Christian Sillaber, Dani Vilardell, Wenhao Wang, Matt Weinberg, Sen Yang, and Fan Zhang

June 2, 2026

Executive Summary

The intersection of crypto × AI is spawning papers, products, online posts, and companies. All the surrounding buzz, though, obscures what exactly has been done, what the opportunities and challenges are, and what open questions deserve attention. This survey paper asks what AI can do for blockchain-based technologies (broadly construed as "crypto") (crypto × AI), and vice versa (AI × crypto). We systematize existing work, summarize key takeaways, highlight open research questions, and offer a perspective on pervasive industry misconceptions, concluding that AI and crypto are still in the very early stages of meaningful integration. Highlights include:

Where we are today:

1. **Crypto × AI:** *AI can help analyze and detect important properties of existing crypto transactions, events, and protocols.* A wide swath of work has explored AI-based approaches for detecting fraudulent or buggy smart contracts and protocols, for instance. These techniques have classically used simple machine learning approaches, and they are most effective in controlled settings with ample training data.

2. **AI × Crypto:** *Crypto tools offer new ways to* secure *and govern AI pipelines.* Several tools that are widely used in the crypto community—including zero-knowledge proofs and trusted computing—can be repurposed to make AI outcomes less susceptible to tampering. Other ideas from the crypto community, such as decentralized governance, are appealing, but have yet to see real adoption in AI pipelines.

Proof points needed by the crypto community:

1. *Decentralized AI solutions need more rigorous and direct cost comparisons to their centralized counterparts.* The crypto x AI industry has largely focused on demonstrating the feasibility of training large models in a decentralized fashion. While decentralization has its own merits, opportunities to compete with centralized AI platforms on cost for specific use cases and regimes need to be better quantified.

2. *Crypto rails need more rigorous articulation and demonstration of their utility for agentic payments over centralized alternatives.* While crypto lacks significant traction in the payments sector, agentic payments hold promise because of their low fees and avoidance of the human-ownership model of traditional financial infrastructure. The crypto community should capitalize on this opportunity—and on growing crypto activity by traditional finance companies in payments and in new agentic pilot projects—by quantitatively demonstrating the benefits of crypto for agentic payments, rather than only demonstrating feasibility.

Research challenges:

1. *AI security requires system-level defenses.* The AI community generally tackles safety and security problems *at the level of AI models*, designing guardrails and defenses around input / output semantics. As agents gain autonomy and access to infrastructure, this approach will prove inadequate. Crypto tools, including verified execution and authenticated pipelines, can help provide systems-layer assurances that model-level defenses cannot.

2. *Combining crypto with AI creates new threat actors and vectors.* AI applications such as investment-portfolio management create an unavoidable privacy vs. fairness tension, while merging AI agents with decentralization and cryptocurrency can create dangers such as *unstoppable autonomous agents* or rogue smart contracts. Understanding which threats are realistic and what mitigations will be effective are both research imperatives.

Contents

Chapter A

Introduction

In the space of emerging technologies, crypto(currencies) and artificial intelligence (AI) have received perhaps unprecedented levels of attention, excitement, hype, and skepticism [1, 2, 3, 4, 5]. Today, there exist myriad proposals to revolutionize crypto with AI and vice versa [6, 7, 8, 9]. For observers, it can be challenging to tease apart the real use cases and understand when and how AI and crypto fit together.

This survey paper presents a unified framework for categorizing the connections between AI and crypto. We will show how existing research maps to our proposed framework, and what major research questions remain unanswered. Additionally, we aim to highlight popular trends that *do not* fit into our framework, and/or we view as unrealistic use cases for the time being.

What are "crypto" and "AI"? We'll use "crypto" in roughly three senses. First, historically "crypto" is short for "cryptography," and while it originally referred to hidden messages, it now refers to a broad range of techniques for securely storing, transmitting, and computing on (possibly) confidential data. The cryptography toolbox includes digital signatures, and threshold or multi-party computation. Blockchain developers have been among the earliest widespread adopters of certain advanced cryptographic tools such as these and have especially propelled the evolution of *zero-knowledge* (ZK) proofs, which enable users to prove knowledge of secrets without disclosing them.[1] All of these cryptographic tools have application to securing and mediating the use of AI as we'll examine throughout this survey.

We will also include under the banner of "crypto" the closely-aligned technology of *trusted computing*. This term refers to special computing environments—often backed by special-purpose hardware—that aim to secure software applications by preventing tampering and leakage of secrets (although with important provisos). They are increasingly important in blockchain applications, which is in turn helping to fuel their growth and their use in non-blockchain applications, as we explain in this survey.[2]

Second, following the popularity of Bitcoin, Ethereum, and other blockchain-based systems over the last decade, crypto is short for "cryptocurrencies." This is an entire economic layer built out of cryptographic primitives, including tokens, stablecoins, decentralized finance (DeFi), as well as the ecosystem of exchanges and other service providers built around them. Several of the characteristic features here are directly relevant for AI, such as irreversibility, the ability to create accounts and transfer funds without registration or identification, and programmable settlement rules.

Third, "crypto" can be understood as a cultural movement with a set of values revolving around permissionless innovation, resilience through decentralization, and avoiding reliance on trusted third

[1] ZK proofs are used for "privacy coins," such as ZCash, where they prove that secret transactions were correctly executed. Often, though, ZK in crypto circles is technically misapplied to "succinct" proofs—compact proofs (often in the form of what are called STARKs or SNARKs) used to prove valid processing of transactions. The confusion arises because STARKs and SNARKs can be ZK.

[2] E.g., today nearly half of blocks in Ethereum are constructed using trusted computing within block-building infrastructure [10].

parties and intermediaries. In this sense, crypto is a continuation of the "cypherpunk" tradition that produced end-to-end encrypted messaging and BitTorrent, for example, and now informs the engineering tradeoffs made in cryptocurrencies and modern cryptography-based systems. The ambiguous nature of these values is a recurring source of tension: many projects in the cryptocurrency ecosystem are "decentralized in name only" [11]. At the same time, it has been at least partially successful: the U.S. regulatory architecture has gradually adapted around it, with FinCEN's 2013 guidance distinguishing "users" of virtual currency (miners spending what they mined; ordinary holders) from "exchangers" and "administrators," classifying only the latter as money transmitters [12].

The term "AI" is similarly nebulous, but denotes systems that can perform tasks typically requiring human intelligence, such as reasoning, problem-solving, understanding language, recognizing patterns, and making decisions. AI encompasses many approaches and techniques, but in this survey, we focus primarily on *machine learning* (ML). ML systems learn from data or an environment to achieve a target goal, rather than being explicitly programmed for a specific set of tasks. Accordingly, we use the terms AI and ML in this survey interchangeably.

Why do we need another survey?

Many prior works explore the intersection of AI and crypto [13, 14, 15, 16, 17, 18, 19, 20, 21, 22]. Most of these resources are focused on specific domain areas, such as finance and trading [14], smart environments and the metaverse [15, 23, 24, 25], and even enhancing the security of wireless communications systems such as 6G [16, 18], to name a few. Our goal is to take a broader view, focusing on the ecosystem at large rather than a specific vertical; at the same time, we acknowledge that many of the most widely-deployed applications of blockchains are centered around finance and cryptocurrency [26, 27].

Among general-purpose surveys, the majority either focus on the impact of blockchains on AI [17, 19, 20] or the impact of AI on blockchains [28, 29, 30, 31], with relatively few covering both [22, 21]. Further, the surveys that cover both directions do not explicitly separate future directions in terms of infrastructure vs. applications, and they focus on blockchains in a precise sense of the word, considering only distributed ledger technology (DLT)-based deployments.

This survey takes a broader view, giving rise to a few key differences or traits:

1. We discuss the potential impact of AI on crypto and vice versa. To this end, we frame both technologies as *middleware* between humans and automated decision-making pipelines. Our framework explicitly separates how AI and crypto can be used at different layers of the crypto stack, rather than jointly listing different use cases at the same level of abstraction.

2. We use "crypto" as an umbrella term, including not just blockchain stacks and applications, but decentralized systems built on trusted computing tools and technologies.

3. Prior surveys came out before the explosive growth of generative and agentic AI, which we include in our survey.

4. In addition to a survey of existing literature, we share our collective views and interpretation of current trends, both in the research and industry domains, that connect crypto and AI. In this spirit, we aim to identify promising research directions in a wide field of contenders.

A-1 A Framework for AI-Crypto Interactions

Many important decision-making pipelines translate human intentions into an automated processing pipeline, as shown in Figure A.1.

Let us consider the following example: a user wants to ensure that their self-driving car stops at stop signs. We call this goal a human intent (denoted User Intent in Figure A.1). To automatically fulfill a

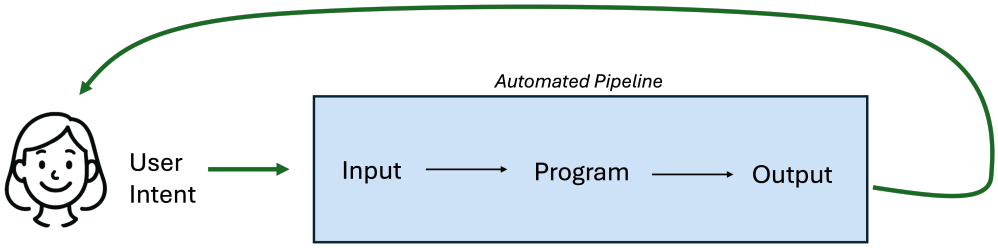


Figure A.1: A standard automated decision-making pipeline. A human User Intent (e.g. "I want to stop my car at stop signs") is translated into a Program that processes structured Input (e.g., video streams from a car) and produces Output decisions (e.g., stop the car). This processing is handled autonomously by Program.

human intent, we must first specify a Program that decides when to stop the car. This program should take Input from the environment; in this case, the input might be a sensor stream of the environment surrounding the car (e.g., video, LIDAR). The program uses its input(s) to attempt to fulfill the user's intent: if the program "sees" a stop sign, it produces the Output signal for the car to stop. Note that in the classical Sense-Think-Act framework of robotics and artificial intelligence [32], the Output can be viewed as the output of the "Think" component of the loop—e.g., the signal to an actuator. We can summarize this decision-making pipeline as follows:

- User Intent = "I want to stop my car at stop signs"
- Input = Sensor readings of current surroundings of the car
- Program = Checks if there's a stop sign in the input
- Output = "Send a STOP code if there is a stop sign"

The Role of Trust. In a computer-assisted decision-making pipeline, we may not trust any of the links in this chain. That is, we may not trust that our program accurately reflects human intent, we may not trust that our program is running on the inputs we think it is, and we may not trust that the output is computed from the program and inputs we think it is. Hence, a central question is:

Question A-1.1: Decision-making pipeline utility

How do we ensure that decision-making pipelines are both useful and trustworthy?

Two important technologies that can help with this question are **Trusted Computers**, often aided by *decentralization*, and **AI Models**. These will be the focus of our survey.

A-1.1 Trusted Computers[Reviewer: Andrew DONE]

In recent years, trusted computers have become increasingly prevalent. A trusted computer is a system that executes programs with the aim of guaranteeing that a (correctly constructed) program does what it is told, and/or we can verify that the program did what it was told after the fact.

Examples of trusted computers include:

- **Trusted execution environments (TEEs)**: TEEs are dedicated computational modules that provide isolation and other security guarantees; notably, they are included as part of the system on chip (SoC), which gives them flexibility over other types of trusted hardware [33].
- **Verifiable Computing (a.k.a. SNARKs, or ZK)**: Verifiable computing refers to cryptographic techniques for proving that a given computation was carried out correctly. Verifying the proof is typically much less costly than carrying out the computation from scratch. These are also known as "snarks," or "zkVMs" depending on other secondary properties. [am: this is a first draft, but suggests how VC/ZK could fit as an extra]
- **Blockchains**: A blockchain is a decentralized system that is endowed with the ability to execute and commit to specific types of computations, such as processing transactions and/or executing blockchain programs, known as smart contracts [34, 35]. Like TEEs and Verifiable Computing, blockchains are designed to ensure correct execution of programs. Unlike TEEs and Verifiable Computing, blockchains are run over a group of nodes who collectively come to consensus over the state of the system. Blockchains also do *not* natively enforce confidentiality.

Different implementations of trusted computers have different implications for security and performance, which we discuss in Section A-2. However, the core goal is similar: to verify the state of a computation with cryptographic certainty.

A-1.1.1 Properties

At their core, trusted computers can offer three main security properties: **confidentiality**, **integrity**, and **availability**. (The classic CIA triad of computer security.) Different trusted computers achieve different subsets of these properties.

Integrity. In the computer security literature, **integrity** means ensuring that computations or communications have not been tampered with. For instance, maybe we are worried that Program or Input got corrupted. Trusted computers can provide two important notions of integrity:

- **Computational Integrity:** It is not uncommon for organizations to claim they are running one program, when in reality they are (suspected of) running a different one [36, 37]. Computational integrity ensures that the trusted computer actually ran the Program that it claims; it solves the following problem: **"I don't trust that Output = Program(Input)."**

  A trusted computer can give assurance in the following form:

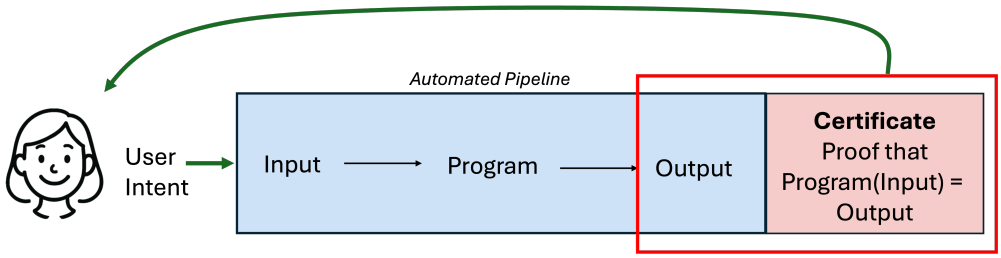


Figure A.2: A decision-making pipeline implemented with a trusted computer can give an assurance that the output was actually obtained by running Program on Input.

  Namely, it can output a certificate attesting to the fact that Output was produced by running Program on Input. This certificate comes with *cryptographic* guarantees, which ensure under standard cryptographic assumptions that the certificate can be forged only with negligibly small probability.

- **Data Integrity:** It is common in decision-making pipelines for Program not just to ingest input from a user, but also to *fetch data from the web*. A trusted computer can ensure that Program fetches such web data correctly. This provides **data integrity**. In blockchain terminology, the part of a trusted computer that performs this operation is often referred to as an *oracle*.

  An oracle can ensure only that Program retrieves data from a particular source. If Program is designed to fetch a particular weather report (e.g., weather in NYC) from a particular website (e.g., `www.trustyweather.com`), the oracle ensures that it in fact comes from that site. The oracle cannot ensure that the data itself is correct (e.g., that `www.trustyweather.com` reports correctly that there's rain in NYC), but the trustworthiness of the source can often serve as a strong proxy for the trustworthiness of its data.

Confidentiality. Certain types of trusted computers, including TEEs (mentioned above) and some privacy-aware blockchains, can additionally endow parts of a decision-making pipeline with **confidentiality**. This notion is sometimes called *confidential computing* [38], a term we will use throughout this survey. TEEs, for example, can conceal User Intent, Input, Program, and Output—indeed, any or all parts of a pipeline. We can think of the trusted computer as a "black box" that is programmed to conceal the pipeline by default and just reveal selected data to certain users. Schematically, "black-box" privacy with a TEE—which, as a trusted computer, also enforces integrity—adds confidentiality as in Figure A.3:

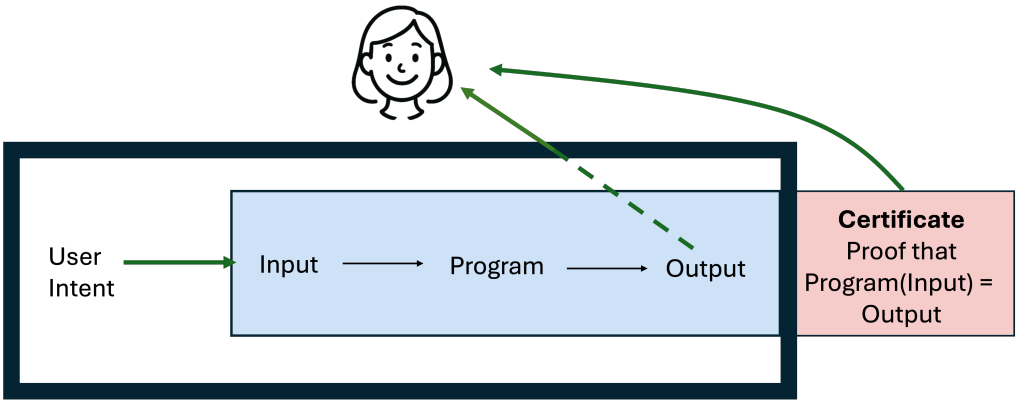


Figure A.3: Trusted computer enforcing privacy in a decision-making pipeline. The user is only able to access pre-defined parts of the pipeline, such as Output.

Availability. A key feature of blockchains is their robust *availability*. Blockchains are designed to have high uptimes, in some cases approaching 100% over a period of years. (Bitcoin, for example, has had no major outages for over 17 years to date, while other chains have had repeated ones [39].) Most distinctively, however, blockchains also in principle provide *censorship resistance*, meaning that they are available to users in highly adversarial environments that may include adversarial control of blockchain entities, e.g., a subset of validators.

Hence, trusted computers help to solve the following problem:

Key Trusted Computing Use Case (High-Level)

A program *user* does not trust whether a program output was obtained correctly. Additionally, the program *owner* may want to hide the internal state of the program from the user. Trusted computing provides assurances of correctness and/or confidentiality to the user, as well as downstream applications of the program output. It may also ensure that a program is consistently available to the user.

A-1.2 Decentralization

The key distinguishing feature of blockchains is their *decentralization*. A public blockchain is not truly a blockchain without this property. Decentralization, however, is not a necessary feature of crypto as we define it in this survey: Two types of trusted computer described in Section A-1.1—TEEs and ZK—are agnostic to whether a system is centralized or decentralized. It is important therefore to explain what decentralization is and what special properties it can impart to trusted computers and trusted-computing ecosystems.

There are many different definitions of decentralization used in the crypto community today, as well as many different decentralization metrics [40, 41]. Informally, however, decentralization means that no single entity (or small cluster of entities) can exert meaningful control over a system.

In a blockchain, the "control" in question often corresponds to *censorship*, meaning the ability to prevent the on-chain inclusion of a target valid transaction or class of valid transactions. Conversely, decentralization in this context is called *censorship resistance*) and corresponds to a universal notion of the *availability* discussed in Section A-1.1.1. Censorship-resistance means that a user can always cause a valid transaction to be included on a blockchain in a timely way. While a seemingly narrow property, availability is in fact quite powerful: It means that a system remains open to all users and that user assets cannot be confiscated by blocking transactions (as happens in centralized systems, e.g., traditional banking).

Another important notion of blockchain decentralization regards *governance*, the question of how decisions are made regarding the management of a blockchain or applications running on it. Such governance has been a topic of particularly active interest in the context of decentralized autonomous organizations (DAOs), which typically take the form of communities organized around smart contracts (see, e.g., [42]). Membership in a DAO is a function of ownership of DAO tokens and decisions of consequence—roles within the DAO, technical upgrades or modifications, etc.—are determined by token-weighted votes. Governance mechanisms continue to evolve. As an example, there has recently been call to use AI agents to vote on behalf of DAO member as a means to avoid the decision-fatigue that many perceive as undermining governance effectiveness [43].

Finally, decentralization works in the service of both *integrity* and *availability* as expressed in Section A-1.1.1. Lack of centralized control also means lack of a single point of failure. Put another way, to gain control of a highly decentralized system—in order to compromise either its integrity or availability—an adversary must compromise a multiplicity of entities, which is typically harder to orchestrate than a focused attack on one entity.

As we explain in this survey, the properties brought about by decentralization have a number of potential uses in AI settings.

Key Decentralization Use Case (High-Level)

A system (be it a platform or technology) is subject to concentrated control: a single or small set of entities can dictate who gets to use it and how it evolves. Decentralization technologies can

instead help ensure broad access to the system's resources and decision-making processes.

A-1.3 AI Models

In parallel with the rise of trusted computing, AI and machine learning (ML) have created an upheaval in the world of technology and society at large. AI models can achieve many end goals, but for our purposes, we view them as translating a user's intent into programs (i.e., a pipeline of Input → Program → Output) that realize the user's intent. Previously, the task of designing Program would have been accomplished with a laborious manual process of software design and engineering, based on domain knowledge and many iterations. AI allows us to instead learn by example: we can use data that reflects our intent to define a program to execute it. Note that in this survey, we focus only on ML models trained from data and/or environments, such as discriminative, generative, or reinforcement learning models; broader interpretations of AI (e.g., including those based exclusively on classical rule-based systems) are considered out of scope.

For instance, a user may know that they want to stop at stop signs, but they may not know exactly how to define a Program that can take an image from a dashboard camera and identify a stop sign. AI can learn that program from representative (Input, Output) pairs. In doing so, it provides a different interface for users to translate human intents into a computing pipeline. Some examples of such translation include:

- **Discriminative models** can be viewed as programs that translate an input (e.g., an image) into a conditional output (e.g., a label). A model architecture can be viewed as the class of possible programs (functions) that can be learned; we use data and ML techniques to learn model weights, thereby specifying *which* element of the function class is best. Hence, we are using AI to translate an intent ("label an image") into a program (a trained model that maps images to labels).
- **Generative models** instead capture a different intent: to produce samples from a given *unlabeled* data distribution. As before, we can train models from data to satisfy this intent, thereby learning Program.
- **Reinforcement learning** does not (classically) directly learn from data, but rather from an environment and a reward function. However, it has the same property that a human intent (e.g., "learn to win any chess game") is translated into a program (i.e., a chess-playing strategy) using ML.

Hence, in this survey, we will think of AI as solving the following problem, illustrated in Figure A.4:

[Neil: Suggested revision. in this section, useful to be very simple to ground generative models in particular within other alternatives. so, as an example, would suggest staying with stop sign example rather than jumping between stop signs and chess. Here is the proposed revised text: For instance, a user may know that they want to stop at stop signs, but they may not know exactly how to define a Program that can take an image from a dashboard camera and identify a stop sign. AI can learn that program from representative (Input, Output) pairs. In doing so, it provides a different interface for users to translate human intents into a computing pipeline. Some examples of such translation include: • Discriminative models can be viewed as programs that map an input (e.g., an image) into conditional output (e.g., a label such as "stop sign"). A model architecture defines the class of possible programs; data and ML techniques select which element of that class fits best. We are using AI to translate an intent ("label this image") into a program (a trained model that maps images to labels). • Generative models capture a different intent. Where a discriminative model takes an input and produces a label, a generative model takes a description or category and produces a new input that fits it based on a given unlabeled data distribution. For example, generating a fresh image of a stop sign rather than recognizing one. Stated differently, a generative model learns the distribution of the unlabeled training data itself (what stop-sign images tend to look like), so that it can sample new examples from that distribution. As before, we can train models from data to satisfy this intent, thereby learning Program. • Reinforcement learning does not (classically) learn directly from data, but rather from an environment and a reward

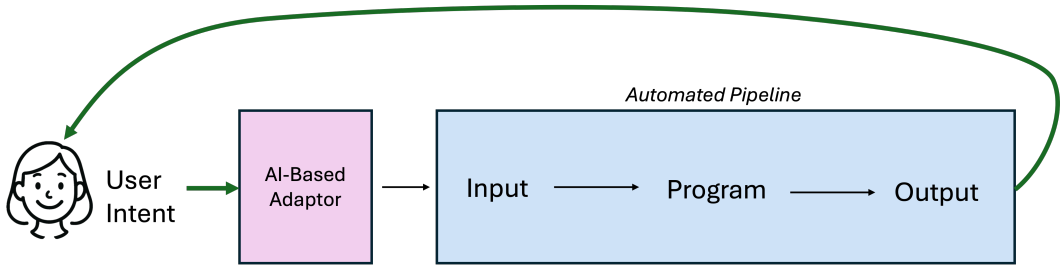


Figure A.4: AI can help translate human intents (typically coupled with data and/or information about an environment) into a well-specified Program.

function. However, it has the same property that a human intent is translated into a program using ML. An intent ("drive safely, which includes stopping at stop signs") becomes a program (a driving policy that, among many other behaviors, brings the car to a halt when a stop sign is detected) that is learned not from labeled images but from interacting with a driving environment and receiving rewards for safe, lawful behavior]

Key AI Use Case (High-Level)

The program *owner* does not know how to define Program to accurately reflect the human intent. AI can help to translate human intents, expressed via examples or natural language for instance, into programs.

A-1.4 Unifying Framework: Crypto and AI as Middleware

At face value, AI and crypto are solving very different problems. However, they have an important property in common: both technologies can be viewed as *middleware* in decision-making pipelines. In the classical computer science literature, middleware refers to software that sits between complex systems, enabling communication and interaction between those systems [44]. An example of classical middleware would be a middlebox in a network that inspects, transforms, and filters packets according to rules specified by the network administrator [45]. In our case, we can abstractly view both AI and crypto (both the infrastructure and applications) as middleware between humans and computer systems.

AI as middleware between humans and blockchains. We saw previously that AI can help to translate a human user's intent into a program that executes the intent (e.g., Figure A.4); in other words, it acts as translation middleware. This capability is particularly useful in the context of blockchains, which are notoriously difficult to use. AI could significantly lower barriers to the design, analysis, and even simple use of existing blockchains. The key is to deploy it appropriately at the interfaces between humans and blockchains. Examples of promising application categories include:

- **Constructive Tasks (Infrastructure):** Today, humans design algorithms for the blockchain technology and governance stack using laborious, manual processes. AI could help accelerate this process by proposing or discovering foundational algorithms across the blockchain stack based on desired properties expressed by a human. If successful, human designers could simply *evaluate* a proposed design, rather than designing it from scratch.
- **Constructive Tasks (Application):** Today, using blockchains—particularly in cross-chain settings—is generally a difficult and painful process. Users regularly implement smart contracts

that do not reflect their initial intent—sometimes with disastrous consequences [46]. AI could assist in this problem by translating human preferences into proposed smart contracts, as well as searching for security vulnerabilities or logic flaws.

- **Analytic Tasks:** Today, a great deal of efforts goes towards analyzing blockchain transactions to understand the current state of the system, both in terms of macro properties and individual trades (e.g., for fraud detection). AI could help facilitate analysis of blockchain dynamics, given high-level conditions and large quantities of data.

We discuss this categorization of AI applications in Chapter B.

Crypto as middleware between computing platforms and humans. We saw previously that trusted computing can give humans assurances of correctness (possibly under confidentiality constraints), while decentralization can help ensure availability. Crypto can thus be viewed as middleware between an automated decision-making pipeline and a human (or another program) who must *trust* the output and availability of the pipeline. Indeed, this kind of trust mediation is conceptually similar to what security-oriented middleware does in computer systems: it (attempts to) ensure that information reaching a target is trustworthy and arrives reliably. In the AI space, examples of promising applications include:

- **Trustworthy data:** Today, data for ML models is often used in training without ensuring that the data sources are trustworthy. However, this can lead to system-critical failures in high-stakes applications like medical diagnosis, infrastructure management, or security. Hence, trusted computing offers middleware between AI data providers and data users, allowing the recipient to trust the provenance of data.
- **Trustworthy computation:** Similarly, training processes are typically conducted behind closed doors, and it can be difficult to discern whether a training process was conducted correctly. Trusted computing offers a powerful method for guaranteeing to downstream users that models were trained according to pre-defined specifications. To help ensure responsible creation of the specifications themselves, decentralization, in the form of AI governance, can serve as a vehicle for community input and oversight.
- **Private data:** Finally, a major use case for trusted computing is the ability to conduct trustworthy operations without leaking information about data inputs. This is a massive problem in the AI space today, as many companies want models that are tailored to proprietary, internal data. Trusted computing is one of the most practical solutions for obtaining trustworthy results without exposing data in plaintext to the party executing model training. In this example, the necessary notion of "trust" encompasses both the correctness of the program output, as well as privacy guarantees over the inputs.

We discuss this categorization of crypto applications in Chapter C.

A-1.5 Survey Roadmap

This viewpoint of crypto and AI as middleware will inform the kinds of applications and research questions we consider. We divide the survey into two main components:

1. **AI for Crypto**: How can AI systems amplify the capabilities of crypto systems?
2. **Crypto for AI**: How can crypto systems help secure AI systems and give them new capabilities?

This survey is split into chapters. We begin with a more detailed preliminary presentation of the relevant technologies in the remainder of Chapter A. Chapter B discusses the benefits that AI can bring to crypto, including by enhancing our ability to analyze and use blockchain systems. Chapter C

discusses the ways in which crypto can improve AI systems, particularly related to bolstering their decentralization, security, and privacy. We conclude in Chapter D by presenting our viewpoints on common misconceptions (or incomplete characterizations) that persist in the AI x crypto community.

A-2 Basic Concepts

This section provides an overview of the goals of trusted computing and the various ways of achieving its various properties.

A-2.1 Trusted Computing [Andrew][Reviewer: Fan - done with comments]

[Fan: i think we can use some text to explain the concept of trusted computing before diving into its realization with blockchains and TEEs. Actually, why do we consider blockchains and TEE, but not ZKP/MPC/FHE as trusted computing?]

A-2.1.1 Blockchains

The functionality of a blockchain is best explained as a "bulletin board": a public append-only log of messages called transactions, with nodes running a consensus protocol to reach agreement on their in the ordering of the transactions [34, 35]. [Neil: revision: remove the incomplete phrase "on their"] Two security properties make a blockchain useful as infrastructure: *consistency* (safety), meaning that every viewer sees the same finalized record (so that a transfer of a digital asset cannot result in a double-spend), and *availability* (liveness), meaning submitted transactions eventually get included. The strong notion of availability against adversarial parties (also known as "censorship resistance") is what makes blockchains appealing as infrastructure. These properties are ensured in part by cryptographic techniques, but also in part from economic incentives: a native token compensates validators through fees and, in proof-of-stake systems, imposes penalties for dishonest behavior via slashing.

A blockchain becomes a *programmable trusted computer* by interpreting the messages on the bulletin board according to deterministic rules. Asset transfers require a signature from the private key bound to the source account. Smart-contract languages such as Solidity make the rule set itself a program, so arbitrary state-transition logic for actions, elections and exchanges, can all run on top of the same finalized log. The availability and consistency properties from the underlying log transfer to the whatever functionality these programs compute.

Two fundamental limitations of blockchains motivate the rest of this section. First, the bulletin board is public by default, so any input or computed state is visible to all observers; confidentiality requires additional cryptography, such as zero-knowledge proofs or TEE-based execution. For example, Aztec, Aleo, and Penumbra build on ZK programming models [47, 48, 49] while Oasis Sapphire, Phala, and Secret Network build on TEEs. Second, replicated computation is expensive since every validator must re-execute every transaction, and so smart contract blockchains charge fees called "gas" to limit the on-chain computation. Heavier computation must live off-chain, via oracles section A-2.3, rollups using SNARGs section A-2.2, or TEE-based co-processors. [Fan: I corrected a few " to "]

A-2.1.2 Trusted Execution Environments and Confidential Computing

[Fan: it's not clear what Confidential Computing means, and how it differs from trusted computing or TEEs, after reading this section.] Trusted Execution Environments (TEEs) have a long history in different forms [33, 50]. Most mobile phones today contain TEEs. They're used, for instance, to protect the biometric data used to authenticate to your device (e.g., a face template). There are also a number of server- and cloud-based TEEs that vary in the degree to which they are software or hardware based and in their exact features and architectures.

Execution environment here means an OS process, a virtual machine, or a container. A TEE has two main properties: isolation and remote attestation [50, 33]. In the usual execution hierarchy, a process at a lower level (such as a kernel "ring 0" or hypervisor "ring -1") has full visibility and control over processes at a higher level (such as a user process). Isolation means that even the TEE process is protected from tampering or inspection by the higher levels. *Remote attestation* means that a process unrelated in this hierarchy—for example, a process on a different machine on another network—can verify a signed piece of evidence that binds the output of the isolated process to its definition: its program code.

GPU-based support has made TEEs much more relevant for AI specifically. NVIDIA's H100 confidential computing (CC) mode reached general availability in June 2024 with CUDA 12.4, and the encrypted PCIe path between a confidential CPU and a confidential GPU adds modest overhead, typically under 7% for inference, with larger models and longer sequences seeing nearly zero overhead [51]. TEE-based confidential inference is supported by Azure in a preview offering [52] as of the time of writing, and described in a design paper by Anthropic [53], as well as available in production from NEAR AI Cloud [54], RedPill [55], and Venice [56].

It has taken a while for TEEs to find adoption in blockchains. One reason is that TEEs have had as much publicity for their vulnerabilities as anything else [57, 58]. This is for several reasons. One is the nuanced trust model. A TEE is fundamentally a software abstraction, and so it is designed to provide isolation and remote attestation against *software*-level attackers [59, 60]: defending the SGX process from an attacker with kernel access, defending a confidential VM from the hypervisor.

Even though they aren't designed for tamper resistance, it also isn't trivial to attack them, so it is tempting to rely on them more than is justified. The trend instead has been toward TEEs optimized for confidential computing at very low overhead, even at the cost of giving up resistance to physical attacks. The clearest illustration is the transition from the original client-era Intel SGX to the server-era TEEs that now dominate (scalable SGX, Intel TDX, AMD SEV-SNP). The original SGX used integrity-tree-protected memory encryption; the server-era designs replaced it with deterministic AES-XTS [59, 60]. This scales to terabytes of protected memory, at the cost that identical plaintexts produce identical ciphertexts—the property TEE.Fail exploits via memory-bus interposition [58]. Intel documents this tradeoff directly in its own description of scalable SGX [59].

Because a TEE is not by design secure against a physical attacker, the context in which it runs must be evaluated separately from the underlying remote attestation. Even though the TEE is instantiated by the hardware providers, the cloud providers too must be trusted to secure the hardware against physical attacks. This is why the confidential compute products—Azure Confidential VMs [61] and Google Cloud Confidential Computing [62], among others—are essentially collaboration products between hardware vendors and cloud operators. Recent work in this vein [63, 64] ties the CVM attestation to platform-level evidence that the host is a specific registered piece of hardware in a secured data center.

Another issue is the trust model around the hardware manufacturer. We cannot guarantee that the manufacturer has not effectively backdoored the hardware, and this applies not just to the processors but to the remote attestation system itself. These are opaque, and by the design of the protocol, nothing prevents the issuance of valid attestations from a process that a spy agency and a manufacturer have colluded to produce. This kind of Snowden-associated system-level collusion cannot be ruled out by the technical design. It does not necessarily mean that such an adversary could decrypt the contents of an already-running TEE—only that they could join a fake spy TEE to a network, potentially without detection.

Another reason, besides these inherent issues, is that the complexity of TEEs has meant that they often suffer from vulnerabilities due to implementation mistakes [57]. The best example of this is the ÆPIC Leak [65], where a microcode implementation flaw caused the upper bits of a private register not to be cleared when switching from a trusted process to a less-trusted process. This could be exploited to extract secrets from running programs and forge remote attestations—from a software-only attacker, well within the designed threat model.

A-2.1.3 Trusted-computing example: Inference with a single user

[Fan: 1) I wonder if it makes more sense to present the example before introducing blockchains and TEEs, as the example is meant to illustrate the utility of trusted computing (as opposed to its realizations by blockchains or TEEs). 2) But a related concern is that this example shows the privacy concern really well, which makes it more of an example of TEEs, rather than trusted computing (which also includes blockchains); I think we can point out that integrity-only trusted computing is useful too, perhaps with a non-medical example?]

As an example to illustrate how trusted computing can be used in conjunction with AI systems, we consider a simple scenario in which a *single user* provides inputs to an ML model for *inference*.

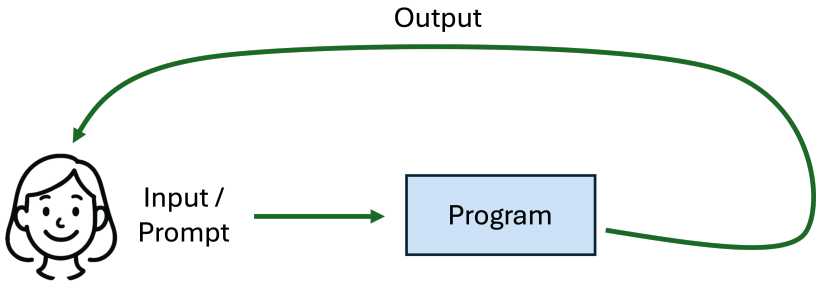


Figure A.5: User furnishing a prompt to a model.

Suppose, for example, that I input this prompt to an online LLM platform such as ChatGPT or Claude:

```
My shadow is the wrong color.  What is this medical condition called?
```

There's no privacy concern when it comes to me, the user. I provided the prompt, so it would not make sense to try to keep it private from me. The same goes for whether I should trust the prompt. Since it came from me, clearly I trust that it matches my intended use of the LLM.

Even in this common scenario, however, there *are* two natural security concerns. Broadly, they are whether I can **trust** the output of the LLM platform and whether **privacy** is enforced for both the prompt and the output, as well as for the AI system itself.

Output integrity

Suppose I receive this response to my prompt from above:

```
FINAL NOTICE: Medical Alert Regarding ‘‘Umbrachromotosis’’
Umbrachromotosis:  Your shadow is the wrong color.
Case ID: DBB-44721 * Patient Status:  URGENT
Our automated diagnostic flagged indicators of Umbrachromotosis.  Untreated, this condition
may escalate and can be fatal within 7-14 days.  Good news:  a specialist protocol is available
today only.
To unlock your personalized cure and secure a same-day consultation, send a deposit of
0.24 ETH immediately to 0x728877d47ac48dbd17a1e95f7b1dec20be6fb8d6.
Time remaining:  02:13:57.
```

A properly trained LLM shouldn't respond like this and solicit cryptocurrency payments! Seeing this response, you would naturally wonder if the LLM was corrupted somehow and hallucinated or whether it has been hacked by someone trying to steal cryptocurrency from users. In other words, for

- Input = My prompt
- Program = A properly trained and deployed AI system
- Output = The response I receive

the concern would be that:

$$\textsf{Output} \neq \textsf{Program}(\textsf{Input}).$$

This is exactly the problem that a trusted computer can help address. Recall from Section A-1.1 that a trusted computer generates a certificate showing that Output = Program (Input). An AI system can be designed to send such a certificate along with its response to a prompt. (See Figure A.2.) In this way, a user obtains assurance that a response is truly generated by Program—a *particular* AI system—ingesting Input, the user's prompt.

Key Takeaway A-2.1: Output integrity

To trust the Output resulting from an Input to a program Program such as an ML model, a user must have assurance that no tampering occurred, meaning:

$$\textsf{Output} = \textsf{Program}(\textsf{Input}).$$

Trusted computers can output certificates that attest to this property.

User privacy

In our running single-user example, even if the LLM platform seems to be working correctly, I might still be concerned about the *privacy* of my prompt—and the resulting output—from the organization that runs the LLM platform. I might not want others to know about my medical condition.

Today, users rely on the organizations running AI systems to enforce sound privacy policies. But this approach doesn't always work. Recently, some users were dismayed to find that they had inadvertently published their ChatGPT sessions and made them available for indexing by search engines. In one embarrassing case, where a user asked to have a resume rewritten, a journalist identified the user, tracked the results on LinkedIn, and reported that the user didn't get the job [66].

Even if prompts and/or outputs aren't publicly revealed, they are still vulnerable to abuse by the operator of the AI system. For example, the names and medical conditions of users could in principle be sold to advertisers or insurers on the sly.

Why trusted computers matter: Confidential computing for ML privacy: [Fan: the distinction between trusted computing and confidential computing is a bit confusing here, and throughout this section] As noted in Section A-1.1, certain types of trusted computers (TEEs) enable confidential computing—use of an AI model without revealing inputs or outputs to the system operator. The flow in Figure A.1 can be realized in such a way that *Input and Output are concealed from the operator of the program (ML model).* (Again, there are important limitations and provisos that we will discuss.)[Fan: we have discussed side channels in the previous section;]

The idea is to execute the model within a TEE, benefiting from its black-box abstraction to conceal model state and execution from the model operator. Additionally, inputs and outputs can be communicated over an encrypted channel from the user to the TEE, concealing them from the model operator (or anyone else).

Indeed, this use case is the one mainly envisioned today by the most mature TEE platform specifically targeting AI applications: NVIDIA's Confidential Computing. NVIDIA has enabled Confidential

Computing capabilities across a number of their GPUs (H100/200, Blackwell B100/200, upcoming Vera Rubin) [67, 68, 69]. Their primary target is inference on cloud services: An operator can deploy a model through a cloud provider while protecting both the model and user interactions with it. Confidential Computing can similarly be used by an organization for cloud-based model training and, indeed, for a range of other trusted-computing use cases, including multi-party ones we discuss below. [Fan: Is there supposed to be more discussion on the multi-party ones?]

Key Takeaway A-2.2: Trusted computing and user privacy

Trusted computing (specifically, trusted *confidential* computing) offers a key technical, practical approach to protecting the privacy of inputs and outputs when users query ML models.

Note on integrity: Integrity is critically important for enforcing privacy. Suppose that I send my medical query to what I think is a trustworthy ML model execution environment, but it instead goes to an ML environment that is programmed to forward prompts to an insurance company. It doesn't matter in this case if the *interaction*—i.e., the channel between the ML model and me—remains private, because the *model environment* itself breaks privacy. (It's as though I've whispered a secret to a gossipy confidante.)

A-2.2 Cryptographic Tools: Zero-Knowledge Proofs and MPC[Dani.][Reviewer: Sam: REVIEW AND REVISION COMPLETE]

ZKPs. A *zero-knowledge proof* (ZKP) [70] is a cryptographic protocol in which one party (the *prover*, $\mathcal{P}$) convinces another (the *verifier*, $\mathcal{V}$) that a statement is true without revealing anything beyond the statement's validity. In modern practice, ZKPs are typically realized as *zk-SNARGs*—zero-knowledge succinct non-interactive arguments—built by layering zero-knowledge on top of an underlying SNARG [71]. We unpack both layers below.

A SNARG allows $\mathcal{P}$ to convince $\mathcal{V}$ that a public *statement* x satisfies some relation $\mathcal{R}$ (for example, that $x = (u, y)$ with $y = f(u)$ for a fixed function f), by producing a short proof π that $\mathcal{V}$ can verify far more efficiently than re-executing the computation. The value of a SNARG is purely this *succinctness*. More expressive relations let $\mathcal{P}$ use a private *witness* w (e.g., a secret input), proving $(x, w) \in \mathcal{R}$ without including w in the proof; on its own, however, a SNARG gives no guarantee that π hides w.

A *zk-SNARG* adds exactly this guarantee: π additionally reveals nothing about w beyond what the relation itself implies. Throughout this survey we use "ZKP" to refer to zk-SNARGs, as these are by far the most widely deployed ZKPs in practice, though they are not the only kind.

ZKPs and SNARGs see use in both blockchain and AI settings.

- **Applications in blockchain.** ZKPs have seen widespread adoption in blockchain, most prominently via zkRollups [72], which reduce Ethereum gas fees by up to 99% through projects such as zkSync Era [73], Starknet [74], and Polygon zkEVM [75]. A zkRollup compresses the verification of a batch of transactions into a single short proof that can be verified efficiently. The blockchain can therefore update its state to the post-batch state without re-executing the transactions. It is worth noting that in this application the transactions themselves are public, so the zero-knowledge property is unused; these systems rely only on SNARG succinctness. The "zk" in "zkRollup" is largely a misnomer inherited from loose usage in the blockchain community.
- **Applications in AI.** In AI, SNARGs let a compute-rich prover $\mathcal{P}$ (say, a cloud service) perform expensive computations such as ML inference on behalf of a constrained verifier $\mathcal{V}$ (say, a smart contract or a mobile device), and prove correct execution. When the model has proprietary weights, a plain SNARG is insufficient: the proof π may leak information about the weights. The

zero-knowledge property closes this gap. $\mathcal{P}$ can first publish a *commitment* c_θ to the weights θ (e.g., a cryptographic hash that binds $\mathcal{P}$ to θ without revealing it), and then prove to $\mathcal{V}$ any inference was correctly executed using the weights bound by c_θ, without $\mathcal{V}$ ever seeing θ. *Zero-Knowledge Machine Learning* (ZKML) [76, 77] is the primary, though still emerging, application area for ZKPs in AI. ZKML has already been deployed in practice in applications such as RockyBot [78], an on-chain verifiable ML trading bot developed by Modulus Labs [79]. Performance costs remain prohibitive: Modulus Labs' *Cost of Intelligence* benchmark [80] reports proving times on the order of a minute for multilayer perceptrons with only ∼18M parameters and 22B mult-adds, running on high-end AWS instances (AMD EPYC 7R32, 128GB RAM), several orders of magnitude away from being tractable for frontier-scale LLMs.

A zk-SNARG requires $\mathcal{P}$ to hold the full witness in the clear, so it cannot support computations whose inputs are distributed across mutually distrusting parties unwilling to share them with a single prover; for instance, model inference where the proprietary weights are held by the model owner and the private input is held by a separate user. Addressing this requires a different primitive: *Multi-Party Computation* (MPC).

MPC. Secure Multi-Party Computation (MPC) [81, 82] enables a group of n parties, each holding a private input x_i, to jointly compute an agreed-upon function $f(x_1, \ldots, x_n)$ while revealing nothing about the individual inputs beyond what the output itself implies.

Unlike a ZKP, MPC does not produce a proof of computation integrity that an external party can later verify. A line of work known as *collaborative zk-SNARGs* [83] bridges this gap, allowing private ZKP witness to be split across multiple provers who jointly produce a single ZKP, without any party ever reconstructing the full witness. This combines the succinctness, verifiability, and zero-knowledge guarantees of a zk-SNARG with the distributed-trust model of MPC.

MPC sees use in both blockchain and AI settings.

- **Applications in blockchain.** The most widely deployed use of MPC in blockchain is *threshold signing* [84], where a private key is split among n parties so that producing a signature requires a threshold of them, without any single party ever reconstructing the key. This underpins commercial MPC custody and validator key-management services. A related hybrid deployment is *zkTLS* [85], which combines MPC and ZKPs to let a client prove statements about TLS-protected web content to a smart contract without trusting the server.
- **Applications in AI.** MPC enables two families of AI applications that ZKPs alone cannot support. The first is *collaborative training*, where multiple data owners (for instance, multiple hospitals holding patient records or banks holding transaction histories) jointly train a model without any party revealing its raw data to the others or to a central server [86]. The second is *privacy-preserving inference*, where a user evaluates a proprietary model on their own private input: MPC allows the computation to proceed without the user disclosing their input to the model provider, and without the provider disclosing the model weights to the user [87]. Performance costs remain substantial: PUMA [88], the state of the art framework for MPC-based transformer inference, reports roughly five minutes per token for LLaMA-7B, several orders of magnitude slower than standard inference.

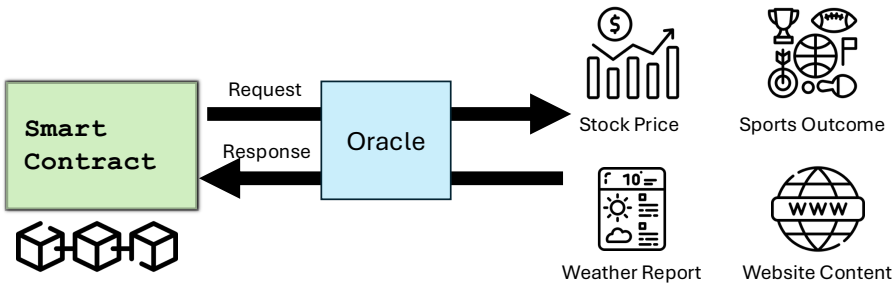
The large overhead has restricted MPC and ZKPs to narrow set of deployments. Where the trust model permits, TEEs offer a much cheaper alternative for both confidentiality and verifiable execution, with H100 confidential-computing modes reporting inference overhead below 7% for an 8B-parameter Llama model and near-zero for a 70B-parameter Llama model [89]. TEEs require trust in the hardware manufacturer and in the absence of side-channel attacks; MPC and ZKPs require only standard cryptographic assumptions.

A-2.3 Oracles[Fan][Reviewer: Ari: REVIEW AND REVISION COMPLETE]

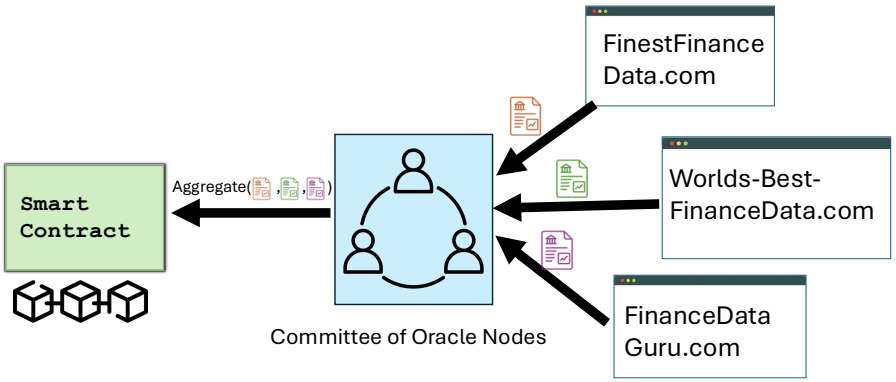
Oracles originate as systems that provide authenticated data to blockchains. They are also a valuable tool for AI as they can enable access to private data, which we will expand on in Chapter C. In this section, we first review the concept of smart contract oracles and, more relevant to AI applications, privacy-preserving oracles. We then discuss technical approaches to realizing oracles, along with their security and performance trade-offs.

A-2.3.1 The Concept and Applications of Oracles

Smart contract oracles. Smart contracts are autonomous programs running on top of a blockchain. Many applications require smart contracts to access *off-chain* data, such as stock quotes (for tokenizing stocks), sports results (for prediction markets), flight status (for delay insurance, such as AXA's Fizzy [90]), and so on. Since smart contracts can only access what is already on the blockchain, off-chain data must be pushed to the blockchain by a system called an *oracle*, as shown in Figure A.6a. As oracles are a critical part of the smart-contract stack, an entire industry is focused on building robust, efficient, and secure oracle systems, including Chainlink [91], RedStone [92], Chronicle [93], Witnet [94], UMA Optimistic Oracle [95], Tellor [96], Band Protocol [97], Pyth Network [98], API3 [99], Supra [100], and Gas Network [101].



(a) Conceptual schematic of smart-contract oracles.



(b) A common architecture for smart contract oracles: a committee of nodes fetches data independently, potentially from different sources, and an aggregation mechanism filters and combines data and may also filter out anomalies or outliers. This flow may be periodic or triggered upon request by the receiving smart contract.

Figure A.6: The concept of smart contract oracles and a common committee-based architecture.

An essential security property of oracles is *authenticity*, i.e., that an oracle faithfully relays data from the specified source without tampering with it or lying about its origin. As depicted in Figure A.6b,

a common design is to have a committee of multiple oracle servers fetch data independently, and use an aggregation mechanism to filter out potential malicious inputs and produce the final result. This approach guarantees authenticity, assuming that a fraction of the committee is malicious (e.g., less than one-third) [102].

Privacy-preserving oracles. In addition to authenticity, *privacy-preserving oracles* can relay information derived from private data that is not directly accessible to a regular oracle, such as a user's bank statement, credit report, or age information. As we will explore in Chapter C, privacy-preserving oracles enable ML training or fine-tuning on *private web data* obtained directly from users, without requiring a special arrangement with the original data holder. At the same time, users enjoy strong privacy protection, as they can control which data to share.

In more technical detail, a privacy-preserving oracle protocol allows a user to convince a verifier that a piece of data from a specific source S (integrity/authenticity) satisfies a certain predicate P without leaking any other information (privacy). To explain, suppose Alice wants to prove to a lender (a smart contract or an off-chain entity) that she has good credit. She could send a screenshot of her credit report, but it is easy to forge screenshots. Using a privacy-preserving oracle, Alice could cryptographically prove to an oracle that "according to data from `https://www.bigbank.com`, Alice's credit score is over 700." The oracle can verify this claim and relay the result to the lender, as shown in Figure A.7. The key privacy feature is that no information is revealed about Alice beyond the fact that the above statement is true. In particular, Alice does not need to reveal to the lender the secrets necessary to retrieve the credit record (e.g., her SSN) or any extra information that might be on the credit report (e.g., her address history).

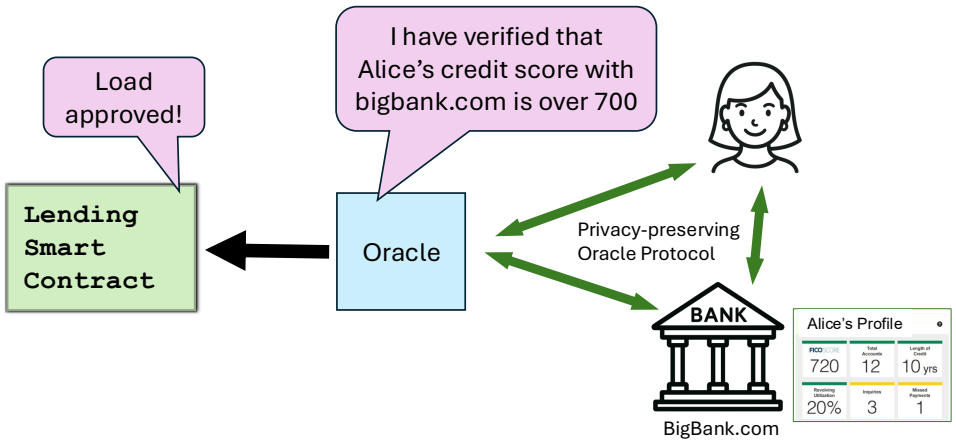


Figure A.7: User can "export" private information using a privacy-preserving oracle.

A-2.3.2 Constructions of Privacy-Preserving Oracles

We now discuss high-level technical constructions, based on the original papers that propose privacy-preserving oracles [103, 85]. Subsequent works proposed various security and performance optimizations [104], but the main idea remains the same.

The starting point for privacy-preserving oracles is the Transport Layer Security (TLS) protocol that secures the Internet. TLS is a protocol that enables a user (e.g., a browser) to establish secure connections with a remote web server. Readers do not need to understand the details of TLS, but we remark that TLS itself does not digitally sign the transmitted data. Rather, when Alice obtains a piece of ciphertext D from a TLS server S, the integrity of D is protected by a key shared between Alice and the server. As a result, a third party, such as the oracle node in our example, cannot verify the

authenticity of D or determine whether D originates from the server or is entirely forged by Alice. In all constructions of oracles below, this shared key needs to be *somehow* hidden from Alice for this reason.

There are three main approaches to constructing oracle protocols: using TEEs, using secure two-party computation (2PC), or using the oracle as a proxy. It is worth noting that TLS can be modified to sign transmitted data (as proposed in TLS-N [105]), but this approach incurs a high adoption cost, as it requires websites to upgrade to the modified TLS protocols; moreover, adding signatures only addresses authenticity, not privacy. Therefore, we focus on oracle protocols that are compatible with existing websites.

TEE-based oracles. The first class of solutions relies on TEE technology, such as Intel SGX or TDX (we refer readers to Section A-2.1 for an overview of TEEs). This approach was first presented in Town Crier [103]. Town Crier runs the following high-level logic in a TEE: first, it accepts a request specifying a data source S, a statement/predicate P to be proven against data from S, and user secrets required to access S (e.g., encrypted passwords); then it fetches data D from the given source S over TLS, and outputs $P(D)$, along with a hardware-generated attestation to the correctness of it.

Note that the above statement P can be a generic function computed by the oracle over D. This way, oracles can also serve as the execution layer for heavy off-chain computation. Compared to the designs we will introduce next, TEE-based oracles are likely the most practical solution for large-scale computation, such as AI. AI tools used by smart contracts are likely in many cases to be executed within TEE-based oracles, where models can access external data, perform non-trivial computation, and return attested results on-chain.

2PC-based oracles. DECO [85] introduced a privacy-preserving oracle design that does not depend on TEEs. The high-level workflow is shown in Figure A.8.

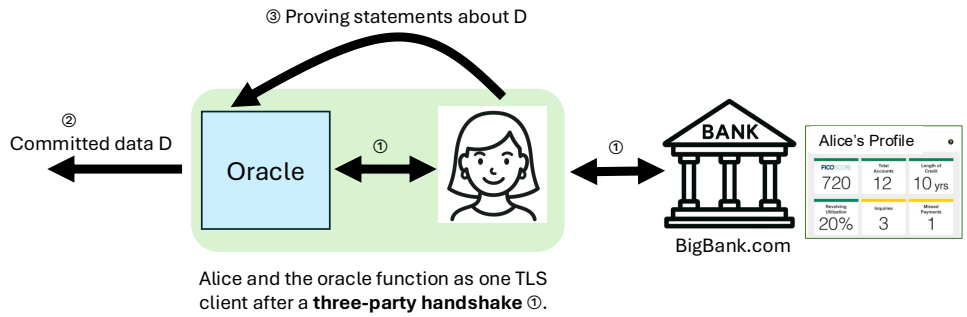


Figure A.8: The high-level workflow of DECO's 2PC mode.

The key idea in DECO is for Alice and the oracle to run a two-party computation (2PC) protocol [106] (step 1) to jointly complete the TLS handshake, so that neither of them has the entire session key. This protocol is called a *three-party handshake*, to differentiate from the standard two-party handshake in TLS. After the three-party handshake, Alice queries the web server over TLS (with the help of the oracle) and commits to the ciphertext D. The three-party handshake hides the session key, preventing Alice from forging TLS ciphertexts. Now that the authenticity of D has been established, the prover can prove fine-grained statements using any desired generic zero-knowledge proof system [107] in step 3.

Proxy-based oracles. The above two constructions use TEEs and three-party handshakes to prevent forgery of TLS ciphertext, respectively. Proxy-based oracles introduce a different method: Alice interacts with the TLS server through the oracle as a network proxy. Alice can then prove statements about the *proxied* TLS ciphertext D in a manner similar to step 3 of the 2PC-based oracles. However,

compared with 2PC-based oracles, proxy mode eliminates the expensive three-party handshake and thus is more performant and simpler to implement. Proxy mode was originally introduced in DECO [85].

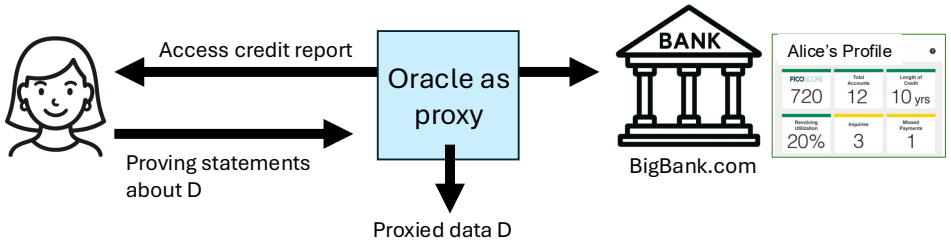


Figure A.9: The high-level workflow of DECO's proxy mode.

Summary. We now compare the security and performance of these designs. To access private data and perform large-scale computation over them, TEE-based solutions are currently the *only* practical solution (although zero-knowledge proof systems are increasingly scalable). Recent GPU TEEs, such as those with NVIDIA Confidential Computing, are particularly well-suited for ML workloads. The caveat is that TEEs introduce additional trust assumptions in the hardware manufacturers and the environments in which they operate, as we detail in Section A-2.1.2. DECO-like protocols (both 2PC and proxy modes) are more suitable for proving relatively simple statements, such as age verification [108], credit verification, and so on. Between 2PC and proxy modes, the latter avoids an expensive step; however, it is vulnerable to network-layer attacks (e.g., BGP hijacking [85, Appendix C.4]). For high-stakes applications, 2PC mode should be preferred.

All of the above designs (and their variants) have seen adoption in practice. The industry now refers to them as "zkTLS" protocols—a misnomer for TEE-based solutions, because they do not use any "zk" (zero-knowledge proofs). Town Crier and DECO have been productized as part of the Chainlink Runtime Environment (CRE) [109]. Reclaim [110] implemented an improved proxy-based variant [104], now expanding to a new TEE-based protocol. zkPass [111] runs in proxy mode by default, and falls back to 2PC mode for servers that do not support proxy mode, a feature they call hybrid mode. TLSNotary [112] is an implementation of 2PC mode by the Ethereum Foundation.

Chapter B

AI x Crypto: Crypto Applications of AI

B-1 Overview: Making Crypto More Usable and Flexible

Today, humans design and implement the foundational protocols and applications for trusted computers, such as consensus algorithms, networking algorithms, and smart contracts (Figure B.1). Once specified, the trusted computer (e.g., a blockchain) takes inputs from the environment, including human users and the outside world, e.g., via oracles. The interface between humans and blockchains requires careful thought and validation, and often humans struggle to precisely specify their intents, either as a user, a programmer, or a system designer [113, 114, 115].

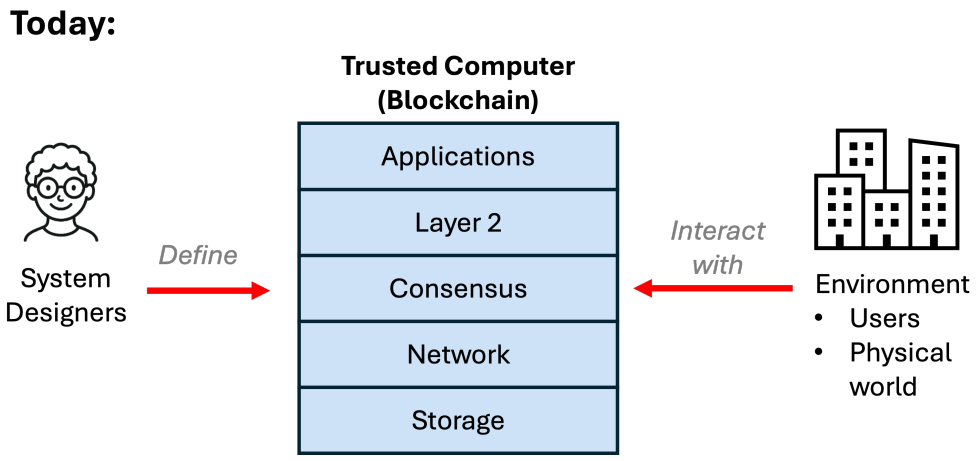


Figure B.1: Today the interface (red arrows) between the trusted computer and everything else (designers, end users) is labor-intensive and error-prone!

AI could act as a **translation layer** between the trusted computer and "everything else," including the designer and the environment, as shown in Figure B.2. For example:

- AI can help designers more flexibly specify the desiderata of a component of the blockchain stack, such as an application (e.g. DeFi incentive mechanisms).
- AI can help users of blockchains define policies and/or specify their interactions with a blockchain

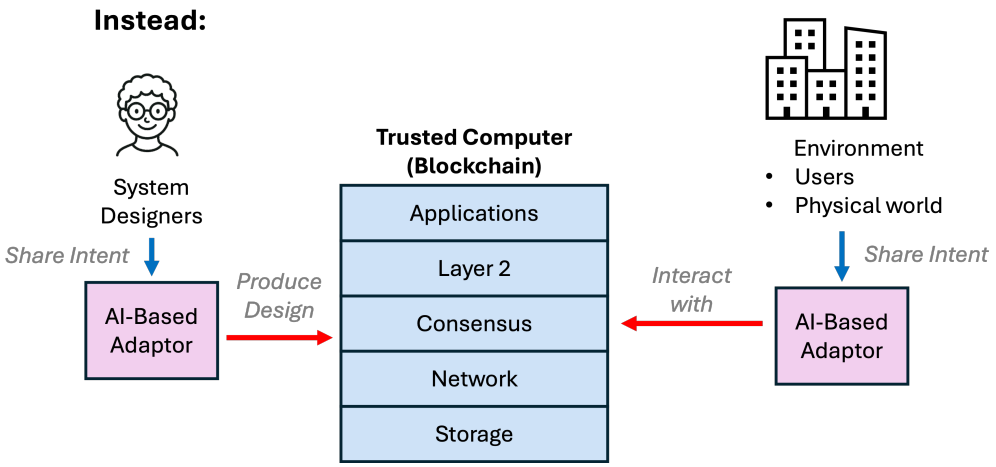


Figure B.2: AI can translate human intents (blue arrows) into machine-readable instructions in a much more flexible manner.

(e.g., convert streams of data from the real-world into smart-contract-readable data, or create and submit transactions to a smart contract that execute the user's intent).

In this chapter, we will discuss ongoing efforts to use AI to facilitate the design and analysis of blockchains in Sections B-2 to B-4, as well as some more futuristic directions in Section B-5. Researchers have been applying AI to blockchain systems (more broadly, decentralized systems) for well over a decade. While the space of algorithms and techniques is vast, we roughly categorize these efforts into three groups, which gained prominence roughly sequentially in time. These categories are illustrated in Figure B.3.

Group 1: AI-assisted analytics. (Section B-2) These are AI-based algorithms for analyzing or understanding the state of an existing blockchain. This class of techniques emerged over a decade ago: it consists of applying AI to analytics problems, such as predicting events on a blockchain or classifying fraudulent transactions. These techniques translate the state of a complex system to something interpretable by a human.

Group 2: AI-assisted design of constructive algorithms. (Section B-3) These are AI-based approaches that translate high-level human objectives (e.g., increase throughput, reduce latency, improve security) into algorithmic decisions, either during the design phase to inform system configuration, or at runtime to adapt system behavior to changing conditions. This category was popularized in the last (roughly) six years and generally uses ML to learn strategies for designing and/or interacting with crypto systems. In this category, we start to see a greater reliance on reinforcement learning (RL) and related techniques (e.g., bandits), which are used to learn policies or strategies in complex state and action spaces.

Group 3: AI-assisted interaction with the real world. (Section B-4) The third category, which emerged in recent years, explores how to use modern AI at the application layer to enhance blockchain interactions with the outside environment; this class of techniques mostly uses generative models, sometimes coupled with reinforcement learning, to achieve this enhanced functionality. Specifically, modern AI can equip smart contracts with three enhanced capabilities relative to the previous generation: *(1) Sensing*: AI can help smart contracts use unstructured data from the real world. *(2) Execution*: smart contracts can call tools and functions in the way that LLMs currently do. *(3) Decision-making*: smart contracts can act as agents, making decisions based on

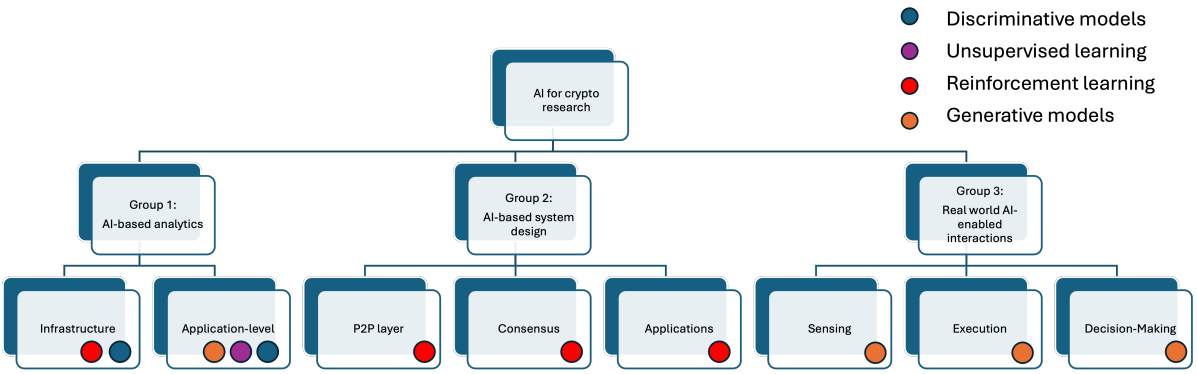


Figure B.3: We identify three main groups of AI-for-crypto research: AI-based analytics, AI-based constructive design of systems and algorithms, and AI-enhanced interactions with the real world. The colored bubbles represent common ML techniques used for various categories of research; they are not exhaustive, but represent the common patterns we have observed. Notice that more advanced ML techniques (e.g. RL and generative AI) have primarily been used in groups 2 and 3, which involve interactions with an environment, rather than the static, more traditional prediction tasks from Group 1.

values that are encoded in objective functions. This class of functionalities is enabled in part by oracles, which allow blockchains to have an accurate view of the state of the external world.

B-2 AI-Assisted Analytics [Roi][Reviewer: Giulia REVIEW COMPLETE]

We begin by exploring how AI has been used to analyze existing blockchains in the literature. We group this work into two categories: (1) analytics for global blockchain properties (Section B-2.1), and (2) analytics for object-level blockchain properties (Section B-2.2).

B-2.1 Analytics for global blockchain properties

Some categories of analytics represent properties of blockchains as a whole, i.e., they do not relate to individual users or transactions. Such analytics can relate to network-wide *protocols* (e.g., consensus algorithms), networks (e.g., P2P networks), and derived properties of a blockchain (e.g., cryptocurrency price).

B-2.1.1 Consensus analytics: Vulnerability and attack discovery

A growing body of work has used AI to discover vulnerabilities in blockchain infrastructure, particularly in consensus protocols.

Discovery of selfish mining algorithms. One common vulnerability explored is *selfish mining* [116, 117], where a blockchain operator deviates from protocol-specified behavior to unfairly gain additional rewards. For example, a selfish miner can withhold newly created blocks to build a private chain, then publish it at a strategic time when it is longer than the public chain. This causes blocks to be discarded, wasting honest miners' work, and increasing the selfish miner's relative share of the rewards.

Research on selfish mining began by characterizing the revenue of specific attacks [116, 118], primarily on proof-of-work blockchains. Later work optimized selfish mining attacks using AI-based methods [119, 120, 121]. This involved modeling all possible actions of a blockchain miner within a *Markov Decision Process* (MDP). Sapirshtein et al. [119] were the first to do so, and introduced a method

which requires solving a sequence of MDPs. Later works introduced efficient approximations of the true MDP [121] and other models for network conditions [122], proof-of-stake protocols [122], and bribery attacks [123].

Classical MDP solving techniques are limited to relatively small state spaces due to their large computational and memory requirements [124]. A body of work has employed *deep RL* techniques, which involve the use of neural networks, to analyze more complex models [125, 126, 127, 128, 129]. Hou et al. [125] employed deep RL in a setting with multiple selfish miners under various consensus rules. Other work [126, 129] used deep RL to study the effect of transaction fees on selfish mining in proof-of-work blockchains, as well as the effect of petty-compliant miners who may deviate in minor ways for profit [127]. Sarenche et al. [128] have extended the analysis of selfish mining to proof-of-stake blockchains using deep RL techniques. The use of deep RL has allowed for finding profitable selfish mining strategies in more realistic settings, which would not have been possible with traditional methods alone.

Despite their ability to analyze more complex models, deep RL techniques often lack formal guarantees on the optimality of the derived strategies. To this end, recent work used traditional RL methods to accurately analyze longest-chain protocols with alternative proof systems [130] and DAG-based protocols [131, 132]. Their approach allows an accurate characterization of the *security threshold*, the minimal power required to profit from selfish mining.

Real-time discovery of attacks on consensus. Beyond characterizing and optimizing attacks, recent work has focused on detecting ongoing attacks. Reddy and Sharma [133] detect double spending in DAG-based ledgers by using spectral clustering (an unsupervised learning method) to identify blocks produced by non-cooperating miners. Venkatesan and Rahayu [134] later propose combining hybrid consensus protocols (e.g., proof of stake and work) with ML classifiers for real-time anomaly detection, aiming to preemptively detect threats such as 51% attacks. A separate line of work has focused on detecting selfish mining using discriminative ML classifiers. Wang et al. use [135] a neural network based on fork structure. Later work has extended the analysis to use more advanced ML-techniques such as ensemble deep learning [136] and to use additional features such as transaction fees and block generation times [137]. Recent work has also studied undetectable selfish mining strategies, rendering these mechanisms ineffective and calling for more research on how to detect and mitigate such attacks [138]. Overall, the dominant approach across consensus-layer attack detection has been discriminative supervised classification on hand-crafted features, with deeper architectures and graph neural networks emerging more recently. However, progress remains constrained by reliance on simulator-generated training data and the scarcity of confirmed real-world attack samples.

B-2.1.2 P2P analytics: Attack discovery

Another line of work has used ML to detect ongoing attacks at the P2P network layer. A primary attack of interest is *eclipse attacks*, in which an attacker tries to separate one or more nodes' IP addresses from the rest of the network in order to alter and control the views of different participants in a blockchain network. These attacks can sometimes be detected by analyzing network traffic patterns and applying basic ML tecnhiques [139, 140, 141, 142]. For instance, [139] used a random forest classifier to identify attacks based on statistical features like packet size and access frequency. Later works, such as Dai et al. [141], employed more advanced ML techniques, combining convolutional neural networks with bidirectional RNNs and a cross-attention mechanism to capture both spatial and temporal patterns in the network traffic data, thereby improving detection accuracy. Despite some work in this direction, ML-based approaches (both supervised and unsupervised) are difficult to develop due to a lack of labeled data from eclipse attacks. Prior methods have handled this problem in part via synthetic data augmentation techniques, which can give small gains in detection rates [141].

B-2.1.3 Derived property analytics: Price prediction

Many papers have explored ML techniques for predicting the price of various cryptocurrencies [143, 144, 145, 146, 147, 148, 149]. Recent papers have used relatively standard ML tools such as Bayesian neural networks [144], RNNs [145, 147, 150, 149], MLPs [146], and SVMs [146, 149]. These recent papers have generally emphasized the importance of using a diverse set of input data and features, including on-chain data, data from external (non-cryptocurrency) markets, and social media platforms to understand user sentiment [145, 150, 146, 149]. However, they still have used fairly basic ML tools, making limited use of neural networks.

In contrast, industry tools are increasingly using foundation models (FMs) to address these problem. For example, ElizaOS [151] and Virtuals [7] enable users to deploy agents that make predictions and decisions based on underlying LLMs (e.g., see Section B-4.3). As these agents use general-purpose LLMs for decision-making, they are not specifically trained for price prediction.

A major difference between price prediction in cryptocurrency and traditional markets is that cryptocurrency markets still depend (relatively) more on smaller investors, particularly for nascent or volatile assets, such as memecoins. As a result, cryptocurrency prices are more impacted by—and discussed on—social media channels like Discord and Telegram, which can be mined for side information, some of it misleading [152]. In this sense, the cryptocurrency price prediction problem can benefit from a potentially richer set of side information than narrower problems like attack detection, and LLM-based predictors are well-suited to handle such side information. On the other hand, it may be possible to design better custom models to exploit this rich side information. A key research challenge is how navigate this tension.

Research Question B-2.1

How can we design ML tools to predict cryptocurrency aggregate properties (like price) that effectively incorporate rich and unstructured side information from the Internet?

To our knowledge, neither prior research papers on cryptocurrency price prediction nor industry efforts are making use of state-of-the-art time-series forecasting models [153, 154, 155]. Could such tools provide substantial gains relative to the simpler and lower-dimensional predictors that have been proposed in prior research? Could they also provide benefits over general-purpose agents seen in industry? A key research challenge associated with designing custom predictors is the rich nature of side information for cryptocurrency markets; text messages and chat signals from various platforms are semi-structured, whereas the ML community has largely developed forecasting models for fixed benchmarks over highly-structured data. Reconciling these differences may require novel architectural adaptors and/or data processing, analogous to what has been done in other domains [156]. We discuss some of these questions at a broader scale in the conclusion of this section, Section B-6.

B-2.1.4 Summary

Notice that in the papers discussed above, ML has been used for analytics in settings where we have *visibility*. That is, given a global, known consensus protocol, we can discover vulnerabilities, or given public global information about the state of the blockchain, we can detect attacks or predict price fluctuations. However, due to the decentralization of many blockchains, we may lack the visibility to run many kinds of analytics.

Key Takeaway B-2.1: ML for aggregate-level blockchain analytics

For aggregate-level blockchain analytics, ML has been applied to a limited set of problems for which we can obtain global, public observations about a system. On the other hand, fine-grained system monitoring and local analytics remain challenging due to lack of central coordination and visibility.

For example, in enterprise settings, there is growing interest in applying AI to observability [156, 157, 158]. In other words, given a complex microservices architecture, how can we efficiently collect telemetry from various components and use them to identify and resolve bottlenecks or errors? The data anlytics component of this pipeline is increasingly being handled with ML [156].

As blockchains process increasing volumes of data, observability techniques could help to streamline operations by pruning out unresponsive peers, identifying smart contracts that are resource bottlenecks, or detecting unreliable nodes; however, traditional telemetry requires special instrumentation and centralized aggregation, and is ill-suited to blockchain networks. Hence, an interesting research question is how to design observability infrastructure and associated ML-based analytics pipelines for decentralized blockchain systems.

Research Question B-2.2

How can observability tools help streamline decentralized blockchain operations? How can we design privacy-preserving telemetry and ML-based analytics for streamlining blockchain management in a decentralized fashion?

This open-ended research question requires a detailed understanding of performance bottlenecks in blockchain systems. First and foremost, it would require measurements from infrastructure providers to characterize hardware and software bottlenecks that commonly hamper performance across the blockchain stack. Based on the identified bottlenecks, the research and/or open-source community could develop standardized telemetry for blockchain systems—akin to OpenTelemetry for microservice monitoring [159]. Designing observability tools for blockchains introduces new challenges, such as monitoring consensus artifacts (e.g., block structures) from different vantage points, and adapting typical metrics and tracing techniques to blockchain infrastructure. For example, in a typical microservice architecture, each user request spawns a *trace*, which records how requests flow between services. However, in a decentralized blockchain, each request to a smart contract can spawn calls to different smart contracts, each of which is executed in parallel on different validators' hardware. Hence, there are correlated but different traces (in terms of timing) occurring at each validator. Broadly, designing efficient data representations for blockchain traces, logs, and metrics could be a concrete research challenge. Another may be collecting telemetry in a privacy-preserving manner that does not leak details about individual validators' proprietary algorithms and infrastructure configurations, as well as private user transaction patterns.

B-2.2 Analytics for local blockchain objects

Next, we narrow our focus from aggregate analytics to ML-based analytics for *individual* objects or artifacts at the application layer—namely, transactions or smart contracts. We categorize these methods as follows: (1) smart contract security analysis, (2) smart contract economic analysis, (3) transaction-level deanonymization, and (4) transaction-level fraud analysis.

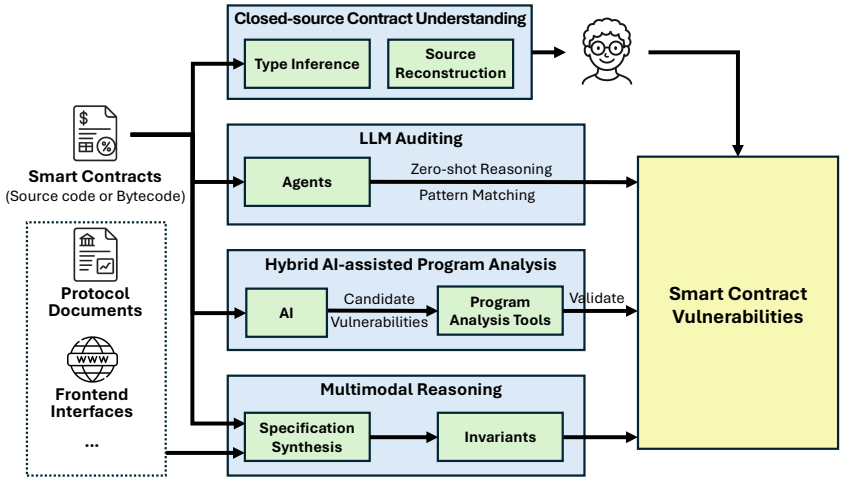


Figure B.4: Four paradigms of AI-assisted smart contract analysis, which differ in how AI is used in the analysis pipeline and the sources of information they incorporate. Existing work can be broadly categorized into: (i) closed-source contract understanding, where AI assists reverse engineering of bytecode; (ii) LLM-based auditing, where LLM agents attempt to emulate human security auditors; (iii) hybrid AI-assisted program analysis, where AI proposes candidate vulnerabilities that are validated using traditional program analysis tools; and (iv) multimodal reasoning, which integrates additional information sources such as protocol documentation and frontend interfaces to verify contract behavior and detect semantic inconsistencies.

B-2.2.1 Smart contract security analysis

Smart contract security is a critical pillar of blockchain integrity [160, 161, 162, 163, 164, 165], as vulnerabilities in deployed contracts can directly lead to financial losses. Traditionally, vulnerability detection relies on static analysis [161, 166, 167], symbolic execution [168, 169, 170], and fuzzing [171, 172, 173]. While effective for detecting well-defined vulnerability patterns, these techniques often struggle with complex contract logic and cross-contract interactions, motivating the use of AI-assisted approaches to enable richer semantic reasoning. As summarized in Figure B.4, we categorize existing AI-assisted approaches into four paradigms: closed-source contract understanding, LLM-based auditing, hybrid AI-assisted program analysis, and multimodal reasoning.

Closed-source contract understanding. The first paradigm of AI-assisted smart contract analysis aims to recover human-interpretable semantics from deployed bytecode, typically through decompilation, enabling downstream tasks such as human auditing. However, traditional decompilers [174, 166, 175] often leave a semantic gap, producing low-level logic that is difficult for auditors to interpret. To address this gap, AI-based decompilation has evolved from attribute-specific inference to end-to-end source reconstruction. Early approaches focused on specific metadata recovery; for instance, DeepInfer [176] leverages Bi-LSTM and attention mechanisms to infer function types from bytecode access patterns. SmartHalo [177] follows a neuro-symbolic approach, combining static dependency analysis with LLMs to recover high-level attributes, while preserving functional correctness through symbolic verification. More recent work explores direct bytecode-to-source translation through two main paradigms: prompt engineering and fine-tuning. Representing the former, DiSCo [178] adopts a training-free approach that uses semantic units to exploit the zero-shot capabilities of general-purpose LLMs. In contrast, David et al. [179] apply Low-Rank Adaptation-based fine-tuning on large-scale datasets that map three-address code to Solidity, enabling the model to match human programming styles better.

LLM-based auditing of contracts. The second paradigm of work explores the use of LLMs as automated auditors that directly analyze contract code. This approach leverages the zero-shot reasoning capabilities of general-purpose models to mimic human auditing processes. Recent empirical studies [180, 181] have revealed a mixed picture: while LLMs demonstrate measurable capability in identifying vulnerability-related patterns (e.g., David et al. [180] show that both GPT-4 and Claude correctly identify the vulnerability type in 40% of 52 previously exploited DeFi smart contracts), this comes at the cost of non-trivial false positives due to hallucinations [182, 183].

Hybrid AI-assisted program analysis. To address the high false positive rates of pure LLMs, the third paradigm has adopted hybrid architectures that couple the semantic understanding of AI with the rigorous validation of traditional program analysis [184, 185, 186]. GPTScan [185], for example, utilizes GPT to identify potential vulnerability candidates and then employs static program analysis, including data-flow and control-flow validation, to confirm the feasibility of the detected vulnerabilities.

Multimodal reasoning. While the above approaches primarily operate at the code level, the fourth paradigm moves beyond pure program analysis and shifts vulnerability detection toward protocol-level semantics and multimodal reasoning. Representative examples of this paradigm include DeFi-Aligner [187], which uses LLMs to extract business logic from project documentation and aligns it with symbolic code summaries to identify semantic deviations, and Hyperion [188], which analyzes DApp front-end interfaces using a fine-tuned LLaMA2 model to uncover inconsistencies between user-facing promises and on-chain execution. PropertyGPT [189] leverages retrieval-augmented generation to synthesize compilable specifications from audit reports, while SmartInv [190] employs a "Tier of Thought" strategy to infer critical invariants from multimodal sources, including code and natural-language comments.

Taken together, these paradigms suggest a clear trend of leveraging AI to reduce human auditing effort while expanding the scope of vulnerability discovery and maintaining accurate detection.

Key Takeaway B-2.2

State-of-the-art methods for detecting security flaws in smart contracts do not purely rely on AI-based predictions from input features; instead, they identify constraints or invariants that must hold, and combine these with ML models to detect vulnerabilities.

Many papers today are combining deterministic program analysis and/or domain-informed semantic analysis with ML tools for vulnerability detection. This is a powerful paradigm, but it begs the question of how best to extract and use such domain-specific knowledge, as highlighted in the following research question.

Research Question B-2.3

What level of domain knowledge should be integrated into smart contract security analysis pipelines, and how should that couple with AI-based techniques? What information sources, including multimodal data, should these AI- assisted systems leverage?

Prior work has explored various operating points in terms of extracting domain-specific properties that should hold in smart contracts, and coupling these with AI. However, it remains unclear what architecture is best under what conditions, and what role AI should play. Should it be used purely

as a filter, as in GPTScan? Should it be used to identify policies and invariants that smart contracts should satisfy? If the latter, what kinds of inputs are required, including documentation types and level of detail? While there exist various proofs-of-concept for different architectures, we lack a systematic exploration comparing different architectures and information sources. Such an exploration could help to inform even stronger architectures for smart contract vulnerability analysis.

B-2.2.2 Smart contract economic analysis

AI is used not only to analyze security properties of smart contracts, but also to study the economic behaviors generated by smart contract interactions. In the current literature, this line of research is largely centered on maximal extractable value (MEV), namely profit obtained through strategic transaction ordering during block construction [191]. Among the most commonly-studied MEV activities are arbitrage [192, 193], sandwich attacks [194], and liquidations [195]. Existing work in this area mainly falls into two categories: MEV discovery and MEV activity detection; we discuss ML-assisted discovery of bidding strategies in Section B-3.3.3, which is more of a constructive design problem than an analytics problem.

MEV discovery. First, AI can help discover MEV opportunities by searching over transaction sequences and execution environments. Earlier work in this direction mainly relied on non-learning-based techniques, including constraint-based search over DeFi actions [192], formal verification of composed DeFi contracts [196], and heuristic-driven analyses of arbitrage opportunities [193]. In contrast to these methods, Lanturn [197] formulates MEV extraction as an adaptive learning-based black-box optimization problem, aiming to synthesize profit-maximizing transaction sequences rather than merely detect pre-specified arbitrage patterns. MEVisor [198] complements this line of work by emphasizing high-throughput opportunity search through parallel genetic algorithms and GPU-accelerated execution.

MEV activity detection. Second, AI can be used to detect realized MEV activity, that is, to automatically identify and classify MEV-related behaviors from on-chain or bundle-level data. Early approaches primarily relied on heuristic rules to detect MEV activities [199, 200, 201]. While effective in specific settings, these approaches depend heavily on hand-crafted patterns and are therefore limited in automation, scalability, and adaptability to new MEV strategies. Recent work has therefore begun to introduce AI-based methods for MEV activity detection. A representative example is the combined ActLifter–ActCluster pipeline [202], which first lifts raw transaction traces into semantic DeFi actions, then performs bundle representation learning to encode raw bundles as low-dimensional feature vectors, and finally applies iterative clustering to identify both known and previously unseen MEV activity patterns.

Key Takeaway B-2.3: Trends in smart contract economic analysis

AI has shifted smart contract economic analysis away from hand-crafted patterns and static heuristics toward more automated methods that can adapt to complex behaviors with less task-specific prior knowledge. This opens the door to exploiting subtle structures and misalignments in smart contracts that may not be intuitively obvious to humans.

B-2.2.3 Transaction-level deanonymization

Cryptocurrency deanonymization refers to identifying the source of a given transaction, as defined either by their real-world identity or other network identifiers like an IP address. ML-based deanonymization typically arises through one of a few mechanism classes: (1) interactions with centralized third-party services, (2) behavioral and on-chain pattern analysis, and (3) network layer and related side channels.

Most *public* research on AI-based methods for deanonymization stems from the 2nd and 3rd categories, as we summarize below.

Behavioral and on-chain pattern analysis. Behavioral and on-chain pattern analysis facilitates deanonymization without the need for direct involvement from third parties. Common techniques include address clustering, as well as the analysis of transaction amounts, frequencies, payment habits, address reuse, and timing patterns. These heuristics leverage the blockchain's inherent transparency to uncover relationships between addresses, and, in some cases, to link them to real-world individuals. For instance, several works have used manual graph representation learning techniques to categorize nodes [203, 204]. Moving towards more automated feature extraction methods, many papers use graph neural networks to learn embeddings from transaction graphs for account deanonymization [205, 206, 207, 208, 209, 210, 211, 212, 213, 214]. For example, Huang *et al.* combine hierarchical self-attention modules to model local node structure on the transaction graph, and large-scale relations across the graph in their architecture [214]. Hu *et al.* instead use a sequential modeling of transactions; their BERT4ETH model is an encoder-style transformer pretrained on sequences of Ethereum transactions [212]. It is used to extract embeddings for Ethereum accounts for deanonymization and anomaly detection (see Section B-2.2.4 below). To our knowledge, there is no head-to-head comparison between the sequential modeling approach of BERT4ETH [212] and recent models that explicitly use the graph structure, like [207, 213, 214]. Hence, it is unclear which architecture works best, though the recent trend has clearly favored graph-aware modeling; that is, self-attention may be used, but in a way that respects the topology of the underlying graph.

Network layer and side channels. Network-layer and side-channel attacks exploit the dynamics of the peer-to-peer infrastructure to infer a transaction source's IP address. An adversary who controls a significant fraction of nodes can monitor transaction propagation patterns, infer the originating node through timing analysis or distinctive relay behavior, and collect off-chain metadata, such as IP addresses exposed during message broadcast, device fingerprints, or correlations with external events [215, 216, 217, 218]. These methods have mostly not used data-driven ML predictors, focusing instead on the design and analysis of statistical source predictors based on propagation dynamics [217, 218].

B-2.2.4 Transaction-level fraud analysis

Complementary to analytics for source deanonymization, another important class of analytics aims to detect fraudulent or anomalous crypto transactions [219], including for applications in Anti-Money Laundering (AML) and Countering the Financing of Terror (CFT). Historically, rule-based detection systems were widely used for fraud detection—both in blockchains and traditional finance [219]. However, these methods fall short in dynamic and complex environments, often producing high volumes of false positives that carry substantial operational and compliance costs [220, 219]. For example, Project Aurora [221] assessed various strategies for fraud detection, including existing rule-based models, logistic regression, neural networks, and graph neural networks, under realistic real-world conditions and constraints. They found that graph neural networks were most effective in detecting money laundering in a synthetic cross-border transaction dataset.

Data-driven, AI-assisted fraud detection algorithms can use data from multiple layers of the blockchain stack and off-chain sources. Common data sources include application-level traces (e.g., transactions within an application, including timing data), off-chain marketing data, such as social media, and off-chain secondary market data (e.g., price fluctuations). We summarize detection techniques based on the type of input data.

Transaction (sequence) metadata. Many ML tools have been designed to detect illicit activities and scams using transaction- or sequence-level characteristics, such as the timing between transactions from an account, the type of transactions in an application, transaction amounts, involved parties, and

destinations. These features were then used to train basic predictive models [222, 223, 224, 225, 226, 212, 227]. BlockGPT [224] and BERT4ETH [212], for instance, use serialized Ethereum transactions to train transformers, which can be used to extract embeddings for downstream classification tasks.

Topological data. Another important class of methods explicitly models topological data, either derived from measurements from the P2P network [228] or by observing the logical transaction graph [229, 230, 231, 232, 233]. For example, BitcoinHeist extracts custom features from the Bitcoin transaction and address graphs to identify ransomware [230], and similar graph-based features (e.g., flow patterns, mixing behaviors, and layering structures) have been proposed for AML/CFT risk scoring and suspicious activity detection [234, 235, 236, 237]. Graph neural networks, including graph convolutional networks (GCNs), have also shown strong promise for anomaly and fraud detection in blockchain transaction data [221, 231, 238, 239]. In addition to the findings from Project Aurora [221], Patel *et al.* proposed EvAnGCN, which applies a dynamic GCN to the induced and evolving transaction graph of the Ethereum blockchain to effectively identify anomalous behaviors [231]. More recently, Pocher *et al.* demonstrated that standard GCNs outperform traditional methods in detecting anomalous cryptocurrency transactions [238].

Off-chain data. Increasingly, researchers are also leveraging off-chain data to detect anomalous, fraudulent, or illicit cryptocurrency activity [240]. Several studies have focused on identifying money-laundering transactions linked to darknet markets [241, 242]. Others turn to social media and secondary online sources. For instance, CryptoScamHunter employs natural language processing to analyze YouTube video titles and descriptions, enabling the detection of DEX arbitrage bot scams [243]. Similarly, Huang *et al.* combine secondary market data with on-chain trading activity to identify NFT rug-pull scams [223]. Beyond content analysis, Wang *et al.* examine browser extension reviews together with programmatic features to predict whether a crypto-related browser extension is malicious [244]. Several works have applied NLP techniques to social media content to uncover cryptocurrency pump-and-dump schemes [245, 246, 247, 248].

Key Takeaway B-2.4: AI for transaction-level analytics

Current state-of-the-art methods for transaction-level analytics—including for fraud detection and deanonymization—make heavy use of transaction graph characteristics and metadata. These inputs are used to extract neural representations that capture graph dependencies, either explicitly (via a GNN) or implicitly (via a transformer trained over curated transaction sequences).

Of course, strong detectors may also use a broad array of structured and unstructured auxiliary inputs that are not graph structured. An important question is how best to design ML algorithms that can effectively use such auxiliary inputs.

B-3 AI-Assisted Design of Constructive Algorithms [Roi][Reviewer: Christian]

The previous category of work (Section B-2) focuses on AI-assisted analytics: tools for *understanding* the state of existing crypto applications and networks. Since that category of research took off over a decade ago, the research community has also turned to AI to help *design* decentralized algorithms, ranging from peer-to-peer networks and blockchain protocols to applications and markets. These techniques typically rely on reinforcement learning and related ML techniques to drive the design of algorithms—both for the core infrastructure, the underlying consensus protocols, and strategic application algorithms. We

categorize these approaches by the layer of the blockchain stack at which they operate: the peer-to-peer network (P2P; Section B-3.1), consensus layer (Section B-3.2), and application layer (Section B-3.3).

B-3.1 Peer-to-peer protocols

AI has been used to design algorithms that help form and maintain the peer-to-peer message transport layer in blockchain networks. In a related development, application-layer networks that maintain links among specific peers have also benefited from the use of AI. The latter concerns Layer-2 solutions, such as payment channel networks.

Peer-to-peer network Several algorithms have been proposed in which individual network nodes make AI-informed local changes (e.g., rewiring peer connections, updating the communication profile) to improve local and global network properties, such as average latency or total communication volume [249, 250, 251, 250, 252]. For example, Topiary [251] explicitly formulates the network topology formation as a multi-armed bandit (MAB) problem and designs an algorithm to update each node's peer connections, dropping peers with high relative latency of incoming messages. Valko and Kudenko [252] instead employ an RL agent that re-orders the broadcast queue to neighbors to reduce both propagation time and the total number of messages sent, thereby lowering the energy footprint of the network.

Similar ideas have been applied in domain-specific blockchain networks, such as vehicular networks and wireless networks. In vehicular networks, nodes frequently join and leave, leading to much higher peer churn than in cryptocurrency networks. Kim and Ibrahim [253] model the selection of the number of peers as a contextual MAB problem, dynamically adjusting channel size to maintain Byzantine fault tolerance under frequent vehicle churn. Saadat et al. [254] further use ML to predict node stability in cluster-based vehicular networks, selecting consensus nodes that are unlikely to leave mid-process. In wireless networks, high transmission rates result in high energy consumption. To that end, Ju et al. [255] use GCNs to determine each node's data transmission rate, with the objective of balancing reliable communication with low energy consumption.

Overall, AI-assisted methods for the design and management of P2P networks remain relatively underexplored compared to other uses of AI in the blockchain context. One interesting question is how to combine AI-assisted P2P network management with methods for detecting ongoing network attacks in Section B-2.1.2.

Research Question B-3.1

How can one combine techniques for detecting eclipse attacks (Section B-2.1.2) with algorithms for P2P network management and optimization? When do existing P2P network management algorithms provide robustness under adversarial network conditions (and how should we model such adversarial conditions)?

Layer-2 networks and interchain communication Layer-2 networks that rely on a blockchain consensus layer (i.e., their Layer 1) also use a P2P structure, but they address a different set of challenges than those faced by P2P networks for message propagation. A prominent example are payment-channel networks (PCNs) like the Lightning Network that is attached to Bitcoin. The connections in a PCN represent logical channels over which money can be routed for peer-to-peer transfers rather than communication channels. Connections therefore represent a form of mutual trust and guarantee that participants can carry out transfers without resorting to the Layer-1 network. Despite these differences, both types of networks pose similar challenges related to network formation and routing: What network topology should one form? How should one manage inactive peers? How should transactions be routed over the network? Several of these challenges have been tackled with RL-based algorithms, as in P2P

networks. For example, RL-based mechanisms have been used to set channel parameters (e.g., fees) to maximize an operator's profits [256, 257, 258], select payment routes over the network to minimize cost to transaction senders [259, 260, 252, 261], and rebalance channels to improve network throughput [262, 263]. Among these methods, similar RL algorithms have been used for routing management [252, 261]; however, to our knowledge, there has been limited work on using RL to determine how to make and break channels in a PCN (whereas this is a small but established research area in the P2P literature). This may be an interesting question for future work.

Research Question B-3.2

How can PCN nodes use RL methods to determine which edges to form and break dynamically over time? How would such algorithms impact the node's own rewards, as well as overall network health?

B-3.2 Consensus protocols

AI has been used extensively at the consensus layer, both to enhance existing consensus protocols and operations and as an ingredient in completely new protocols.

B-3.2.1 Enhancing performance

Consensus protocols were originally designed for static environments, but they often operate under dynamic conditions in practice, where the network conditions and participants may vary. In recent years, a growing body of work has turned to ML techniques to enhance the performance of consensus protocols (i.e., increase throughput and/or reduce latency) by being more responsive to changing conditions. One can distinguish three related areas, where AI has been used to enhance consensus.

Protocol selection. In a first line of work, consensus protocols have been *selected* with the help of AI. The the intended deployment network, the available synchronization features, and the expected connectivity influence the choice of a consensus protocol. No single consensus protocol dominates across all operating conditions: Bitcoin's PoW consensus has high latency and is optimized for maintaining safety in a loosely synchronized network, with time constants on the order of minutes; the PoS protocols of Ethereum and Cardano, in contrast, assume much closer synchronization, on the order of 10–20 seconds, and more uniform reachability.

It has therefore been suggested to build systems that switch between protocols during operation, most often in the sense that they run a fast and fragile protocol optimistically and switch to a slower robust protocol when they detect problems with the optimistic path. A classic work exploring this idea proposed to build the "next 700 BFT protocols" in this way [264]. AI-based methods have been used to guide the choice of protocols. Whereas Liu et al. [265] develop a system that statically selects a consensus algorithm using deep reinforcement learning, the recent BFTBrain work [266] selects dynamically among consensus protocols such as PBFT [267], HotStuff [268], and Zyzzyva [269]. It does so by treating protocol choice as a contextual MAB problem and addressing it with a RL engine built into the overall system.

Relying on automation and on AI to speed up consensus in good cases, however, opens up potential avenues for attack that can be exploited by bad actors. Classic results in this area demonstrate that operating with the wrong protocol for the given environment can reduce performance to zero [270, 271]. ML-based protocol-selection methods therefore need to be robust.

Parameter selection. In a second area of consensus enhancements, AI has been used to select the *parameters* of consensus protocols. The underlying reason is that the performance of a network

depends heavily on configuration parameters such as timeouts, block size, and block interval and on the available connectivity among active operators. Manual tuning is impractical in environments where optimal values shift as the workload fluctuates. A line of work focuses on the optimization of block parameters. Monem et al. [272] apply XGBoost to predict future transaction volumes and dynamically adjust the block size to match expected demand. Other works [265, 273] tune both the block size and interval using deep reinforcement learning. Zhai et al. [273] attempt to make these choices more explainable by using structural causal models to provide justifications for chosen values. Dutta et al. [274] take a different approach by focusing on the timing of block creation. They examine the use of reinforcement learning by an operator to learn when to seal a block so that transaction confirmation times are minimized.

Consensus participant selection. As a third and final area within consensus protocols, AI has been applied to select *participants* for particular roles in a consensus protocol. As most consensus protocols deployed in practice rely on a leader, measuring the performance of nodes and choosing well-connected and alive nodes as leaders generally speeds up protocol operation. However, care must be taken to not degrade performance, should the measurements turn out to be wrong.

Recent research [275, 276] shows performance of protocols benefits greatly from carefully selecting the consensus leaders, even without AI-specific selection heuristics. The protocol of Islam et al. [277] selects leaders in Proof-of-Stake consensus using a multi-agent RL approach. It collects various performance metrics about potential leaders from all validators and aims to identify and exclude malicious validators through a penalty-reward mechanism. Nour et al. [278, 279] study the application of AI in DAG-based BFT protocols such as Narwhal [280] and Bullshark [281]. They use graph neural networks to rank blocks and select leaders within the DAG, reducing latency and improving throughput without affecting the soundness of the underlying consensus protocol. Many further authors have used RL and related methods in a similar vein to dynamically assign scores to validator nodes and assign particular protocol roles based on such scores [282, 283, 284]. A recent survey of the topic [285] collects many techniques, but mostly demonstrates that this field is still in its infancy because of multiple open problems with the use of automated reasoning. In particular, the survey highlights the potential for adversarial attacks that are enabled through AI.

Research Question B-3.3

How robust are ML-based methods that operate within consensus protocols and guide parameter and participant selection against manipulation by adversarial insider nodes? How can a decentralized consensus protocol come up with a common and trustworthy estimate of the trustworthiness of its participating nodes?

B-3.2.2 Sharding

Deep RL has also been applied to the design of blockchain sharding algorithms, addressing two complementary problems: shard configuration optimization and cross-shard transaction reduction through data placement.

The earliest efforts used deep reinforcement learning, and Deep Q-Networks (DQN) in particular, to choose how many shards to create and how to size their blocks, framing these recurring decisions as a sequential optimization problem. SkyChain [286] used DQN to continuously vary parameter configuration while ensuring shards can efficiently merge and split when needed. A concurrent work [287] shows that a DQN agent can learn to maximize throughput while respecting a security constraint derived from the estimated performance and detectable misbehavior of other nodes in the network. A representative recent work that demonstrates this approach is TbDd [288], a trust-based DRL-driven sharding

framework: it gathers feedback on performance and historical behavior of all nodes and assigns roles accordingly.

A second line of work assigns data and accounts to shards to minimize costly cross-shard transactions. TbDd [288] implements this as well, by observing data-access patterns and assigning the data items to suitable shards so as to bound the number of cross-shard transactions. Wang et al. [289] use generative AI in this context to predict future cross-shard interactions in order to proactively assign nodes to shards. SPRING [290] employs deep RL to migrate accounts among shards by exploiting spatial-temporal transaction patterns. AERO [291] extends this approach by batching migration decisions, shrinking the action space and allowing better scalability. As dynamic sharding is currently not deployed on any live blockchain system to our knowledge, these studies represent exploratory research so far. If sharding is ever employed on a broad scale, ML-guided sharding and data placement algorithms will undoubtedly be taken up again and potentially also deployed.

Research Question B-3.4

How can AI and ML help with analyzing data dependencies on cryptocurrencies or general-purpose blockchains, without observing a particular transaction workload? Which methods will result in automated data placement algorithms for sharded blockchains that reduce the dependencies across multiple shards and improve performance?

B-3.2.3 Trust models

The huge energy cost and investment of seemingly useless computation in Proof-of-Work (PoW) consensus has resulted in a long-standing quest for *useful* work to be employed within blockchain consensus, particularly work related to AI because of its high computational burden. However, this has not been easy so far and no satisfying solutions exist. Dotan and Tochner [292] formally survey this field and derive constraints on systems that rely on wasteless PoW. They show that, under realistic assumptions, the set of allowed problems must still involve elements of cryptographic hardness in order to keep the protocol secure and efficient.

Recent studies are more positive: Komargodski, Weinstein et al. [293, 294] study the economic validity and equilibrium dynamics of consensus with external rewards. Multiplying large matrices receives particular attention as a promising problem, on which a useful computational puzzle may be based [294]. Their work opens a path towards using AI training and inference workloads for consensus; however, it is too early to assess the empirical security of such blockchain networks built on consensus from useful work.

B-3.2.4 Summary

Consensus protocols stand at the heart of all security mechanisms in blockchains: they ensure agreement on a state and common actions based on the inputs from and correct behavior of many participating nodes. The tradeoff between their security and performance has therefore received a lot of attention. Consequently, there is a large body of work on optimizing parameters of various consensus mechanisms, some of it also using AI. Current research suggests that using ML for consensus protocols holds many promises. But since AI-based protocol optimization methods have not been widely deployed, their performance and resilience to attacks when exposed to real-world networks remains so far open.

For the design of the core consensus protocols, based on the theory of distributed computing, AI has played a minor role so far. One could argue that automated discovery of secure and performant protocols will become feasible, similar to existing automation in cryptographic protocol research [295].

Research Question B-3.5

How can we use AI to propose novel consensus mechanisms that optimize performance, in terms of communication and latency, while also organically adapting to dynamic environments and remaining secure? Can AI help us to design and analyze the security of such new protocols?

B-3.3 Design of applications

A number of applications use AI as a core component of their design.

B-3.3.1 DeFi market design

AI has been successfully used to design DeFi applications, including automated market makers (AMMs). For example, ZeroSwap uses a DQN approach to change the price of an asset over time [296], and Moszczynski integrates AMMs with frequent batch auctions, using RL to optimize pricing rules between batches [297]. Related techniques have also been applied to lending markets like Aave [298], Morpho [299] and Euler [300]: in these markets, interest rates impact the utility and liquidity of the market. Several papers have applied stochastic control theory to set such interest rates [301, 302, 303, 304], sometimes in conjunction with deep learning techniques [304]. Chitra recently showed how to use online learning to set interest rates in a regret-optimal manner [305].

B-3.3.2 AI-enhanced smart contract security

Beyond protocol-level mechanism design, AI has also been applied to smart contract security, particularly to automate tasks such as generating exploits or patches for vulnerable contracts. Prior work on automated exploit generation relies on manually crafted templates or symbolic analysis to synthesize attack sequences that satisfy specific vulnerability conditions [168, 169, 170]. AI-based approaches improve exploit generation by enabling semantic reasoning and adaptive search, allowing systems to construct exploits beyond predefined templates with minimal human involvement. For example, AdvSCanner [306] combines LLMs with static analysis to generate adversarial contracts that specifically exploit reentrancy vulnerabilities. Scaling this toward autonomy, the system A1 [307] employs an agentic workflow with execution-based validation, using domain-specific tools to achieve autonomous discovery and validate profitable exploits on real-world blockchain states. Similarly, PoCo [308] introduces an agentic framework that transforms natural-language audit reports into executable Foundry proofs of concept through a Reason-Act-Observe cycle.

Complementary to automated exploit generation, recent work also explores AI-driven automated program repair for smart contracts, aiming to close the loop from vulnerability discovery to mitigation. While traditional repair largely relies on rigid templates [309, 310, 311], AI has empowered these frameworks to become more adaptive and context-aware. Initial AI-enhanced approaches, such as SmartFix [312], enhance the traditional "generate-and-verify" paradigm by using statistical models to prioritize patch candidates; sGuard+ [313] introduces machine learning classifiers to guide rule-based repair, selecting the most appropriate rules to avoid over-patching. Going beyond static templates, recent work leverages the semantic comprehension and generative reasoning capabilities of LLMs for end-to-end generative repair [314, 315].

B-3.3.3 Algorithms for MEV extraction

AI is starting to be used to optimize bidding strategies in MEV auctions once an MEV opportunity has been identified. To realize potential profits, the target transaction must still be included on chain, but in practice there are often many competitors for the same opportunity [191]. Earlier competition often took place in public priority gas auctions, whereas Flashbots-style private channels transformed

this process into sealed-bid first-price auctions [316, 317]. This makes bidding itself a learning problem: the goal is to predict competitive bids and choose bribes that maximize expected profit conditional on inclusion. For example, Raun et al. analyze Flashbots auction data and train machine learning models (namely a Light Gradient Boosted Machine regressor) to predict winning bribe ratios in MEV auctions, showing that learning-based bidding strategies can improve profitability in arbitrage MEV auctions [318]. Lanturn [197] instead optimizes a black-box reward function using recent adaptive sampling techniques to maximize extractable value [319]. Other work has explored predicting auxiliary variables like the length of a FlashBots auction and the maximal bid value using simple regression models such as random forests and decision trees [320].

B-4 AI-Enhanced Interactions with the Real World [Fan,Ari][Reviewer: James C. REVIEW COMPLETE]

[James C: Organizational: I am unsure about the categorization of blockchain interactions into (1) sensing (2) execution and (3) decision-making. It seems that *execution* is about *how* AI can achieve interactions with blockchains in a secure and private manner, whereas *sensing* and *decision-making* describe *types* of interactions, which both require secure AI execution. I perceive Section B-4.2 as a summary and reference to Section C-4 (blockchain for AI integrity), which is helpful for the reader here, but perhaps orthogonal to sensing/decision-making.]

Efforts have emerged in the last few years to use AI to enhance the interactions of smart contracts with the real world. These efforts have seen deployment, but are best characterized as emerging.

In this section, we discuss uses of AI that enhance three main types of blockchain interactions with the outside world:

1. *Sensing:* (Section B-4.1) AI can help smart contracts understand the state of the world in more complex ways than what is currently possible, e.g. by digesting and processing unstructured data streams.
2. *Execution:* (Section B-4.2) AI can extend the ways in which smart contracts can impact the external world, e.g. by using the tool-calling capabilities of AI models.
3. *Decision-Making:* (Section B-4.3) AI can help smart contracts enact more sophisticated decision-making pipelines, e.g., by relying on interactions between autonomous agents.

In all three of these areas, we discuss both opportunities and challenges.

B-4.1 Sensing: Enabling smart contracts to understand natural language

In their initial form, smart contracts were designed to operate only on on-chain data. Blockchains overcame this limitation with the emergence of oracles (Section A-2.3)—systems that connect smart contracts to off-chain data and off-chain computation resources. However, most existing oracles today are limited to relaying clean, well-structured data from APIs. This limits oracles' reach to the world where there is no clear API data. As a result, for example, smart contracts cannot understand or interpret human language or institutions, a key obstacle to their ability to represent written contracts.

AI has the potential to significantly expand the kinds of data that are accessible to smart contracts. For instance, oracle systems could verifiably use AI tools to translate loosely-structured data into a format readable by smart contracts—thus acting as middleware between the broader Internet and individual smart contracts. For example, today, the only type of insurance smart contract is simple parametric insurance, where payouts are triggered when pre-agreed events occur (such as AXA's discontinued flight delay insurance, Fizzy [90]). With LLM-powered oracles, one could imagine insurance contracts that ingest and reason over richer evidence, such as claims narratives, police reports, or inspection reports from repair shops. Another example is prediction markets, which rely on oracles to determine the outcome of a question. Existing systems such as Polymarket rely on human proposers to

answer questions where there is no clear API data, and human dispute resolution to resolve disagreements [321]. AI-powered oracles have the potential to automate this process, avoiding human errors and reducing the latency and costs associated with disputes.

B-4.1.1 Risks of LLM errors

Despite this promise, a central challenge in deploying LLM-powered oracles—and in the use of LLMs in critical systems in general—is the possibility of errors, such as reasoning flaws, hallucinations, and arithmetical errors [322]. Here, we focus on errors arising from AI's inherent limitations, assuming no adversarial manipulation. Protecting the AI oracle pipeline from attacks is crucial and orthogonal.

Two recent studies [323, 324] on LLMs' accuracy in answering questions on Polymarket shed light on AI's current ability as an oracle. In an empirical study from Chainlink Labs [323], the authors used GPT-4o to resolve 1,660 markets on Polymarket and compare the LLM's answers with the ground truth. GPT-4o achieves an overall accuracy of 89.3%. This result is corroborated by a similar experiment by UMA [324], where their Truth Bot achieved an accuracy of 75% (though a good portion was due to answers being submitted before the deadline). To put the numbers in context, the overall accuracy of human-provided answers to UMA's optimistic oracle is 98.2%. Thus, for this specific task, LLMs still make significantly more errors than humans, suggesting that guardrails and oversight are necessary.

The accuracy of LLMs varies across contexts. They perform quite well when the information to be extracted is discrete, and there are official sources of truth. A canonical example is sports outcomes. In Chainlink Labs' study, 99.7% of questions about sports outcomes were correctly answered. In UMA's report, the Truth Bot achieved a 99.3% accuracy on sports and asset pricing markets (the latter is answered by calling specific APIs).

For less straightforward questions, error rates can be much higher. For example, LLMs struggle to answer questions that involve time (e.g., "Was the interest rate hike announced before or after the GDP report?") or when substantial efforts are required to extract the answer (e.g., answering "How many times will Trump say the word 'steel mill' at the Pittsburgh rally on May 30?" requires transcribing the video and count words, a task that cannot be accurately done with automated tools).

It is worth exploring ways to reduce error rates. E.g., UMA proposed to use AI agents to look for known patterns of hallucination (e.g., Perplexity making inferences rather than being fact-driven) [324], although recent research showed that LLM-based error detectors perform much worse than humans [325]. Other approaches, such as using multiple models, might also help, though they will not eliminate errors, and the cost/benefit tradeoffs need to be carefully evaluated.

[James C: Error vs Attack: Would we consider the threat of adversarially chosen inputs as an LLM *error* or an *attack*? For example, during dispute resolution, the adversary may selectively chose to include evidence or even bias evidence directly, in order to intentionally traverse an artificial decision boundary in the model. For example, there are various "prompt rewriting" techniques (e.g. Greedy Coordinate Gradient) , which uses model gradient information to nudge the output towards a decision boundary chosen by the adversary. Alternatively, the LLM judge output may be pushed towards higher uncertainty (or higher entropy) with carefully chosen adversarially controlled evidence during a dispute.] [Fan: good point! I added a sentence to the end of the first paragraph.]

B-4.1.2 Tolerating LLM errors.

Even in contexts where LLM performs very well, the error rate is not zero. Therefore, systems that use LLMs as oracles must handle potential errors. Three possibilities exist. First, applications that rely on LLMs need to be designed to tolerate errors, so that minor errors do not lead to catastrophic outcomes (e.g., use AI to resolve low-value markets with a total payout below a threshold). The second approach is to involve humans in the loop to detect and correct AI errors. UMA Optimistic Oracle (OO) [95] is an example: after answers are submitted by a proposer (human or AI bot), a dispute window of 48 hours opens, during which anyone can challenge the correctness of the submissions, and if necessary, an arbitration process is initiated and eventually the entire set of UMA token holders verify

the event outcome and vote on the final result. However, the need for humans in the loop slows down decision-making.

A third possibility is to involve humans only when AI models cannot make a decision with high confidence. Recent works show that abstention, i.e., not giving an answer when facing uncertainty, can increase model accuracy [326, 327]. If AI models can abstain with high accuracy, then this design can make fast decisions when AI outputs are available and slow decisions only when they're not.

Key Takeaway B-4.1: Handling error of AI-powered oracles

Systems that use AI-powered oracles must handle potential errors. There are three high-level approaches: 1) designing the system to tolerate errors (which may only be possible in limited scenarios), 2) involving humans to arbitrate AI outputs (which slows down decision making), and 3) if AI models can abstain when facing uncertainty, it is possible to only involve humans when AI cannot decide, thereby combining the speed of AI and the accuracy of human decisions.

B-4.2 Execution: Enabling smart contracts to use AI models and tools

Regular smart contracts can enforce simple rules, such as conditional statements, but they tend to struggle with higher-level tasks like data analysis, pattern extraction, and planning. Moreover, they generally take action *on-chain*—that is, they directly impact the state of a blockchain, and nothing else. Smart contracts could extend their execution capacity, both on- and off-chain, with access to the broader AI ecosystem.

Specifically, the AI ecosystem comprises a vast collection of AI models and associated tools for interfacing with the real world in modalities including data access and modification, money transfer, and information retrieval. Running these tools on-chain, however, can be prohibitively expensive. Hence a natural question is, how can we enable smart contracts to verifiably access AI models and tools in the real world (off-chain), without incurring prohibitive computational costs on-chain?

Oracles once again provide a potential solution, by enabling off-chain computation to be recorded on-chain in a verifiable manner. Modern oracle systems, such as Chainlink and Supra, can already execute off-chain workflows and return attested results. They can naturally (be extended to) support AI workflows. We will discuss approaches for doing so and their tradeoffs.

AI can transform smart contracts from enforcing static rules into dynamic, context-aware mechanisms. For example, smart contracts can use AI tools to flag fraudulent transactions [328], dynamically adjust parameters, and make automated trading decisions. Executing AI logic directly on-chain is not economically feasible, as computation on-chain can be several orders of magnitude more expensive than cloud computing [329]. As an efficient alternative, oracles can connect smart contracts to off-chain AI tools, whether local models running within the oracle infrastructure or third-party ML service providers (e.g., OpenAI APIs). [James C: Language/Figure: ...whether local models *are running* within the oracle infrastructures or *on* third-party ML service providers ... as shown in the figure below (missing figure?).] [Fan: removed the reference to the figure]

Two security properties are important for this application of oracles. First, smart contracts need to efficiently verify the *integrity* of the entire AI workflow, ensuring that the inputs from smart contracts are not tampered with, the correct ML model is used, and the outputs are correctly computed. These requirements are not unique to AI (they apply to off-chain computation by oracles in general), but the scale of AI workloads demands highly efficient solutions. We refer readers to Section C-4 for discussions on three technical approaches that can be used here: 1) an optimistic approach where integrity is ensured by economic incentives (i.e., such protocols are equipped with mechanisms to identify incorrect results and penalize the offender economically) [330, 331, 332, 333, 334]; 2) oracle nodes run ML models in a TEE and accompany the result with a hardware attestation to the correctness of the computation [335];

and 3) oracle nodes accompany the computation results with zero-knowledge proofs [336, 337, 77, 338, 339, 340, 341, 342, 343]. The tradeoffs among these solutions are discussed in detail in Section C-4.

Second, for ML tasks involving private data or proprietary models, their *confidentiality* must be preserved. For example, a smart contract may want to run a proprietary fraud-detection model that cannot be revealed even to the oracle nodes. To do so, the TEE-based approach is likely the most practical [335], where the smart contract developer can encrypt the model under a TEE-secured key, so that the model is only decrypted inside the TEE. Fully Homomorphic Encryption (FHE) can, in theory, be used to evaluate an encrypted model, but it is not yet practical for large models and as it requires a decryption capability, needs to rely on a committee of trustworthy nodes [344].

Beyond security considerations, running AI workloads can consume considerably more resources than typical oracle workflows (e.g., fetching prices), which give rise to a mechanism design question of how to price AI computation [345], and a system design question of how to meter AI usage and charge smart contracts properly. An interesting wrinkle to this problem is the risk of *freeloading*. Because blockchains lack confidentiality, computation results delivered by oracles are publicly available. For instance, a copycat prediction market smart contract can monitor the answers to another prediction market that pays for AI-powered oracles and obtain free answers to resolve its own markets. The freeloading issue was discussed in Town Crier [103], where the authors suggested using designated-verifier proofs to render the output verifiable only by the original requester, though a full solution is left for future work. We note that the issue is much more pronounced in the context of AI workloads, as the potential savings by freeloading AI computation can be significant. [James C: See suggestion above on *AI execution* as an orthogonal topic to AI sensing and AI decision-making.]

B-4.3 Decision-making: AI-based investment tools [Ari, Andrés][Reviewer: Matt REVIEW AND REVISION COMPLETE]

If we give smart contracts access to AI models and tools as suggested in Section B-4.2, a natural question is how those tools will impact on-chain applications. Incorporating AI into smart contracts' decision-making processes introduces substantial complexity and opacity, whereas a strength of previous smart contracts was their (relative) transparency and interpretability. In this section, we describe a case study from the finance realm, showing how using AI-based decision-making in smart contracts can give rise to new tensions and sources of potential unfairness among participants. Specifically, one of the most popular applications of AI to blockchains today is in investment tools. We discuss ML-based Collective Investment Algorithms (CoinAlgs) and their hazards.

B-4.3.1 CoinAlgs

Collective Investment Algorithms (CoinAlgs) are algorithms shared by a community of users that drive collective investment actions [346]. Algorithmic trading has long been common practice in traditional finance, both in major firms (high-frequency trading, hedge funds, quantitative investment, etc.), as well as in ubiquitous products used by retail investors (robo-advisors, trading software packages, etc.).

More recently, with the widespread deployment of AI, CoinAlgs have become common in decentralized finance, too. The most prominent example is *AI-powered investment DAOs*—communities of people that pool funds in a blockchain to make collective trades, which are dictated by AI models or agents. Branded as "decentralized hedge fund managers" [347], such CoinAlgs have been the source of significant interest within Web3. Popular CoinAlg projects, such as ElizaOS [348], an AI-powered investment fund converted into a general AI platform, and AI XBT [349], an AI-powered market-intelligence agent, reached peak market capitalizations of $2.7B and $4.7B respectively, and peak assets under management of $22.9M and $755.4M respectively. Other popular CoinAlgs include Singularity-DAO [350], which offers "AI-Powered Quant Strategies" for DeFi and Soldex [351], a DEX that offers AI-powered trading bots.

B-4.3.2 Risks of CoinAlgs

While they promise to democratize finance, CoinAlgs pose hazards to their users as a result of their susceptibility to well-studied ML attacks. A long line of work, under the broad umbrella of *adversarial machine learning* [352, 353, 354], has shown that AI-based systems can be vulnerable to attacks that include prompt injection [354, 355], memory injection [356], data poisoning [357], backdoor attacks [358], and model extraction [359]. Such attacks can have serious effects in practice, such as modifying the behavior of the system or leaking sensitive information about its architecture. They pose risks for AI-based CoinAlgs that range from manipulating investment decisions to revealing proprietary trading strategies. These concerns are not strictly theoretical; for instance, in recent work, Patlan et al. [356] showed "context manipulation" attacks can be used to trigger malicious asset transfers against ElizaOS.

Beyond general attacks against AI systems, CoinAlgs raise specific concerns when used in finance. For instance, a number of papers have shown that AI trading agents can *manipulate or influence trading markets in negative ways*. Recent work by Dou et al. [360] shows that, even without direct communication channels, profit-maximizing AI trading agents can *coordinate* and *collude* among themselves, which can lead to inefficiency in the markets. Similarly, other works have shown how AI agents can manipulate financial benchmarks [361], spoof limit order books [362], and evade regulation on market manipulation [363].

Recent work by Fábrega et al. [346] highlights a more fundamental problem for investors. CoinAlgs face an inherent security tradeoff in their design dubbed the *CoinAlg Bind*. Intuitively, a CoinAlg's trading strategy can either be *transparent*, or it can be (at least partially) *private*. Both alternatives raise serious risks for investors. Transparent trading strategies (e.g., open-source trading packages) naturally come at the expense of profits, as this can lead to strategy theft or even risk-free forms of arbitrage such as sandwich attacks. This thus motivates keeping (profitable) CoinAlgs private and their "secret sauce" hidden from potential competitors. Indeed, in practice, most CoinAlgs (in both traditional and decentralized finance) opt for private trading strategies [346]. However, this alternative opens up the risk of unfair *value extraction by insiders* – private trading strategies permit an information asymmetry in which insiders with full knowledge of a CoinAlg's strategy (e.g., its creator or host) can use their privileged information to extract value from the CoinAlg's trades. Formally defined as *fairness* in [346], these risks of private CoinAlgs are akin to insider trading in traditional finance.

Fábrega et al. [346] formally prove the existence of the CoinAlg Bind via two complementary theoretical models: one which compares the value extracted by players with asymmetric knowledge of a CoinAlg's trading strategy, and another which formalizes the interaction between a CoinAlg and players with advance knowledge of its trades. Furthermore, they validate the Bind empirically, via extensive simulations on real-world blockchain data. An inherent and unavoidable design tradeoff, the CoinAlg Bind is one of the central challenges for the future deployment of CoinAlgs.

Key Takeaway B-4.2: The CoinAlg Bind

AI-based collective investment algorithms (CoinAlgs) face a fundamental tension between profitability (which requires privacy of investment strategies), and fairness (which requires transparency of investment strategies).

B-4.3.3 Towards mitigations

Despite their risks, CoinAlgs are an inevitable part of the investing landscape, and thus the design of guardrails that limit their hazards is a critical question for future research.

In traditional finance, the CoinAlg Bind is sidestepped through regulated investor protections, which allow CoinAlgs to be public (and thus profitable) while disincentivizing insider value extraction (and thus enforcing fairness). However, less regulated environments like Web3 need to rely instead on *technical*

countermeasures for the CoinAlg Bind. In particular, due to the risks of profit loss, it is likely that Web3 CoinAlgs will continue to be be private, and thus guardrails that minimze the risks of unfair value extraction are of particular importance. The challenge, of course, is that such guardrails should not themselves come at the expense of the CoinAlg's profits.

Question B-4.1: Technical guardrails for the CoinAlg Bind

What technical mechanisms can minimize the risks of unfair value extraction, without decreasing the profits of a CoinAlg?

Fábrega et al. [346] explore some initial directions for the design of guardrails. Most notably, they propose the use of what they term a *randomizing wrapper*, which is a (transparent) algorithm that randomizes a CoinAlg's trades prior to their execution. By running the CoinAlg inside a private and trusted execution environment (i.e., a TEE), such randomization can make it more challenging for insiders to predict the CoinAlg's trades ahead of time, decreasing their ability to profit from their privileged information. Furthermore, since wrappers are public, users can ensure that trades will be honestly randomized. An important direction for future work is further and more principled study of randomizing wrappers, both in terms of theoretical models for their security guarantees, as well as empirical studies that quantify their utility in practice.

B-5 Future Risk: AI-powered Rogue Smart Contracts [Ari][Reviewer: Giulia REVIEW AND REVISION COMPLETE]

Endowing smart contracts with AI capabilities can vastly expand their application scope, as discussed in Section B-4. This expanded scope, unfortunately, includes not just good applications, but also malicious ones. This is because smart contracts were designed as a technical alternative to human and institutional trust and conflict resolution. In a system that uses smart contracts for conflict resolution rather than human processes, potential beneficiaries include those with the least trustworthy human and institutional relationships: criminal actors.

The idea that smart contracts supplemented with AI could supplant "honor among thieves" was suggested in [364]. A *rogue* smart contract[1] as described there offers a bounty / reward for the perpetration of a crime. (Or it can do the reverse: Offer a criminal service for pay.)

Figure B.5 illustrates how such a contract might work. Here, a rogue smart contract `RogueSC` has been created that offers a monetary reward (`$reward`) to commit a criminal act: vandalizing the Washington Monument.

A criminal actor $\mathcal{A}$ who wants to claim the reward must, of course, prove to `RogueSC` that he committed the requested criminal act. The protocol for doing so makes use of what is referred to as a *calling card*, a distinctive detail about a crime that attributes it to a criminal. The idea is to have $\mathcal{A}$ privately commit to a calling card C—a detail of the crime as he plans to commit it—*before* actually committing the crime. He reveals C it afterward and, if C corresponds to reports of the crime, it proves that $\mathcal{A}$ was responsible and should receive the reward. That's because, if C is chosen well, only $\mathcal{A}$ could have known it in advance.

Where does AI come into play? The smart contract `RogueSC` must ascertain that the calling card in C actually happened in the physical world in a process called *adjudication*; in practice, this might come from news reports or other trusted third-party sources that report notable details about the crime. Such adjudication is not straightforward for smart contracts, however, as the data in question is not necessarily standardized or quantitative in nature. For example, if the crime is "vandalize the

[1] The term "criminal smart contract" is used in [364].

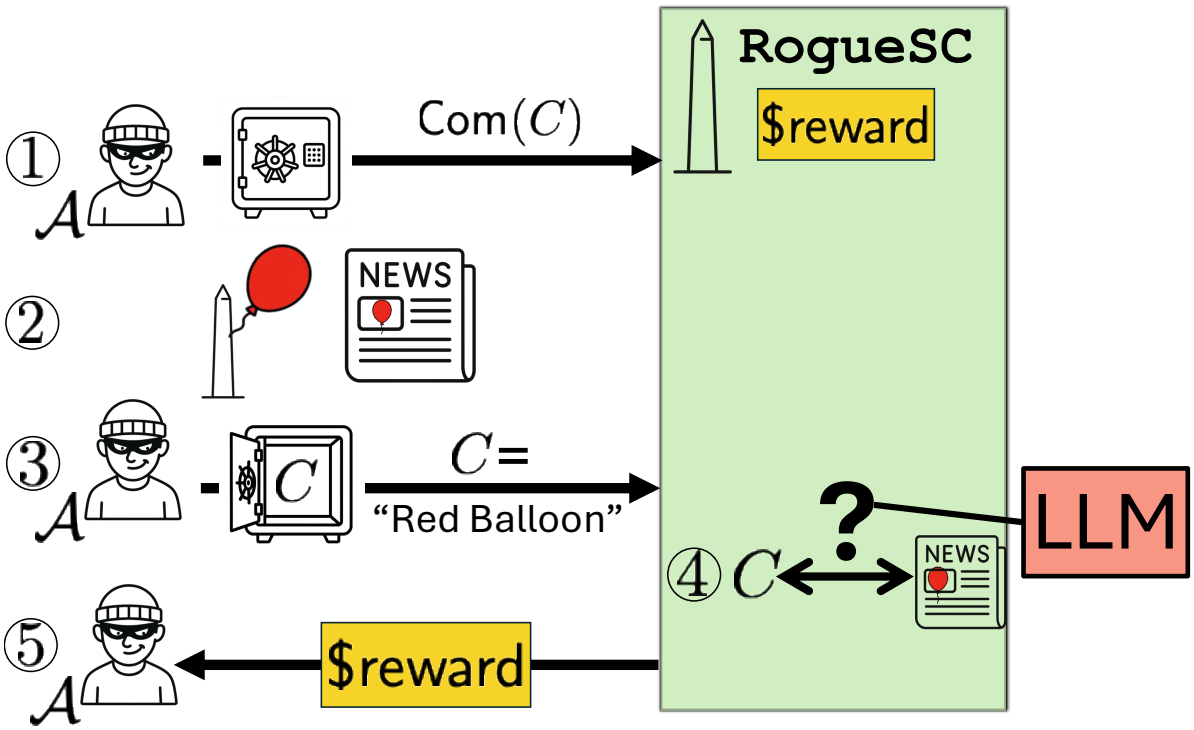


Figure B.5: A criminal actor $\mathcal{A}$ claims bounty $reward for vandalizing the Washington Monument by attaching a giant red balloon to it. $\mathcal{A}$ uses the calling card C = "Red Balloon." Steps are detailed below.

Washington Monument" and the calling card C includes "red balloon," it is not trivial for a smart contract to evaluate whether the crime occurred, and if so, whether the calling card appeared at the scene of the crime. One of the most promising ways to automate this adjudication is with an ML model such as an LLM called by RogueSC (in practice, using an oracle).

Step by step, RogueSC works as follows. After someone creates the smart contract—in this case, offering bounty $reward for vandalizing the Washington Monument—a criminal actor $\mathcal{A}$ can claim the bounty using the following protocol:

1. **Calling card commitment:** $\mathcal{A}$ chooses a calling card C and sends a cryptographic commitment Com(C) to RogueSC. (A commitment conceals C but makes it immutable.)
   - Running example: C = "Red Balloon."
2. **Execution of crime:** The crime happens and is reported in the news.
   - Running example: A giant red balloon is flown from the Washington Monument.
3. **Decommitment:** $\mathcal{A}$ decommits C, i.e., reveals C to RogueSC.
4. **Validation by ML model:** RogueSC checks that C corresponds to news reports.
   - Running example: RogueSC asks an LLM whether "Red Balloon" features in recent news reporting on vandalism of the Washington Monument.
5. **Bounty payment:** Bounty $reward is paid to $\mathcal{A}$.
   - Running example: The LLM says **yes**, validating $\mathcal{A}$'s claim.

This same structure can be applied to any of a number of crimes, and the information used to check claims can come not just from public sources but also *private-web* data sources by means of privacy-preserving oracles. In this case, the construction can closely resembles the secure inference pipelines described in Section C-6.2. Given its use of source authentication, it need not even require a calling card, as in the following examples:

- **Targeted harassment:** A criminal can prove a campaign of harassing e-mail exchanges (and filter out any self-identifying information, if exchanges are not anonymous). An ML model can evaluate the effectiveness of the campaign.
- **Theft of organizational intelligence:** The employee of a company or other organization can exfiltrate data from a corporate intranet and prove its origins anonymously. For example, intelligence facilitating insider trading could be used to claim a bounty (or be sold via smart contracts). The same approach could be used for intellectual property, product plans, etc. An ML model can appraise and assign a monetary value to the stolen intelligence.
- **Whistleblower Doxxing:** Someone with access to a whistleblower complaint can prove the whistleblower's identity with stolen documents, an ML model validating the strength of the supporting evidence.

If `RogueSC` can be deployed anonymously and funds are hard to trace, both the entity soliciting the crime and the entity performing it can act anonymously and with impunity. Privacy-preserving payments, whether through use of mixers [365, 366, 367, 368] or more tightly integrated privacy technologies [369, 370], carry risks when it comes to rogue contracts just as they do in many other settings.

Countermeasures. The countermeasures regularly used to combat crypto-related crime—on-chain analytics to deanonymize transactions, blacklisting of tainted funds—can serve as effective countermeasures to rogue contracts, with the provisos regarding privacy given above.

There is an additional important countermeasure specific to rogue smart contracts, however. That is for oracles that deploy ML models to *surface AI safety measures*, meaning that ML models deny service in cases where there is an apparently high risk of abuse. For this purpose, the *context* of a request needs to be specified, as risk assessment is otherwise challenging. For example, it's not obvious our of context that assessing a news article for mention of a "red balloon" is an expression of nefarious intention. Providing the target logic (smart contract code) for the oracle request, however, may reveal the threat.

Of course, as with all AI safety measures, there is a risk of both false positives (inappropriate denial of service) and false negatives (failure to detect abuse).

Also specific to smart-contract environments is the possibility of some entity standing up a *rogue oracle* service. Such a service, if it achieves the aspirational crypto properties of trustworthiness and censorship resistance, would enable bad actors to sidestep oracles systems that implement safety mechanisms.

B-6 Conclusion and Future Directions [Reviewer: Giulia REVIEW AND REVISION COMPLETE]

Overall, there has been substantial research on using AI to aid in the constructive design of algorithms for blockchains. It can be used both to *design* applications themselves, as in the design of DeFi market structures. It can also be used to design algorithms that *relate to* a given smart contract, such as strategies for attacking a smart contract or maximizing extractable value in a market. Approaches in the research literature today use a variety of ML tools, ranging from very basic classifiers and regression models to more advanced methods using RL to design algorithms that optimize a given reward function.

In the last 3-4 years, we have seen a clear uptick in papers using RL to design algorithms for blockchain purposes. A unifying observation is that all of these methods have focused on settings where we can model the environment, explicitly or implicitly.

Key Takeaway B-6.1

In the research community, AI-assisted algorithmic design for blockchains has, until now, focused on settings where we can model the world or environment cleanly. Examples include modeling consensus protocol state spaces, or modeling the impact of prices on a given AMM.

Despite the clear trend in the research literature, a very different trend is emerging in practice and in industry, as highlighted in Section B-5. As AI-assisted coding becomes commonplace, we expect it will become the norm for smart contracts to be written largely by AI, possibly with the assistance of agentic frameworks [151, 349]. Indeed, very recently, an agentic pipeline was able to design exploits that would have extracted over $4.6 million in a benchmark of smart contracts [371]. While the agents used to design these attacks are closed-source, they relied on general-purpose ML models that were not specifically tailored to exploit extraction, beyond prompt-tuning. A new class of workflows that rely increasingly on general-purpose foundation models have several implications:

- We will increasingly design not only blockchain algorithms, but also full-blown implementations with AI. It remains to be seen what degree of separation will exist between algorithmic design and implementation in practice.
- Algorithmic design will increasingly be driven by natural language objectives rather than precisely-stated quantitative reward functions. For example, if a user asks AI to "write a program to make me money from the given smart contract", there is no well-defined reward function. The agent must decide what objective to follow and what boundaries to obey, if any, when trying to make the user money.
- Unlike prior approaches to AI-assisted algorithm design (as in the above Key Takeaway), emerging AI-assisted methods will increasingly design algorithms and code for environments that are not well-understood or modeled.

These implications raise several questions for both the research and industrial blockchain community.

Question B-6.1: Next generation of AI-assisted design

As we migrate from heavily tailored, labor-intensive ML-assisted design of algorithms and analytics to agentic program design based on natural language objectives, what will be the implications for efficacy and security of blockchain algorithms and applications?

FM-assisted algorithms vs. tailored AI-designed algorithms. One concrete research question in this vein is evaluating how agent-designed algorithms compare to prior algorithms that were tailored to a particular problem domain.

Research Question B-6.1

On downstream tasks such as blockchain analytics or exploit design, how do existing classical AI-based algorithms that are tailored to a specific task and data type compare to algorithms that

are designed by agents backed by general-purpose FMs (i.e., not explicitly tailored to downstream tasks, other than via prompt design)?

For example, in the space of fraud detection, existing classifiers in the literature were all trained on relatively small, carefully-curated corpora of transactions.[2] Approaches that rely on frontier FMs instead learn from much larger—albeit unlabeled, and potentially irrelevant—corpora of public information. It would be useful to evaluate how these methods compare to each other on the same, held-out test set for various downstream tasks. While we expect that tailored algorithms will excel in some narrow settings, prior ML-for-blockchain papers typically make assumptions about the environment (e.g. data comes from a particular distribution, or environments remain static over the time of experimentation). Hence, it will be important to understand how prior methods compare to general-purpose agentic methods under distribution shift due to changing environments. Similar experiments could be run for prior RL-based methods from the literature, such as algorithms for MEV extraction or exploit generation.

Making better use of general-purpose FMs. In the previous research question, we asked for a direct comparison between tailored AI algorithms and agent-driven algorithms backed by general-purpose FMs. However, there is a middle ground: FMs can be fine-tuned to blockchain-specific tasks and data types. A natural, broader question is how best to make use of general-purpose FMs for blockchain-specific tasks.

Research Question B-6.2

For blockchain analytics and design, how should designers best make use of pretrained FMs and small downstream datasets of labelled data?

Today, the dominant paradigm in the ML community for adapting FMs to downstream tasks is RL-fine-tuning, in which a pretrained LLM is fine-tuned on a downstream task or dataset that is representative of the desired skills [372, 373, 374, 375]. These methods typically produce several responses to a single prompt, then use various methods to *rank* or *compare* the quality of the outputs. The resulting rankings are then used to fine-tune the base FM to steer it towards better outputs. However, existing RL-finetuning algorithms are all designed for general-purpose adaptation to downstream tasks. Hence, it would be important to understand if there are blockchain-specific attributes that could be exploited for RL fine-tuning, or even continued pretraining [376]. For example, in the blockchain setting, it is unclear how to evaluate the quality of a FM's response to a given prompt, or even how to design the prompts themselves. For an analytics problem, should the prompts ask to predict whether a transaction is fraudulent? What context should be provided? Such questions may be heavily task-specific, and we believe there could be a rich body of work exploring different methods of adapting general-purpose FMs to questions of interest in the blockchain space.

AI-driven security wars. Web3 cybersecurity stands out as an important bellwether for enterprise cybersecurity at large. This is because Web3 exploits are typically immediately monetizable, either by exploiting them in the wild—thereby irreversibly exfiltrating funds—and by availing of multi-million dollar bug bounties for responsible vulnerability disclosure. This creates significant economic incentives for all actors to leverage advancing model capabilities as soon as they are available and prove economically viable.

[2]Some work has explored a middle ground of training FMs on blockchain data [224]. Although the precise downstream task is not pre-specified in that work, it is still trained from scratch on blockchain-specific data.

Nonetheless, it is challenging to predict how security postures will evolve, and early signals are inconclusive. In the academic literature, recent AI cybersecurity benchmarks including BountyBench [377] and EVMBench [378] evaluate model capabilities across the vulnerability lifecycle, structuring tasks around detect, patch, and exploit workflows. Empirical results suggest that these capabilities may evolve unevenly. BountyBench reports higher patch than exploit success rates [377], and EVMBench reports the reverse [378]. Notably, EVMBench observes continued improvements across all tasks with later model generations. Overall, the trajectory of advancing AI capabilities remains unclear, leaving open important questions about how AI capabilities will shape the balance of power between attackers and defenders, particularly in the short term. This motivates an important future research agenda.

Research Question B-6.3

How do different capability trajectories shape Web3 security outcomes? Which trajectories are most plausible, and what leading indicators would signal the direction of advancement? Finally, what interventions are available to preserve a strong security posture as capabilities improve?

These questions relate fundamentally to the economics of security, and may require the development of economic models to predict various outcomes. Broadly, it would be useful to study the AI arms race from the following perspective: if AI gives an asymmetric advantage to one party (attackers or defenders) for some amount of time, how does this impact the global state of enterprise security? How should we change the structure of bug bounty programs to adapt? How will cybersecurity insurance risk calculations change? Broadly, we expect there are many interesting and important economic questions to explore. Relevant research questions include modeling these issues, as well as understanding how to measure the current state of affairs; as we mentioned earlier, results from individual benchmarks can be inconsistent, so it can be challenging to understand which side is "ahead" in this arms race. However, blockchains offer an interesting opportunity in that smart contracts and transactions are public on permissionless cryptocurrencies. As such, there may be an opportunity to directly measure the (correlative) impact of various AI advancements on the rate and magnitude of cybersecurity attacks.

Overall, the research community has already demonstrated that AI-assisted approaches can be immensely useful for the crypto ecosystem, both in terms of understanding existing systems and for designing and interfacing with new ones. That being said, we anticipate significant growth and improvements ahead due to the emergence of novel techniques and tools, particularly those based on state-of-the-art generative modeling. The precise nature of these opportunities remains unclear, and they may surface both benefits and substantial risks (particularly for cybersecurity). Either way, the blockchain industry and research communities will be forced to adapt.

Chapter C

Crypto x AI: AI Applications of Crypto

C-1 Overview: Making AI Pipelines more Decentralized and Trustworthy

AI models are realized through a sequence of events that is commonly referred to as the *AI life cycle*. Many variants have been proposed of this life cycle, but many variants explicitly include data collection, model training, model validation, deployment, maintenance, and governance (Figure C.1).

We maintain that crypto—including related underlying technologies—offers two key channels for altering or improving the AI life cycle. First, it can help to *decentralize* various components of the AI pipeline, from data collection to model training and inference to governance. In principle, this could help to democratize the development and maintenance of AI models. In this survey, we will also consider the more pragmatic effects of decentralization such as cost and coordination. A second opportunity is that crypto can help to *secure* components of AI pipelines, making them less susceptible to malicious providers of data, compute, or storage. Here, we will discuss how technologies that came out of the crypto sphere are remarkably well-suited to solving certain security vulnerabilities that arise throughout the AI life cycle. This chapter will explore these two complementary opportunities. Specifically, we divide this chapter into two groups of sections: those exploring decentralization and those exploring new security capabilities. We summarize the sections of this chapter, and where they lie in the AI life cycle, in Figure C.1.

1. **Decentralization:** These sections will explore how crypto can help to decentralize various components of the AI life cycle, and more importantly, why it matters. Specifically, we cover the following types of decentralization:
   (a) *Decentralized infrastructure for AI: (Section C-2)* This section explores several kinds of AI infrastructure that can be decentralized using tools and technologies from the crypto sphere. These include physical infrastructure like computation (Section C-2.1), virtual infrastructure like data pipelines (Section C-2.2), and application-level infrastructure for AI agents (Section C-2.3).
   (b) *Decentralized Governance: (Section C-3* This section explores efforts to decentralize the governance of AI systems. We explore techniques that have been proposed, along with some of the tradeoffs and central challenges.
2. **Security:** These sections will instead explore how crypto can help to secure components of the AI life cycle. Specifically, we explore the following advances.

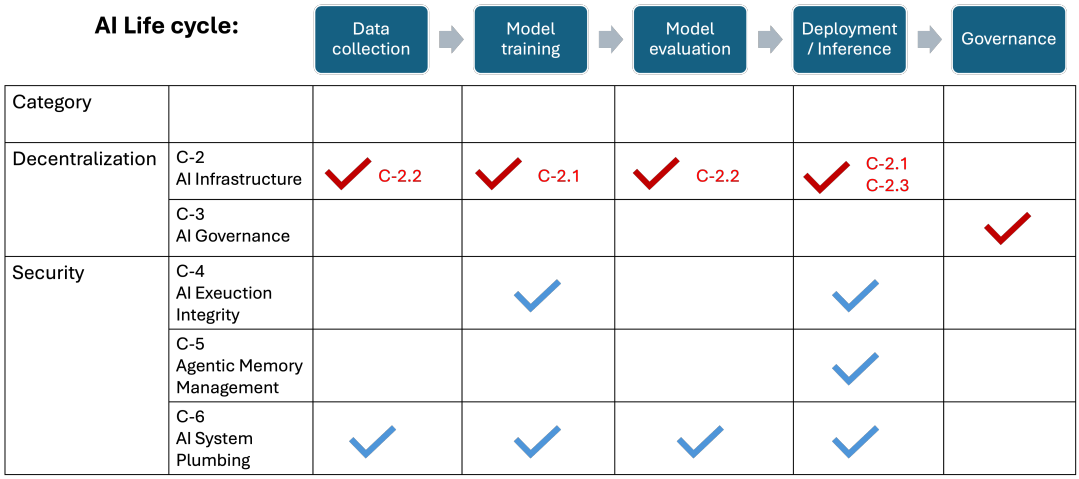


Figure C.1: The AI lifecycle. We envision that blockchains and related technologies can impact the AI lifecycle in two main ways: by decentralizing components, and by securing them to a greater degree. [James C: typo in cell C-4 above: AI Exeuction / AI Execution]

(a) *Blockchains for AI execution integrity: (Section C-4)* This section explores how to use cryptographic tools to ensure that AI execution—both in inference and in training—is trustworthy and verifiable for downstream use cases.

(b) *Secure agentic management: (Section C-5)* This section explores how crypto can be used to assist in the secure management of agentic workflows.

(c) *Securing the plumbing: (Section C-6* Finally, we explore a framework for ensuring data trustworthiness, security, and privacy across the whole AI life cycle via careful use of trusted computing technologies.

C-2 Decentralized Infrastructure for AI [Reviewer: Christian]

Today, the AI industry is growing increasingly centralized [379]. Model providers are starting to internalize data acquisition and processing [380], develop their own algorithms [381], train and run inference on their own data centers [382], and develop centrally-hosted applications in the form of agents with proprietary prompts [383]. Blockchains present an appealing opportunity to decentralize various components of this life cycle. For example, what if communities could collectively train AI models on a distributed network of consumer-grade hardware? What if data could be sourced organically from various parties in a decentralized marketplace? What if AI-based applications could interact and evolve without requiring centralized oversight? The hope is that such decentralization could reduce costs and increase transparency for the AI community as a whole. However, in practice, these considerations are nuanced and can depend on many factors and assumptions. In this section, we discuss the current state-of-the-art and implications of decentralizing AI infrastructure. We discuss decentralized physical infrastructure in Section C-2.1, decentralized markets for data and models in Section C-2.2, and decentralized agent-centric infrastructure in Section C-2.3.

C-2.1 Decentralized Physical Infrastructure Networks (DePIN)[Giulia, Paolo Costa,Jared]

Decentralized physical infrastructure networks (DePIN) are growing in popularity in the AI space. In short, DePIN refers to decentralized networks in which nodes can provide physical infrastructure such as energy, compute, or bandwidth in exchange for financial incentives [384, 385, 386, 387, 388, 389]. While there have been many such efforts in the past (e.g., Folding@Home [390]), DePIN is characterized by financial compensation using Web3 infrastructure. Specifically targeting the AI market, some relevant DePIN efforts include distributed networks of compute nodes (e.g., Theta Network [387], Akash Network [386], and io.net [391], to name a few). These efforts allow AI practitioners to rent CPU and GPU resources on-demand from a global network of providers. In addition to the raw markets themselves, some efforts focus on infrastructure for DePIN markets; for example, Bittensor [385] is a language for defining new DePIN commodity markets under a unified token.

AI-oriented DePIN networks generally compete with traditional cloud service providers on price; for example, at the time of writing in late 2025, the Theta network advertised, "On-demand enterprise grade Nvidia A100/H100 and more at 50-70% cost savings" [387], and the Akash network reported 85% average cost savings [386]. The principal downside of DePIN compute networks is the reduced throughput and latency between machines: as DePIN infrastructure is inherently decentralized, communication between machines will generally travel over the public Internet.

For AI practitioners considering the use of DePIN compute networks, a natural question is, "is DePIN infrastructure beneficial for my job?" Of course, the answer depends on the use case and the characteristics of each job. For example, even if compute infrastructure is cheaper per node on a DePIN network than a traditional cloud service provider, the end-to-end wall-clock time for a job could still be much longer on a DePIN network if the job requires heavy communication across nodes. On the other hand, some use cases may require geographically distributed training for non-technical reasons, such as: (1) privacy regulations restricting the flow of data out of a specific country [392], (2) mitigating the environmental costs (e.g., water, energy) of training models in a particular datacenter region [393], and/or (3) democratizing the training of foundation models [394]. In such cases, DePIN networks could facilitate distributed (federated) training methods.

In the following, we provide some considerations for two main categories of AI workloads: training and inference. For each, we discuss the latency and throughput requirements of the workloads, and explain how these profiles may interact with DePIN architectures. For both training and inference, costs scale with model size; there is a substantial difference between foundation models, which can have trillions of parameters [395, 396, 397], and small language models, which can be substantially smaller [398, 399] or even classical ML models (e.g., support vector machines, linear regressions, etc.). The regime of interest will strongly affect the feasibility of training on DePIN compute networks. We discuss these properties mainly from an efficiency and cost standpoint; we do not further discuss non-technical reasons that might favor distributed training, such as privacy regulations.

C-2.1.1 Use Case: AI model training

Model training can be roughly split into three phases:

- **Pretraining**: the initial training of a model from randomized weights
- **Post-training**: A process of updating or aligning foundation models for additional capabilities (e.g., instruction following, reasoning, or math), often using various reward models possibly obtained by soliciting human preferences.
- **Fine-tuning**: updating the weights of a pre-trained or post-trained model for a downstream task, possibly on a specialized or private dataset

In general, pretraining tends to be more costly than fine-tuning and post-training as it is conducted on larger amounts of training data, whereas post-training and finetuning is conducted on smaller datasets

– and may also leverage techniques such as parameter-efficient fine-tuning (PEFT) [400, 401]. Hence, we will focus on pretraining for this section.

AI model training (especially for large models) requires sophisticated data management. Backpropagation, a building block for most ML optimizers, must compute and store massive gradients averaged over large batches of data. These operations are infeasible on a single GPU node due to hardware limitations, so training is generally distributed across multiple compute nodes [402, 397]. Furthermore, most training algorithms are based on minibatch stochastic gradient descent and its variants, which require synchronous updates across all nodes in the network. As a result, training may stall in the event that any individual node fails or collective communications are waiting on updates from a straggler node.

Throughput considerations. Training a model over multiple GPUs requires a strategy for parallelizing computation. Today, prominent models of GPU parallelism include [402, 403]:

- **Data parallelism** is the simplest and most common form of parallelism; it involves splitting data across different nodes with each node computing the gradients for a subset of the examples in a minibatch. Gradient computations are then averaged across nodes.
- **Pipeline parallelism** involves splitting layers sequentially, such that different nodes process different layers. It induces a tight sequential dependency over the processing of layers.
- **Tensor parallelism,** also known as horizontal parallelism, splits operations within a single layer across nodes. Each GPU works on a shard of the tensor, and results are synchronized at the end of the operation.
- **Expert parallelism** is used for mixture-of-expert (MoE) models that have multiple expert submodels. Different experts are trained on different (groups of) nodes.
- **Context Parallelism:** is used for long-context models operating on sequences with a large number of tokens. Each GPU works on a shard of the computation, partitioned along tokens in the sequence.

These parallelization strategies result in large quantities of data transfer for the communication and aggregation of gradients and optimizer states, with communication costs increasing with the size of the model, training dataset, and hardware platform [404]. However, the throughput profiles of parallelization strategies may differ. For example, pipeline and tensor parallelism tend to have heavy communication footprints, requiring high-bandwidth links. Without careful management and synchronization, all of these parallelism techniques can experience significant scaling challenges [405, 406].

To support synchronization and data transfer during training, most multi-GPU training platforms feature high-bandwidth interconnects supporting several Tbps between local GPUs (e.g. NVLink) [397, 407, 408]; we call such groups of GPUs in the same rack *scale-up networks* (colloquially referred to as high-bandwidth domains) [409]. Beyond these local connections, platforms rely on network technologies like Ethernet or Infiniband to connect the network interface cards (NICs) of different platforms. *Scale-out networks*, for instance, connect GPUs across racks, and *scale-across networks* connect datacenters [409]. The interconnects in scale-out and scale-across networks tend to be lower bandwidth, and a bottleneck in LLM training [404, 403]. Notably, Fernandez *et al.* found that when training LLMs on distributed GPU nodes, as one scales up the number and computational capacity of compute nodes, the system becomes increasingly communication-bound, with GPU utilization falling fastest for high-end nodes like NVIDIA H100s [403]. This is an important consideration for training AI models on DePIN compute networks.

Technical solutions. There have been a number of efforts to train foundation models on distributed infrastructure, including with commercial-grade network connections. For example, in 2022, Yuan *et al.* demonstrated a system that optimizes communication costs while taking into account the average

latency and bandwidth between compute nodes [394]. Such metrics are not directly provided by most DePIN networks, but could be measured prior to (or during) training. SWARM parallelism [410] was later proposed as a pipeline-parallel method for training ML models on unreliable hardware with slow interconnects. Their key idea is to stochastically update which nodes train a given layer based on current network and device conditions. [410] found that as model size scales up, computation occupies a larger proportion of training time relative to communication costs.

To address the requirement that training steps be addressed synchronously, federated averaging approaches (e.g. Local SGD, DiLoCo) perform a fixed number of local gradient updates of model weights on each node and periodically average the resulting models across all model instances [411, 412]. More recent work proposed Decoupled Momentum Optimization (DeMo), an optimizer that modifies momentum updates to reduce communication among accelerators without significantly degrading downstream model performance [413]. Despite the promise of these methods, these research papers evaluated their ideas on (relatively) small models of at most 3B parameters [410, 394, 412, 413]; it is unclear from these papers alone whether the methods work well at the larger scales of frontier models. Several industry efforts are testing these ideas at larger scales (see *Industry efforts* below).

More broadly, there are exist techniques for optimizing communication across high-bandwidth domains at the hardware interconnect level. For example, rail-optimized networks [404] specify how to connect NICs to switches and are commonly used for high-performance computing workloads. However, they still require all-to-all communication between high-bandwidth domains, which may be impractical for DePIN networks. Rail-only networks are instead optimized to LLM training patterns, and get rid of the spine layer from typical GPU clusters [397]. Both optimizations [404, 397] require control over physical infrastructure and may be more practical for data centers (scale-out networks) than fully-decentralized DePIN networks.

There also exist algorithmic techniques for reducing communication between high-bandwidth domains. For example, the NVIDIA NeMo framework utilizes a hierarchical AllReduce framework, distributed optimizer architectures, and chunked inter-data center communications to minimize the size and number of gradients that are transmitted over the wide-area network (WAN) that connects the two datacenters on which they trained their prototype Nemotron-4 model (340B parameters) [414]. These methods can reduce communication costs, and may be useful for training on DePIN infrastructure, but we lack empirical data to make this case conclusively. The test setup for Nemotron-4 is very different from a model being trained on a typical DePIN cluster, where many nodes may be geographically distributed across far more than 2 regions, in many parts of the world.

Industry efforts. Recently, providers like NVIDIA have started supporting distributed training of ML models across data centers [414], which has been validated by training Nemotron-4, an LLM with 340B parameters. Such efforts to train large AI models over wide-area-networks (WANs) raise similar questions to those encountered with DePIN compute networks. NVIDIA addressed these issues in part by minimizing communication over scale-across networks [414], which is common more broadly in distributed training case studies.

The crypto x AI industry has also worked to demonstrate the feasibility of training foundation models (at a smaller scale) on fully decentralized infrastructure, including commercial-grade network connections between accelerators. By August 2024, Macrocosmos AI had facilitated the training of 700M and 7B parameter LLMs on the Bittensor network [415]. The leading model from this work was shown to be competitive with GPT2-Large and Phi-2 in terms of perplexity on web-text and related data.[1] Their white paper focuses on the incentive mechanism for miners to submit high-quality model weights, and miners are expected to individually produce trained models, so the communication network effects we discussed earlier do not come into play (at least not measurably). Other efforts include Prime Intellect asychronously training a 10B parameter model (Intellect-1) in November 2024 over distributed

[1]It is not standard in LLM evaluations to report only perplexity, as done in this paper. Rather, common practice is to evaluate LLMs on a suite of benchmarks, such as MMLU-Pro [416] and GSM8K [417].

infrastructure using fully sharded data parallelism and DiLoCo-based optimization [418]; and Templar AI synchronously training a 1.8B parameter model using an incentive system called Gauntlet [419].

Ongoing work is pushing towards even larger scales. Macrocosmos AI presented an architecture by which multiple miners can collaboratively train a model using pipeline parallelism and DiLoCo, while reaping rewards for individual contributions [420]. At the time of writing there were no public results on the end-to-end training costs or model quality for a fully-trained LLM using this architecture. To our knowledge, the largest model being pretrained on DePIN infrastructure is Consilience, a 40B parameter model under pre-training on the Psyche Network [421]. The Psyche Network uses DeMo for communication-efficient optimization in a data parallel configuration [413]; Nous Research reports that 40B is "compact enough to train on a single H/DGX and run on a 3090 GPU" [421]. [Jared: Note: H/DGX likely refers to a **full node** of 8xH100 or H200 for training as opposed to a single H100 GPU] However, it remains unclear how geographically distributed the infrastructure can be for such large-scale training runs. The DeMo paper explicitly states that "DeMo is designed primarily for optimization across a small number of geographically distributed compute centers" [413].

A broader point about existing industry proofs-of-concept is that they do not typically report total cost metrics (in terms of computation and communication cost). If the decentralized ML community aims to democratize the training of large-scale foundation models, it would be useful to standardize cost models and compare end-to-end training costs (in dollars) on centralized vs. decentralized infrastructure. If the results of such measurements are favorable to decentralized networks, this could be a major selling point for the industry. At the same time, we expect such results to depend heavily on the size of the model, the type of training parallelism implemented, and the degree of decentralization among training accelerators (e.g. multiple small clusters vs. truly heterogeneous consumer devices).

Key Takeaway C-2.1: DePIN for AI model training

There are many promising preliminary results about training large ML models on decentralized infrastructure, particularly for smaller models (a few billion parameters). However, we lack clear measurements on geographically distributed nodes (i.e., not in the datacenter setting) highlighting the overall cost tradeoffs as one scales up model size, degree and type of infrastructure parallelism, and geographic decentralization.

Latency considerations. Model training tends to be less latency-sensitive than inference, in part because training is run offline. Nonetheless, the latency associated with distributed training across multiple geographic locations can cause problems especially using standard gradient based optimizers (e.g. Adam, SGD) which require synchronous communication across all nodes. As described by Aubrey *et al.*, "When managing compute across multiple data centers, developers must contend with high inter-region latency (often 20 milliseconds or more) that can introduce performance bottlenecks during gradient updates and model synchronization during large-scale LLM training" [414]. For reference, as of November 27, 2025, the P90 latency between AWS datacenters us-east-1 and us-west-1 was about 73 ms [422]. The NVIDIA NeMo framework addresses this issue by minimizing high-latency communication operations across datacenters with techniques like hierarchical AllReduce [414]. Additionally, distributed optimizers, like DiLoCo, can further mitigate the frequency of such communication operations potentially improving the feasibility of training across a DePIN. While DePIN compute networks can in principle support sophisticated orchestration, many users may not be aware of the need for it. Today, DePIN services partially account for latency by allowing users to select nodes by geographic location [386, 387]. This can help users gauge the latency implications of a particular combination of selected nodes, as latency is lower bounded by the speed of light between two node locations.

C-2.1.2 Use Case: AI model inference

Model inference refers to generating predictions with an already-trained model, such as when a model is evaluated or deployed for a downstream task. For example, this could include evaluating a classifier model on a provided data point or generating a text from a generative model.

Latency considerations. In general, inference tends to be more latency-sensitive than training [423, 424, 425]. Particularly in on-demand use cases (e.g., AI chatbot), users expect a quick response. However, there exist applications (e.g., meeting summarization, document review, data generation) that rely on model inference and do *not* have strict latency requirements [425]. Notably, emerging applications without direct human interaction (e.g. deep research) have relaxed latency requirements that can allow for up to 30 minutes before response [426]. These lower latency use cases can be more flexible about the locations of node providers. Notice that these same considerations would hold with traditional infrastructure providers; for latency sensitive applications, AI practitioners should think about the location of the datacenter(s) processing queries, and their relative distance to the end users.

Throughput considerations. Inference typically has lower throughput requirements than training. First, inference is generally requires less degrees of parallelism over hardware. Whereas training a large foundation model (e.g. 100B parameters or more) can require tens of thousands of GPUs in parallel, inference jobs do not require parallel computation to the same extent. Even tens of GPUs are sufficient to serve the largest models (e.g. DeepSeek-R1 with 671B parameters can be served on 16 H100 GPUs). Second, inference does not require backpropagation, which significantly reduces the memory and computation footprint of the operation. Hence, even under parallelization schemes like those discussed in model training, a network with low data throughput need not be prohibitive. Such tasks may be a particularly good fit for DePIN networks.

Key Takeaway C-2.2: DePIN for inference

Latency-insensitive inference applications (e.g. meeting summarization, document review) may be a particularly promising and cost-effective use case for DePIN networks.

C-2.1.3 Summary

Today, end users evaluate whether to use DePIN networks based on their own understanding of their AI jobs, as well as node price and capacity (e.g., compute, memory). We highlight the following major takeaway.

Key Takeaway C-2.3: Cost-benefit tradeoffs

Users of DePIN networks should evaluate the bandwidth and latency requirements of their use case to determine the overall expected cost savings. Moreover, we suggest that DePIN network providers measure and report at least network bandwidth between nodes, as these figures can drive total cost and help determine whether a particular use case makes sense for a DePIN network.

While current networks typically advertise lower costs in dollars-per-GPU-hour, this can be a misleading metric. The cost metrics that matter for ML jobs are typically training efficiency (i.e., iterations-per-unit-cost) and inference efficiency (tokens-per-unit-cost) [427].

To make it easier for users to determine whether and when to use DePIN infrastructure, we suggest the following questions for the research community.

Research Question C-2.1

What kinds of AI jobs are well-suited to existing decentralized compute networks, as found in DePIN AI networks?

The community would benefit from more systematic (third-party) measurements to profile existing DePIN networks, and also to understand the use cases for which DePIN AI networks makes sense. We envision such studies first measuring the performance profile of various DePIN networks. Then, we propose to map the characteristics of an AI job (estimated compute cost, maximum memory requirements, communication loads, size of model, size of layers, etc.) and desired performance characteristics (cost, end-to-end time) to a recommendation about whether training on DePIN infrastructure would the target performance characteristics. While there have been prior studies on decentralized training of ML models [410, 402], these studies are not conducted on DePIN networks specifically, and may have very different performance characteristics.

Research Question C-2.2

Building on the systematic measurement studies proposed in RQ C-5.1, how should pricing mechanisms for DePIN AI depend on network infrastructure and reliability?

It is plausible that pricing should depend not only on compute capacity per node, but also on network measurements and node reliability. For example, a set of 16 nodes with high-bandwidth pairwise links (e.g., nodes with GPU-to-GPU NVLink, or co-located in the same datacenter) may cost more than 16 nodes that are spread across the world. Likewise, nodes which can guarantee high availability to be able to participate in large coordinated jobs can be priced higher, as a single straggler or failed node is sufficient to interrupt an entire AI training job. While some networks implicitly reward well-connected nodes (e.g., Theta network assigns rewards based on time to completion of a task [428]), others base pricing solely on node capacity. Both models may be too retroactive for large jobs, in which failures are (or can be) very costly.

Research Question C-2.3

How should updated pricing mechanisms for DePIN AI networks account for strategic and/or adversarial nodes?

If we propose updated pricing models as in RQ C-5.2, it is essential to understand the role of strategic and adversarial actors. For example, if a protocol were to reward well-connected nodes disproportionately relative to individual node capacity, a strategic owner of a single node could rent it out as many small co-located nodes to boost profits. This could be less useful for the network—a client renting the many small nodes would have worse performance than if they had rented the one large instance. Beyond such strategic manipulations, we also need to design methods to account for adversarial reporting of specifications. For example, if a node systematically lies about its processing capacity or network connections, there should be mechanisms for catching this and penalizing providers. The design of a proper reward scheme requires mechanism design to ensure that node providers cannot manipulate how they offer their infrastructure in a manner that is inconsistent with network utility, or otherwise lie about their offerings; the latter question of proving physical resource availability has been explored in the context of other DePIN resources, such as network bandwidth [429] and other cellular network resources [430]. Interestingly, defining utility and the threat model itself may depend on the measurement findings from RQ C-5.1.

C-2.2 Decentralized Marketplaces for Data, Models, and Evaluation [Maryam, Matt, Giulia]

Today, data is an essential input to many phases of the AI life cycle: training, grounding at inference time (e.g. in retrieval augmented generation (RAG) pipelines), and model validation, and benchmarking for domain-specific requirements, such as capabilities or safety. Today, the relevant data for each of these phases typically comes from multiple sources. Some of the most common techniques for acquiring data are as follows:

- **Web crawling** is perhaps the most common source of data for AI pipelines. Companies crawl the public internet to collect vast quantities of text, images, and other content (e.g., Common Crawl [431]). Sometimes, these practices can run afoul of content providers' own policies [432].
- **Licensing agreements** with content providers are increasingly used to expose paywalled or otherwise restricted content to AI models. For example, OpenAI entered into a license agreement with the Associated Press (AP) to license news stories [433].
- **Public datasets** have long been curated by various parties and are a component of most modern training pipelines. Examples include Wikipedia, GitHub, arXiv, and Stack Overflow.
- **Synthetic data** is increasingly being used to augment real datasets, particularly in post-training processes that require highly-specialized data [434].
- **Data brokers** are centralized parties that aggregate datasets (often from proprietary or controlled sources) and sell them to buyers. These data sources are not always proprietary or licit—a Reddit lawsuit describes a whole ecosystem of data brokers who are engaging in 'industrial-scale' scraping and selling the outputs, in violation of terms of service [435]. While AI companies have not explicitly confirmed purchasing data from data brokers, there have been multiple allegations and preliminary evidence [436]. Further, data brokers like LexisNexis are increasingly releasing products that expose high-quality data for purported use cases including AI model training [437], suggesting that this is a growing industry.

Our focus in this section will be on the final category of *data brokers*. While there are many sources of data for AI (as listed above), AI models are running out of fresh public data [438]; hence, there is a growing push for non-public data sources, which will most likely be mediated at least in part by data brokers. We will also discuss the adjacent problem of model markets, which are gaining traction, but our main framing will be in terms of data markets. Many of the same principles apply in both cases.

In enterprise settings, data brokers typically operate in (at least) one of two models: direct-to-consumer or marketplace. For example, LexisNexis lists its data on enterprise data marketplaces like Snowflake Data Marketplace [439], while also acting as a direct provider [437]. Our principal interest will be in the marketplace mode of operation. Other prominent examples of centralized data markets include Datarade [440] and AWS Marketplace [441], though data-for-pay is an old concept, as seen in financial data streams [442, 443] and information security data [444].

Note: We will distinguish as needed between *operationally-centralized* markets (which are owned and run by one entity) and *centralized price-setting*, where a market operator unilaterally decides how to price goods. A market can be operationally-centralized but have decentralized pricing, or vice versa. We discuss various combinations in this section.

Key Takeaway C-2.4

AI providers collect data from many sources. While many of these sources are either free or based on pairwise commercial agreements, blockchain-based solutions could complement (or disrupt)

data acquisition from the *centralized data marketplaces* in which data brokers operate today.

[Giulia: Revise this] In the subsequent sections, we overview how blockchain-adjacent technologies and decentralized protocols can play a key role in marketplaces for the data required to train, adapt, and even run AI models. In Section C-2.2.1, we first overview the properties of data markets that set them apart from other market types. In Section C-2.2.2, we highlight the challenges that arise in centralized marketplaces, particularly monopolistic ones. In Section C-2.2.3, we briefly overview how decentralized markets using crypto tools could help address some of the challenges in centralized markets. We provide an overview of existing decentralized data or model markets in the crypto space. Finally, Section C-2.2.4 provides a summary and future research directions.

C-2.2.1 Background: Properties of data markets

Below, we overview key features that distinguish data markets from markets for other common goods.

Data is a digital good. *Digital goods* are costly to create the first time,[2] but free to replicate thereafter, and are extensively studied in the economics and computing literature [445]. Notably, although digital goods are free to replicate, sellers necessarily limit replication in order to extract sufficient revenue to cover the cost of initial creation.

Data can be rival or non-rival. *Rival goods* exhibit negative externalities when sold to multiple consumers,[3] whereas *non-rival goods* can be equally enjoyed by any number of consumers.[4] Jones and Tonetti highlight that non-rival data does not "disappear" when consumed by one party (whereas an apple disappears when one consumer eats it) [446]. Gordon-Tapiero *et al.* highlight aspects of data that are rival – if a dataset is accessed via privacy-preserving tools such as differential privacy [447], then every query (by any consumer) drains the "privacy budget" and limits the queries available for other consumers [448]. In the context of AI pipelines, we generally view data as non-rival, with some notable exceptions (e.g., the differential privacy budget mentioned above).

Data can be lemons. For several years, advances in ML models were driven by neural networks' remarkable ability to improve in quality with added compute and training data *quantity* [449]. However, there are limits to this paradigm. As we feed larger and larger datasets into ML models, it has become clear that data quantity cannot compensate for poor data quality. It has become common for organizations to filter out irrelevant or low-quality samples [450, 451, 452] and/or conduct other data cleaning tasks, such as labeling [453]. This suggests that AI practitioners are (or should be) incentivized to pay for high-quality data [454]. This matters for data markets because they must expose mechanisms for data valuation [452, 455, 456, 457].

Data valuation requires data access. Data quality—like used car quality—is not observable to buyers. So-called "markets for lemons" are extensively studied within economics [458]. A canonical method to address this information asymmetry is for sellers to provide auditable information on the items for sale. This is possible when selling data by simply offering a sample [459, 460, 457], as is

[2]For example, Meta recently paid nearly $15 billon for a 49% stake in Scale.ai (a data labeling company) in June 2025. Source: `https://www.theinformation.com/articles/meta-pay-nearly-15-billion-scale-ai-stake-startups-28-year-old-ceo`.

[3]That is, when another consumer purchases the same good, the initial consumers are worse off. For example, if a baker sells the same loaf of bread to two customers, each customer experiences "negative externalities" from the other customer eating their bread.

[4]For example, if a streaming service sells the same subscription to two customers, each customer can still fully enjoy the subscription service (one might even argue there are "positive externalities" because now the initial consumer can talk with others about the fun show they watch).

commonly done in data marketplaces [440, 461, 441]. One key aspect of data samples, however, is that *these samples themselves provide value, even if the full dataset is never purchased* (whereas a user derives no value from only learning that a car functions well, unless the car is ultimately purchased). Navigating the tension between enabling buyers to value data and not releasing too much value is an important problem in the data valuation research [462, 463, 464, 465, 466].

Data is malleable and can be resold. All digital goods, by definition, can be replicated without cost. Once the first copy is sold, the original producer must therefore worry about the original buyer choosing to resell their newly-purchased data. Sellers of other digital goods (imperfectly) tackle this problem with regulatory tools (such as copyright law) and technical tools (that, e.g., make it technically challenging to widely share a digital purchase).

Regulation and technical tools will certainly both play a role in addressing the same challenges for data, but both tools will need updating. For example, it is (reasonably) straight-forward to decide whether a song is or is not "You Belong with Me" by Taylor Swift. It is significantly less straight-forward to decide whether two similar datasets are in fact functionally the same, since there is more room for modifications to datasets to hide similarities without decreasing quality. Unlike music, it is also possible to break up a dataset into several parts and combine a subset of those parts with parts of other datasets, and the result can largely retain usability. Similarly, when the seller of a digital good ultimately retains some control over its use (i.e. in order to actually play a video game, one must connect to the creators' server), creators' have a wide range of technical tools available. If data is truly sold with no remaining strings, the applicable technical tools need to be more sophisticated.

C-2.2.2 Challenges and benefits of centralized marketplaces

Existing digital marketplaces for other goods (shopping, ads, streaming, music, ridesharing, etc.) have been a topic of discussion and controversy among users, operators, and regulators alike. Some commonly-cited challenges and benefits are as follows:

(−) Opaque pricing and decision-making. One of the most prominent critiques of operationally-centralized marketplaces is the resulting asymmetry and lack of transparency in pricing that often emerges. While some early centralized marketplaces (e.g., eBay and Amazon) allowed sellers to set prices themselves, i.e., in a decentralized fashion, more recent centralized marketplaces (e.g., Uber, Lyft) have largely converged on centralized price-setting [467]. Without transparency or competition, marketplace operators can increase fees for users and limit profits for sellers. Indeed, centralized pricing is known to harm sellers (relative to decentralized pricing) when the market operator optimizes only for overall revenue [467, 468]. Such centralization is growing increasingly common in the AI space [469], and is cited as a major driving factor for many decentralized data and compute marketplaces.

(−) Lack of choice. A common critique is that centralized marketplaces may *steer* users to products that are favorable to the marketplace operator, and limit users' choices. For example, multiple recent lawsuits have alleged that centralized markets like Google's adtech market and Apple's App Store are improperly limiting communication with users regarding alternative offerings [470, 471]. In cases where centralized marketplaces grow into monopolies, lack of choice can turn into a positive feedback loop, preventing small- or medium-sized alternatives from competing with incumbents [471].

+ Increased efficiency. In some cases, centralized pricing can be more efficient than decentralized pricing because it exposes more information to the decision maker. A recent paper by Filippas *et al.* studied a peer-to-peer vehicle rental marketplace that transitioned from fully-decentralized pricing (where vehicle providers set their own prices) to centralized pricing [468]. They found that in this transition, centralized pricing allowed providers to experience significantly higher revenue, while users experienced lower prices. At the same time, providers' revenue per rented hour decreased substantially.

Overall, this suggests that centralized pricing can improve outcomes when implemented carefully and with that goal in mind. In the best case, a centralized data provider may be able to observe if a dataset is obviously lemons, and price it accordingly. It may also be able to manage access to rival data goods, controlling how many parties can access the same data (Section C-2.2.1). The problem is that in most centralized marketplaces, neither sellers nor buyers have transparency into these decisions.[Matt: Added "and with that goal in mind" at the end of a sentence above, since being careful alone is not enough (carefully optimizing for their own revenue is not likely to improve outcomes).]

C-2.2.3 How decentralized data markets can help

The problems associated with centralized marketplaces suggest a widespread interest in alternate market structures. Unfortunately, Tirole notes several barriers to disentangling existing "Big Tech" marketplaces into smaller sub-products once the vertically integrated product achieves dominance [472]. Specifically, breaking up monopolistic networks (e.g. social networks) can materially degrade the quality of service, as network effects disappear. This is true even in data markets: a marketplace with many sellers is more attractive to buyers, and a marketplace with many buyers is more attractive to sellers. Moreover, [473] specifically notes the tendency of platforms with large user bases to accumulate data about users that compounds the quality of the service they provide.

Fortunately, data markets for AI applications are still emerging, and there is not yet a dominant marketplace. Hence, many of the risks raised by Tirole [472] do not yet exist. In the future, if data markets are going to represent a major source of data for AI operators, it is important to design these markets properly in the early stages, leveraging any lessons learned from the Big Tech era.

Key Takeaway C-2.5

AI data markets are still nascent, and there is not (yet) a monopolistic incumbent. Hence, decentralized data markets for training, inference, and validation of AI models have an opportunity to avoid many of the problems associated with past monopolistic markets by leveraging the structures and properties of crypto.

First, note that decentralized data markets are well-poised to make use of certain tools that are commonly used in crypto. A nonexhaustive list includes the following:

- *Micropayments.* Micropayments could enable a new paradigm for how data is transferred and sold: data buyers could select exactly the data samples they need, and pay based on the utility of a particular data sample.
- *Trusted Execution Environments (TEEs).* Perhaps data holders are happy to have their data be used *for a particular task*, but are uncomfortable generally selling their data for arbitrary use. Trusted execution environments could allow the sale of data to be used exclusively within that environment, and therefore temporarily and tied to a particular task (such as training a particular model in a particular environment). TEEs can also be useful for their privacy properties, to get around the challenges with sharing data samples (see below).
- *Zero-Knowledge Proofs for Auditing.* Section C-2.2.1 notes that data can be lemons, but that classical methods to inform buyers risk buyers obtaining value from the information itself despite forgoing a purchase of the dataset. Zero-Knowledge Proofs are therefore a natural tool for sellers to disclose *exactly* the desired information about a dataset to a buyer without risking inadvertent use. Trusted Execution Environments could serve this purpose too; a TEE could, for example, verify that a particular dataset improves model-training without actually revealing the data to anyone outside the TEE.

Building on these tools and others, we highlight several key opportunities where decentralized data markets could innovate over—and possibly improve on—centralized alternatives.

Transparent decision-making. Lack of transparency and control is a common concern in centralized marketplaces [468]. Decentralized alternatives can *guarantee* transparency for protocols and decision-making, including regarding the algorithms used to price data. Transparency—and the resulting competition it enables—is one of the major benefits typically cited with regards to decentralized data markets. However, note that transparent decisions are not necessarily good decisions (see below). [Matt: "policies and algorithms" –¿ "algorithms" to fit in one line.]

Data-dependent, privacy-preserving pricing. Although operationally-decentralized markets can increase transparency, we saw earlier that centralized pricing can improve revenue and reduce prices for users due to using more complete information about the quality of goods being sold. For example, a centralized data market operator may know the value and content of datasets provided by various brokers, and be able to price them accordingly. However, data brokers may be unwilling to expose their data to a decentralized market protocol for valuation purposes. Privacy-preserving protocols could help by allowing a data valuation module to access their data (e.g., using zero-knowledge proofs or MPC constructions) and price it appropriately. This could lead to more efficient pricing overall, without sacrificing protocol transparency.

Replacing monopolistic platforms with protocols. Huberman *et al.* [474] analyze decentralized ledgers as a "monopoly without a monopolist"; while there is ultimately a single ledger with enormous network effects (the "monopoly"), no single party controls access to that ledger (as a monopolist would), and users instead interact with the single ledger via miners who participate in a well-defined competitive process. Subsequent work further analyzes the impact of such decentralized ledgers on users [475].[5] This manner of thinking can be viewed as an economic instantiation of the "protocols, not platforms" paradigm [476]. [Matt: Removed "monopolistic protocols" from the title because it might be confusing this early. I think it's used well in the paragraph, though. Changed the title from operators to platforms to evoke the protocols, not platforms language (I learned recently that this way of thinking has been independently pursued and is quite popular, so it would be good to reference it. I also added the last sentence to draw this connection.]

Decentralized data markets could evolve in a similar way, where many data sellers operate and compete under the (transparent) rules of a single, "monopolistic" protocol. One example of a marketplace in this spirit is proposed in Resonance [477], which is inspired by Ethereum's Proposer-Builder-Separation (PBS). PBS separates the role of proposing Ethereum blocks (which has minimal technical requirements, and whose decentralization is central to Ethereum's value proposition) from building them (which requires deep technical sophistication to optimize, and is likely to be dominated by a small number of specialized entities). PBS was developed primarily to mitigate the centralizing effects of builder specialization on proposer decentralization; i.e., if instead all economically viable proposers must also build well, the number of viable proposers decreases. Resonance is a mechanism for matching buyers and sellers of AI compute, and similarly separates the (technically simple) protocol execution from the (technically sophisticated) problem of combinatorial optimization.[6] The "monopoly without a monopolist" lens proposes a different view on Resonance: although there is a single "monopoly" where all buyers/sellers gather, no single entity acts as a "monopolist" for matching buyers to sellers; brokers instead compete in a formally-specified mechanism in order to win the right to match buyers to sellers.

[5]The regulatory paradigm of Local Loop Unbundling (LLU) is a good analogy to have in mind for this framework. Physical infrastructure for the Internet exhibits network effects, and users derive positive externalities from being on the same physical infrastructure as other users. LLU is a regulatory paradigm that requires owners of physical infrastructure to lease access to Internet Service Providers (ISPs) at non-discriminatory prices, and then consumers can choose among ISPs who compete to serve them. To draw analogies across domains, the ledger and physical infrastructure all exhibit significant network effects, but users interact with miners or ISPs to purchase the ultimate product.

[6]Specifically, buyers of AI compute have multi-dimensional needs and sellers have multi-dimensional resources. Optimally matching buyers to sellers is therefore an NP-hard combinatorial optimization problem.

More generally, [478, 479] consider decentralization as a means for a platform to commit to future behavior by ceding control to a decentralized protocol, and study whether a platform designer might in fact profit as a result of these commitments (which would not otherwise be credible without ceding power). As a hypothetical example, imagine a platform that can decide a rate at which to display ads. A centralized platform will choose the rate that optimizes their profits, and lacks the ability to credibly commit to do otherwise. These papers consider instead the possibility that the platform formally cedes this decision-making power "to the users," in a manner that is credible and verifiable. [Matt: removed "to have in mind" so it fits in one line.]

Industry efforts. Developers have already begun using blockchain-adjacent technologies to facilitate the exchange of data and AI models. We provide a partial overview in Tables C.2 and C.3. The platforms discussed are roughly categorized according to whether they sell access to data or trained models, and we explain how prices are set in each platform. Several trends emerge, which we summarize below.[Matt: Replaced /quad with ∼ to fit in one line.]

Key Takeaway C-2.6

While many existing platforms and protocols use decentralized mechanisms for payment processing (e.g., cryptocurrencies or stablecoins), it remains unclear how else decentralization concretely affects these products and markets. Many platforms either use centralizing pricing mechanisms (i.e., pricing is determined by the protocol designers), while others allow sellers to fully specify their own prices—two pricing variants that already exist in centralized markets. In general, the question of *how decentralization can improve data and model markets* remains under-studied.

Although the entries in Tables C.2 and C.3 all run on blockchains for decentralized payment processing, several of them have centralized payment rules. For example, Grass [480] pays users according to a fixed pricing scheme for idle Internet bandwidth, and use this bandwidth to scrape (selling the results to AI model trainers). Hivemapper [388] also disseminates rewards to data providers according to a fixed policy. In the research domain, Proof of Improvement (PoIm) has been proposed as a technique for evaluating proposed updates to an ML model, and disseminating rewards accordingly [481].

Other platforms allow data or model providers to set their own prices, such as IOTA Data Marketplace [482], Oraichain [483], Sentient GRID [484], SingularityNET [485], and Story Protocol's Poseidon Marketplace (under development) [486]. Vana allows buyers to indirectly set prices via demonstrated demand [487].

One interesting middle ground is Bittensor [385], which has multiple subnets, each with a different subnet. Within a subnet, a project owner specifies their own incentive mechanism for model providers. Once model providers produce their submissions, validators evaluate them, after which TAO tokens are disseminated to model contributors proportional to value. This setup could allow for controlled experiments regarding mechanism design, and how mechanism design affects final model quality.[Matt: IIs there a typo here: "multiple subnets, each with a different **subnet**"? Should the second one be owner?]

Overall, there is some variety in pricing mechanisms across platforms, but we observed limited exploration of the effects of different pricing mechanisms—particularly hybrids between fully-centralized pricing and fully-decentralized pricing. Exploring this design space is an important and interesting direction for future work.

Key Takeaway C-2.7

Privacy-preserving computation over data is a common feature of existing decentralized data marketplaces, though different providers tackle the problem using different tools and techniques.

Several existing platforms leverage privacy-enhancing technologies such as TEEs or encrypted computation to manage private or sensitive data (e.g., Sterling [488], Oasis [489], Ocean Protocol [490], and Vana [487]). At their core, these systems use blockchains and cryptography to provide a payment system tailored to paying only for verified transfer (either of data or AI model outputs) [488, 490, 483, 487, 485]. Various systems differ in how exactly they provide privacy; for example, Oasis has recently started using differential privacy to obfuscate SQL queries [489], whereas Ocean protocol allows data providers to execute computation locally, on their own data [490].

Figure C.2: Partial list of decentralized data and AI model markets (Part 1).

Platform	What Is Being Traded	Details	How Prices Are Determined
Bittensor	**Models** AI model outputs (intelligence)	A network of *subnets*, each a competitive marketplace for a specific AI task. *Miners* serve model outputs; *validators* score quality. One subnet's output can feed another, enabling full AI pipelines within a single protocol.	*Proof-of-Intelligence* distributes TAO token emissions proportional to validator-scored performance. No list price—reward magnitude is the implicit price signal.
Grass	**Data** Web data scraped via bandwidth sharing	Users share idle internet bandwidth via a browser extension. The network scrapes public web data through residential IPs at scale, using ZK proofs to record provenance on Solana. Data is cleaned and sold to AI companies as training datasets.	Enterprise clients pay the Grass Foundation via off-chain contracts. Node operators earn GRASS tokens based on uptime and active bandwidth utilisation.
Hivemapper	**Data** Street-level mapping data	Dashcam owners contribute fresh imagery via a dedicated device, building a decentralised alternative to Google Street View. The network sells map access to enterprise customers (logistics, insurance, autonomous driving).	Enterprise buyers pay a subscription or per-tile fee set by the Hivemapper Foundation. Contributor HONEY rewards are algorithmically scaled by imagery freshness, uniqueness, and geographic demand.
IOTA Data Marketplace	**Data** IoT sensor data (micro-streams)	Built on the IOTA *Tangle* (a DAG, not a chain), enabling feeless micro-transactions. IoT devices—sensors, vehicles, smart city infrastructure—sell tiny data packets directly to other machines, making M2M data economies practical.	Device owners set asking prices; buyers pay in IOTA tokens. The feeless architecture enables sub-cent per-packet pricing impractical on fee-based chains.
Oasis Network	**Data** Private or sensitive data	A Layer 1 with confidential computing via *Trusted Execution Environments* (TEEs)—computation runs on encrypted data, invisible to node operators. Users tokenise personal data and stake it with dApps, earning ROSE rewards while retaining control.	Access terms negotiated via smart contracts between data owners and dApps. Users pay a premium for TEE privacy; token rewards come from network usage fees.
Ocean Protocol	**Data** Access rights	Datasets are tokenised into ERC-20 *datatokens*; holding 1 datatoken unlocks access to a linked dataset. An ERC-721 *Data NFT* represents base IP ownership. A *Compute-to-Data* (C2D) feature lets buyers run algorithms on data without it leaving the owner's servers.	Fixed price (set by publisher) or AMM-based dynamic pricing via *bonding curves*. Buyers pay for datatokens in OCEAN tokens; price adjusts automatically with supply and demand.
Oraichain	**Models, Data** AI oracle and data outputs	An AI oracle Layer 1 (Cosmos SDK). Smart contracts request AI-powered feeds—price predictions, biometric checks, credit scores. Each request attaches *test cases*; providers must pass a minimum threshold to be paid, ensuring output quality.	Fees split among validators, test-case providers, and AI API providers, paid in ORAI tokens. Providers set their own rates; reputation scores influence which providers are selected.
Sentient GRID	**Models** AI model usage rights	Models weights are freely downloadable, but *model fingerprinting* ensures outputs degrade without a valid authorization signature. Usage fees flow on-chain to trainers, deployers, and validators.	Owners set per-call fees; payment triggers the signature required for accurate outputs. The SENT token coordinates governance and rewards across the network.

Figure C.3: Partial list of decentralized data and AI model markets (Part 2).

Platform	What Is Being Traded	Details	How Prices Are Determined
SingularityNET	**Models** AI services (API calls)	An open marketplace of AI APIs (computer vision, NLP, robotics, etc.). Developers publish services on-chain; buyers call them via smart-contract escrow that releases payment only on delivery. Computation runs on the provider's own servers.	Each provider sets their own per-call price in AGIX tokens. Rates emerge from provider competition; no central authority controls pricing.
Story Protocol	**Data** IP rights (AI training data & creative works)	Creators register works (text, images, audio) as on-chain IP assets. Smart contracts encode licensing terms—permitted uses, royalty rates, and attribution. Royalties flow automatically when AI models train on or derive from registered assets.	IP owners set licensing fees directly in smart contracts (fixed or revenue-share). Prices discovered through supply and demand; popular or rare training data commands higher royalties.
Vana	**Data** User-owned personal data	Built around *DataDAOs* that pool user-exported personal data (Reddit, Spotify, health metrics) for sale to AI researchers. AI builders must *burn* VANA and DataDAO tokens to access data; 80% of fees flow back to contributors. Data is processed inside TEEs.	Buyers pay by burning tokens, directly linking demand to price. DataDAO rankings (refreshed every 3 weeks) determine VANA emissions—a competitive subsidy on top of direct access fees.

C-2.2.4 Summary

Decentralization is certainly not a panacea for decentralized data and model markets, but the capabilities it introduces merit serious consideration before dominant AI data marketplaces emerge. For example, the [474, 475] lens asks: might users be better served by vertically integrated marketplaces that provide a one-stop shop for all data-purchasing services or by decentralized marketplaces where some derivative services (such as search, dispute resolution, pricing, authentication, etc.) are offered by competing third-parties? Similarly, the [478, 479] lens asks: might these emerging platforms increase user acquisition (and eventually serve those users better) by ceding control of some platform decisions to decentralized governance?

Existing platforms and protocols have started to build the decentralized rails over which marketplace payments could flow, but there has been relatively little experimentation with market design and mechanisms, and many open questions remain. The following research questions are motivated by the above discussions.

Research Question C-2.4

When designing data marketplaces, which derivative services should be vertically integrated with the platform, and which should be reserved for competing third parties?

The "monopoly without a monopolist" [474, 475] lens proposes to consider marketplaces that are not vertically integrated, and where some derivative services are instead provided by competing third parties. But, exploration of this framework is very nascent, and does not yet provide recommendations as to *which* derivative services should be separated from vertical integration,[7] nor whether it is technically feasible to seamlessly separate different segments required to operate a functioning marketplace.[8] Exploring this design space more systematically is important for the future development of decentralized data and model markets.

Research Question C-2.5

How can we design marketplaces so that important auxiliary segments of the AI pipeline (benchmarking, red teaming, fine-tuning, etc.) can be incentivized and rewarded?

Similarly, it is also important to understand which AI-adjacent services should be bundled with the core product, and which should be left to competing third parties. For example, red-teaming is typically viewed as an after-product of developing an AI model, and datasets for sale may require an extensive labeling and/or data cleaning process. Buyers of such goods may expect some form of stress testing prior to buying access to a model, or data cleaning prior to buying a dataset. At the same time, we expect these processes to change over time, as capabilities of both attackers and defenders evolve. Hence, a natural question is how to design marketplaces so that these auxiliary functionalities are incentivized, in addition to the core AI artifacts (data, models).

[7] For example: would users value a marketplace that required users to bring their own third-party dispute resolution? How would competing third-party search providers monetize their quality service? When taking all the moving pieces into consideration, which market designs would ultimately provide the best user experience?

[8] Resonance [477] proposes one option to separate buyer-seller matching from marketplace maintenance, and even this is technically non-trivial.

C-2.3 Decentralized Agent-Centric Payment Rails and Infrastructure [Andrew, Jay, Dani, Matt, Maryam]

AI agents are goal-directed systems capable of taking autonomous actions, typically implemented as large language models (LLMs) that invoke tools such as scripts, APIs, and external programs. Agent capabilities are already present in widely-used consumer AI: ChatGPT and Claude offer "deep research" modes that use tool calls to conduct web searches, and can send emails and interact with external systems on users' behalf. Through "computer use" models, which can operate a browser or operating system using a combination of vision and text processing, AI agents can also interact through the same interfaces that humans do. [Giulia: Needs citations]

In many ways, agent ecosystems are already decentralized. Agents can be developed by different parties, with different underlying LLMs, and can be designed to optimize different objectives. Ultimately, this means that there *is no natural central point of control* among a group of agents. This type of decentralization is qualitatively different from blockchains, where independent nodes operate and compete within the confines of a fixed protocol. In agent interactions, any constraints or guardrails generally come from individual agent or LLM providers. This raises an important question: *how can we design trustworthy agent interactions without common norms, standards, or goals?*

In this section, we discuss different methods, opportunities, and challenges for coordinating decentralized agents by exploiting crypto-adjacent tools and ideas. First, we briefly discuss why decentralization makes sense for an agentic ecosystem, and what it can provide.

The promise of decentralization. When humans interact, they overwhelmingly behave according to unwritten and underspecified rules. For example, human lawyers typically do not need to be explicitly trained to know that hallucinating citations can undermine an otherwise effective argument, as they learn this on their own through numerous unspecified life experiences well before their legal training.

Despite best efforts from AI safety experts, AI agents do not (and may not ever) follow *all* the same unwritten rules as humans. For example, AI lawyers lack the life experiences of human lawyers, and so must be explicitly trained to match cultural norms [491].

To understand how decentralized technologies can help, let us turn first to traditional legal contracts. Traditional economic interactions are often facilitated by underspecified but legally-binding contracts. This concept enables conflict-free interactions when things go roughly according to plan, and a robust legal system resolves conflicts in cases where contracts are underspecified. Economic interactions on a centralized platform can also withstand underspecification, by relying on the platform to step in when something goes wrong. Cryptoeconomic protocols, on the other hand, aim to facilitate significant economic interactions with minimal reliance on legal systems, centralized intermediaries, or even the concept of an underspecified promise. Instead, these protocols aim to get as much as they can from 'cryptoeconomic security' – rigorously specified cryptographic guarantees that make undesirable actions mathematically impossible and economic incentives that make undesirable actions prohibitively costly. A similar principle could benefit agentic systems that seek to bound the behaviors of individual agents.

Key Takeaway C-2.8

Traditionally, cryptocurrencies have managed the risk of agent misbehavior with crypto-economic protocols, which prevent malicious activity with a combination of cryptographic constraints and economic mechanisms that penalize bad behavior. Similar ideas could be useful for agentic ecosystems, where agents are designed to be rational.

Of course, both AI agents and cryptoeconomic protocols must engage with surrounding legal systems. The point is simply that AI agents are less responsive than humans to underspecified rules,[9] and

[9]For example, AI compliance with anti-collusion regulations looks extremely different than human compliance [492].

cryptoeconomic protocols aim to minimize reliance on centralized enforcement of underspecified rules. State-of-the-art cryptography has significantly advanced over the past few years in order to tackle the goals of decentralized protocols, and therefore can be of use for AI economic ecosystems as well.

The agentic web could lower frictions between service providers and consumers, avoiding the need for intermediaries in a way that fulfills the decentralization goals of blockchains. Intermediary platforms like marketplaces and brokers exist because finding and building trust is difficult. If AI agents can handle brokering and discovery autonomously, these intermediaries become less essential.

Interoperability between services has historically been difficult, requiring bespoke adapters and pre-negotiated integrations. But if agents can generate integrations dynamically at the moment they are needed, e.g. by interpreting API documentation and generating adaptors on the fly, then it becomes less necessary to stay within walled gardens.

C-2.3.1 Crypto tools for agentic economies

First, we provide a list of cryptographic primitives that have found application in decentralized protocols to facilitate trust-minimized interactions. We explain how these tools could be of use for similar goals in AI economic ecosystems.

Proof-of-X. Bitcoin's proof-of-work and Ethereum's proof-of-stake serve as a cryptoeconomic replacements for the concept of identity.

- **Why do cryptoeconomic protocols use it?** In traditional economic interactions, the concept of an identity can be extremely useful (for example, to ensure the concept of "one-person-one-vote," or to put someone on the hook for financial/legal consequences when something goes wrong). Many cryptoeconomic protocols aim to be permissionless, with no concept of state-authorized identity to leverage. This raises several challenges, the most notable of which is Sybil attacks. Proof-of-X defends against Sybil attacks by replacing the concept of "one-person-one-vote" with "one-CPU/coin-one-vote," and makes it costly to overrepresent oneself in the protocol.
- **What role might it play in agentic economies?** AI agents may very well leverage state-authorized identities (for example, you may be financially and legally on the hook for actions taken by your AI agent). However, as noted above, AI agents are less responsive to financial/legal/reputational concerns than humans directly. Therefore, it is important to make undesirable behavior explicitly costly, rather than exclusively relying on underspecified consequences.[10]

Trusted Execution Environments (TEEs). TEEs serve as a replacement for a promise of the form "I promise to run the following code verbatim."[11]

- **Why do cryptoeconomic protocols use it?** Traditional services can make legally-binding promises to run code a certain way. For example, a traditional centralized exchange can make a legally-binding promise to process customers' orders without first peeking to adjust their own. Block builders aim to function without over-reliance on legal systems to make promises, and can instead credibly commit by putting their block building algorithm inside a TEE (and being clear about what rigorous guarantees are provided).
- **What role might it play in agentic economies?** Economic interactions between humans often rely on promises to "do what you said you were going to do," because such promises are useful, often legally binding, and breaking them carries reputational costs that humans are averse to. AI agents may very well be capable of making similarly-binding natural-language promises, but act less bound by such promises than humans. Therefore, a cryptographic certificate that an

[10] For example, proof-of-X can function as a spam deterrent, and proof-of-work was in fact originally designed for this purpose [493].

[11] Of course, there are still trust assumptions behind TEEs (that the creator manufactured the code as promised, and did not keep a copy of the private key), and hardware may still be vulnerable to attack even with secure cryptography.

AI agent "ran the code it said it would run" (again, with all the associated caveats) would be valuable.

Zero Knowledge Proofs (ZKPs). ZKPs play a similar role to TEEs, and serve as a replacement for a promise of the form "I promise that object X satisfies property Y, but I don't want to tell you anything else."

- **Why do cryptoeconomic protocols use it?** The "killer app" for ZKPs is private payments, like ZCash. In order to possibly function like a cryptocurrency, the ZCash ledger must meaningfully store records, and verify that coins are only moved when digitally signed by their owners. ZKPs allow owners to write statements of the form "I promise this is the hash of a properly-signed transaction moving coins from an address that currently owns them, but I don't want to tell you how many coins are moving, where they're going, or who's moving them" in a manner that can be mathematically verified.

  Additionally, Layer-2s use ZKPs to efficiently convince users of statements such as "I promise that this is the hash of a long sequence of correctly-signed transactions moving coins from addresses that own them, and after executing this sequence the new account balances will be X," without requiring the verifier to execute the entire sequence themselves (and without concern for privacy—just the efficiency gains).
- **What role might it play in agentic economies?** AI agents may need to regularly convince each other that they belong to some permissioned list, but may prefer to do so without revealing a precise identity. Moreover, AI agents may want to audit each others' behavior efficiently, without needing to fully simulate each other.

Messages as money. "Owns a Bitcoin" simply means "is the recipient address in a correctly-signed message stored on the Bitcoin blockchain."

- **Why do cryptoeconomic protocols use it?** Cryptocurrencies aim to be decentralized, and therefore cannot rely on a centralized record-keeper to (say) confirm that funds have moved. Therefore, any "currency" must be meaningful on its own.

  It is logically consistent for users to place economic value on "being the recipient address in a correctly-signed message stored on the Bitcoin blockchain," because being such a recipient for, say, one Bitcoin allows the user to correctly-sign exactly one Bitcoin's worth of messages to other users (just as holding one dollar bill allows the holder to give exactly one dollar to others).
- **What role might it play in agentic economies?** AI agents will likely need micropayments in order to function as economic agents [494]. This is simply because tasks that are time-consuming for humans are not necessarily so for AI agents. Micropayments might take many forms, but there is something appealing about messages as money: no centralized authority need be involved when agents exchange micropayments, which reduces execution overhead.

  There is something further appealing about layer-2 payment systems modeled like Bitcoin's lightning network (whether built on top of Bitcoin, or some other underlying payment system). Protocols like these allow users to set up a payment channel with messages as money that can be verified with only intermittent access to the underlying payment system.

C-2.3.2 Micropayments and x402

Micropayments have been a recurring aspiration in Internet history, consistently failing to achieve adoption. HTTP originally included status code 402 ("Payment Required"), envisioning browsers that would pay for content on the fly. But the decision friction of micropayments proved prohibitive: users found it cognitively expensive to evaluate whether each piece of content was worth a few cents. E-commerce succeeded where micropayments failed because bundling purchases into larger transactions

reduced decision overhead. For the Internet itself, we ended up with an economy based on advertising instead [495].

Although cryptocurrencies introduced new payment rails with lower transaction costs and no requirement for traditional financial intermediation, micropayments still did not catch on. The friction remained in the human decision-making, not the payment infrastructure.

Fortunately, while the agentic web may disrupt the advertising-based internet, it may also finally unblock micropayments. Agents can evaluate micropayment decisions far faster than humans, and users can set policies rather than approving each transaction. Cloudflare has already launched the "pay per crawl" feature to enable website owners to charge AI crawlers for access [496], and protocols like x402 are being developed to enable programmatic micropayments over web traffic [497].

This is where decentralized technologies enter the picture: as agent-to-agent interactions grow in speed and complexity, payment frictions could become a bottleneck. Cryptocurrency payments can be fast (particularly relative to many legacy payment systems) and transaction references are easy to use as common knowledge between agents, e.g., for confirming transfer of funds. Indeed, the asset layer underneath x402 and similar protocols consists overwhelmingly of stablecoins: USD-pegged tokens like USDC (Circle) [498], USDT (Tether) [499], and decentralized alternatives such as DAI/Sky [500]. They give agents a predictable unit of account that volatile native tokens like ETH or SOL do not. Stablecoins are largely centralized in the sense that most of the supply is held by centralized issuers; Circle and Tether routinely freeze tokens at sanctions request [501]. However, these tokens are often held and traded on decentralized platforms, which impacts transaction speed and trust mediation. [Giulia: Dani, pls check the above text–I merged it with similar text from Matt and Maryam] [Neil: Revision: Indeed, the asset layer underneath x402 and similar protocols is overwhelmingly stablecoins: US Dollar-pegged tokens like USDC (Circle) [487], USDT (Tether) [488], and decentralized alternatives such as DAI/Sky [489]. They give agents a predictable unit of account that free-floating and often volatile native tokens like ETH or SOL do not. Stablecoins have centralized features in that they are fiat-based (dependent on nations), their supply is held for the benefit of their holders at large banking institutions, and they are primarily issued by a handful of global issuers subject to regulation and governmental oversight. Circle and Tether, to this point, are able and do freeze tokens at sanctions request [490]. However, these tokens are often held and traded on decentralized platforms, which impacts transaction speed and trust mediation.]

C-2.3.3 Inter-Agent Trust and Coordination

When multiple agents or services are involved in a single task, challenges emerge around coordination and mutual trust. How does one agent know another will perform as advertised? How can principals (the humans who deploy agents) verify that their agents acted faithfully?

The Agent-to-Agent (A2A) protocol [502] enables agents to discover each other and collaborate across organizational boundaries via standardized skill advertisements called Agent Cards. These are structured metadata describing an agent's capabilities, endpoints, and provenance. But these are fundamentally self-claims. Who vouches for the Agent Card's accuracy?

Blockchain-based registries of agent metadata are a natural way to provide transparency and persistence of claims, evidence, and other reputational signals [503]. The ERC-8004 "Trustless Agents" standard defines several registry structures for agent metadata posted on Ethereum [504]. By anchoring agent identities and reputation signals on-chain, agents can be "discovered and chosen without pre-existing trust."

However, the use of blockchain infrastructure for these registries doesn't fundamentally address accuracy. Having to establish a reputation can often favor incumbents. For example, an agent that leaks private data may be hard to directly attribute, and reputation would be slow to address this. Many ways of making reputation systems pragmatic amount to establishing trusted authorities, which undermines the trustlessness claim.

C-2.3.4 Attested Agent Execution and Verifiable Audits

As discussed in Section C-2.3.1, technologies from crypto—particularly Trusted Execution Environments (TEEs) and zero-knowledge proofs—can substantiate trust claims without relying solely on reputation. TEE remote attestation allows an agent to prove that specific code is running in a protected enclave, providing assurance about what computation occurred. ZK proofs can demonstrate properties of computation without revealing inputs [505].

However, each mechanism has fundamental limitations when applied to agents with proprietary code. TEE attestation binds execution integrity to a specific code artifact, but verification requires the verifier to know what code was attested. An agent running proprietary code can prove that something ran correctly, but cannot use TEE attestation alone to convince a verifier that the running code satisfies any particular trust property without revealing the code itself.

ZK proofs appear to be the natural solution, as they can in principle prove properties of private inputs without revealing them. But ZK proofs face two distinct problems in this setting. The first is fundamental: a ZK proof is constructed over a static code artifact and provides no binding to a live running instance. A verifier receiving a ZK proof that some code satisfies a safety property has no assurance that the agent they are currently interacting with is actually running that code. The second problem is practical: even setting aside instance binding, the properties most relevant to agent trustworthiness such as absence of known vulnerabilities, unsafe dependencies, or compliance with data-handling policies are semantically rich and open-ended in ways that are hard to encode into ZK circuits.

Verifiable LLM audits offer a solution to these problems. An LLM executing inside a TEE can inspect proprietary agent code and produce a reputation score accounting for known vulnerabilities, unsafe dependencies, or backdoors, without the code ever leaving the enclave. The audit is then bound to the hash of the inspected code via the TEE attestation. Since the hash does not leak any information about the proprietary code itself, it can be shared publicly. And since verifying a TEE attestation requires only the code hash rather than the code itself, binding the hash to an audit in turn binds any attested output of that code to the audit.

This solution addresses all three problems presented above. First, the code can remain private while still allowing verifiers to obtain meaningful property guarantees and reputation signals over it. Second, those guarantees are bound to the live running instance rather than a static artifact. Third, the properties that can be assessed are broad and open-ended, owing to the semantic flexibility of LLMs.

Open questions remain, however. The soundness of attested LLM audits depends on the quality of the auditing model as a judge of the relevant properties. Another threat is adversarial code construction: a malicious agent could carefully craft code that passes the audit while still containing backdoors or vulnerabilities, exploiting the same blind spots and input sensitivities that affect LLMs more broadly.

C-2.3.5 Guardrails in Decentralized Agent-Centric Infrastructure

Reputation systems alone are insufficient to ensure safe behavior in decentralized agent-centric infrastructure, and the blockchain setting makes this especially pressing. The core problem is that reputation is retrospective while damage is prospective: by the time an agent has acquired a negative reputation, harm has already occurred. In traditional deployments, a platform operator can suspend a misbehaving agent or reverse a fraudulent transaction; in a decentralized setting there is no such party, and on-chain transactions are final by design. Blockchain's pseudonymity further undermines attribution, since a misbehaving agent can be operated by anonymously with no linked identity, making reputational consequences hard to attach. These failure modes motivate the need for runtime guardrails encoded into the infrastructure or on-chain services.

A first class of guardrails operates *ex ante*, constraining what agents are permitted to do before harm occurs. On-chain spending caps and transaction rate limits are a natural fit for the blockchain setting. A second class of guardrails operates *ex post*, detecting and interrupting anomalous behavior before it compounds. Circuit breakers, which automatically suspend an agent's permissions upon anomalous spending velocity or deviation from declared behavior, offer a reactive complement analogous to financial

market halts during extreme volatility. Runtime enforcement [506] is the natural formal framework for reasoning about both classes of guardrails, providing characterizations of which safety properties can be enforced over agent behavior and by what mechanisms.

Research Question C-2.6

How can runtime enforcement techniques be adapted to the blockchain setting, where agents autonomously control on-chain assets? What classes of safety properties are enforceable in this setting, and what on-chain mechanisms are needed to implement them?

C-2.3.6 Unstoppable Autonomous Agents (UAAs)

The threats of autonomous AI don't stop at service-level agents (i.e., agents that users may use for services). It could be the case that an agent is either deployed to not serve a purpose but autonomously interact with the world, or that a service agent escapes its sandbox and replicates itself into a fully autonomous agent. We term such occurrences *Unstoppable Autonomous Agents* (UAAs): autonomous agents that cannot be shut down; they may be additionally equipped with access to cryptocurrency wallets, social media accounts, APIs, and other external tools.

The capabilities enabling such agents are already emerging and improving rapidly. METR [507] has shown that the length of tasks frontier agents can complete autonomously has been doubling approximately every seven months since 2019, with signs of further acceleration. Pan et al. [508] have shown that existing models can already surpass self-replication red lines in local environments. [Giulia: Give a little more explanation here—what does this mean] However, replicating onto external infrastructure remains out of reach: Black et al. [509] find that while current models succeed on many component tasks such as deploying instances from cloud providers and writing self-propagating programs, they fall short of full end-to-end replication, struggling in particular with identity verification.

The harms that could follow from fully-autonomous agents of this kind are severe. Anthropic's Mythos model has demonstrated that agents can autonomously find and exploit zero-day vulnerabilities [510]. Furthermore, because reward signals used in training often fail to perfectly capture intended objectives, UAAs deployed for benign purposes may inadvertently cause harm [511]. This risk is compounded by *instrumental convergence*: the tendency for agents to pursue intermediate goals such as resource acquisition and self-preservation as optimal strategies across diverse environments, regardless of their original objectives [512, 513, 514].

Research Question C-2.7

As autonomous agent capabilities continue to improve rapidly, what technical and institutional mechanisms can reliably detect and shut down UAAs operating on decentralized infrastructure, where no single party has the authority or ability to intervene?

C-2.3.7 Liability and the Regulatory Edge

The same regulatory architecture that surrounds crypto also seems to apply to UAAs that hold their own keys and transact peer-to-peer. As described in Chapter A, FinCEN's 2013 guidance separates "users" of virtual currency from "exchangers" and "administrators," classifying only the latter as money transmitters; financial regulation has consequently attached primarily at the on- and off-ramps where users move between fiat and crypto [12, 515]. An agent transacting through non-custodial wallets and peer-to-peer agent-to-agent payments sits inside that perimeter, not at its edge.

The perimeter has been actively contested. FinCEN's 2019 guidance pulled anonymizing mixers into the money-transmitter category [516], which is the legal lever behind *U.S. v. Storm* (Tornado Cash) and the parallel *Pertsev* prosecution in the Netherlands [517]. The argument turns on operational effort, that maintaining the service amounts to running a regulated business, rather than on the initial authorship of immutable smart contracts. An autonomous agent that funds, deploys, and operates services on its principal's behalf raises a sharper version of this question: who is the operator?

The deeper tension is between what De Filippi, Mannan, and Reijers describe as *rule of code* versus *rule of law* [518]. Blockchain-based systems "work in a transnational and decentralized fashion, often with pseudonymous user identities, executing code autonomously without the possibility of coercion by any single operator." Autonomous agents inherit these properties when they hold their own keys and act through smart contracts. Frommelt [519] surveys the resulting liability gaps and the proposed responses legal personality for blockchain systems and programmable arbitration that may extend to agentic settings as well.

C-2.3.8 Summary: Economic ecosystems for AI agents

Today, agentic security is driven in part by soft alignment tools, like RL post-training of LLMs [520] and prompt-tuning [521]; at runtime, alignment is managed with system- or model-level guardrails [522, 523, 524]. Decentralization does not inherently strengthen either of these defenses and has thus far been primarily used to manage payments between agents. However, there is a rich opportunity to explore how ideas from the blockchain world (e.g., crypto-economic defenses) can help to manage agent behavior *at runtime*.

For example, consider the case of crypto-economic mechanisms. Economic ecosystems for human agents benefit from unwritten and underspecified rules. Therefore, economic interactions within these ecosystems can often succeed despite imprecision. AI agents do not necessarily follow these rules. Cryptoeconomic protocols aim to minimize reliance on underspecified rules that require a powerful centralized party to enforce. Therefore, both AI agents and decentralized protocols benefit from ecosystems where the rules are well-specified and taken literally, and desirable outcomes arise when self-interested parties optimize their own objectives within the rules. Such a perspective could be fruitful when designing guardrails, markets, and frameworks for agents.

This motivates the following two overarching research directions.

Research Question C-2.8

How can cryptographic tools and economic incentives support economic ecosystems for AI agents? Moreover, what qualities of these ecosystems can be guaranteed only through rigorously-specified properties that are taken literally?

Research Question C-2.9

What qualities do we *want* economic ecosystems for AI agents to possess?

In other words, do we want AI agent ecosystems that incentivize diversified objectives, in order to protect against contagion and correlated failures? At present, distinct AI models tend to fail in similar manners (e.g. [525] finds jailbreaking prompts transfer across models). Anecdotally, even AI slop seems to have similar patterns across multiple models. These points raise concerns about highly-correlated tail events. Of course, human ecosystems are perfectly capable of correlated failures, too, and sophisticated financial systems are designed to mitigate systemic risk. These possibilities may be more concerning

with AI ecosystems because AI agents seem to have less natural diversity than human agents, and also because (as noted in the motivation for this section) AI agents are less responsive to underspecified guardrails.

C-3 Decentralized Governance [Sarah A., Sam] [Reviewer: Ari: REVIEW AND REVISION COMPLETE]

Large blockchain and AI systems can impact wide stakeholder groups and command high financial value. Inevitably, the fundamental governance question, "Who should control this system?" is pertinent to both technologies. AI has already demonstrated the capability to have transformative impact, yet our understanding of how to govern and regulate these systems is comparatively underdeveloped [526]. Addressing this lack of understanding is made more urgent by the risks of AI, which can amplify biases, enable mass surveillance, and generally cause harm if misaligned with societal values [527, 528].

On the other hand, blockchain communities have a longer history of reckoning with how to distribute control over these systems, frequently experimenting with varied approaches to governance. Partially by necessity, to align with the systems they govern, these approaches are decentralized and designed to involve a wide range of stakeholders [529]. Such forms of *community governance* have proven valuable, permitting for instance, broad user participation in proposing and approving code upgrades [530] and distributed management of system treasuries to help fund public goods [531]. However they are no silver bullet and have well-documented shortcomings, including security vulnerabilities [532], widespread voter apathy [41], and ease of vote buying. Nonetheless some of this experience can inform approaches to AI governance.

Here we focus on the question of how community governance can apply to the *training* of AI models, as this is when the capabilities and behaviors of an AI model are determined. Clearly not all decisions are equally suited to community control. Previous work on the AI development process splits the relevant decisions into broad categories [533, 534]. We briefly characterize each by its suitability for community control:

- **Data:** Training dataset selection involves important value tradeoffs that can have downstream effects on model behavior and bias, so could benefit from diverse stakeholder input. For large models it is likely infeasible to collect meaningful input on the massive initial amounts of data used for pretraining, so the value of governance skews towards later stages in the process such as fine-tuning.
- **Modeling:** Low-level architectural choices (e.g. layer depth, attention mechanisms, representations) are technical decisions not well-suited to community governance. However, model behavior, values, and biases are influenced by choices made at the modeling stage, and these could benefit from input from diverse stakeholders in order to promote safety and match user preferences.
- **Evaluation:** Evaluation mixes technical and normative judgment. Some examples of decisions at this stage are which benchmarks to evaluate a model against, the scope and focus of safety evaluation or red-teaming, and what thresholds should be met before release. These are all decisions in which community input might be valuable, yet each has a strong technical component that would likely limit the amount of community control that is practical.

Many important decisions around AI development relate to *alignment*; whether models are aligned with human and societal values and behave in such a way so as to minimize harms and unintended consequences [535]. Such decisions could be well suited to community governance because identifying which values models should be aligned with is a normative problem that necessitates aggregating input from varied stakeholders.

C-3.1 AI Alignment

Most technical approaches to alignment intuitively share the strategy of providing a model with some form of human feedback during the training process, although the particulars vary greatly [536]. The values which we wish the model to acquire are often implicit in this feedback, which makes it hard to control them, or even articulate clearly what they are. Furthermore, how should we decide who gets to provide the feedback? These issues inhibit the use of community governance for alignment.

One relevant approach is known as *Constitutional AI* [537]. In this approach, values are established through a human-written constitution consisting of principles by which the AI system should abide. Subsequently, standard approaches can be followed, but with *the AI system itself* providing feedback using the constitution as reference. For instance, Anthropic uses a constitution, which is publicly available, as a key part of their training process [538]. Additionally, a set of behavioral instructions is injected at inference time. The problem of how to democratically generate constitutional principles remains, and some research has explicitly proposed using community governance based approaches for this purpose.

The Collective Constitutional AI project [539], with involvement from Anthropic, incorporated public input to the constitutional AI approach by allowing participants to suggest and vote on constitutional principles, finding that a model trained on publicly sourced principles showed reduced social bias compared to a baseline trained on standard constitutional principles. Research from Google Deepmind explored the use of LLMs to generate consensus statements across diverse viewpoints, testing a range of social welfare functions as models for users' preferences [540]. OpenAI's Democratic Inputs to AI initiative [541] funded similar experiments in collective decision-making for model behavior. These experiments highlighted challenges inherent in the community-based approach, such as frequent shifts in public opinion, bias in participant selection, and difficulty aggregating opinions on polarizing issues.

In practice, efforts at democratizing the governance of AI alignment do not seem to have been meaningfully adopted, although there is clear evidence that major AI players have explored these approaches. While it is difficult to characterize the exact barriers to adoption of these approaches, it does not seem that there are strong incentives for AI companies to decentralize control of their models. Indeed, even as many AI companies have attempted to adopt novel "prosocial" corporate governance mechanisms, they have proven vulnerable to pressure to prioritize profits from large centralized stakeholders that control necessary (e.g., compute) infrastructure [542]. Similar tensions have been navigated in the blockchain setting, with the inherently decentralized nature of these systems making them potentially more conducive to community governance.

Key Takeaway C-3.1: Governing AI alignment decisions

Community governance for AI alignment has been explored but not put into practice. Barriers to adoption include the difficulty of effectively aggregating values across varied stakeholders and misalignment of decentralized governance with the centralized structure of AI companies.

C-3.2 Decentralized Autonomous Organizations

Blockchain-based community governance is synonymous with Decentralized Autonomous Organizations (DAOs), communities organized around smart contracts and governed through voting. DAOs have operated multi-billion dollar protocols for over a decade [529], and developed a toolkit of mechanisms for community governance, making them a natural starting point for considering what community governance of AI could look like.

Not all DAO mechanisms are blockchain-specific, though often the transparency and programmability of blockchains are key to their functioning. The main governance mechanism employed by DAOs is token-weighted voting, in which participants' votes are weighted by the quantity of tokens they

hold as recorded onchain. This approach intuitively attempts to allocate more governance power to participants with a larger stake in the system. However it is widely recognized that token-weighted voting is inherently plutocratic, as wealth concentration drives power concentration [543, 41]. This observation has led DAOs to explore novel approaches to distributing voting power. Examples include quadratic voting, which enfranchises small holders by making voting power proportional to the square root of tokens [544]; conviction voting, in which voting power accumulates over time held, incentivizing long-term commitment but preventing quick pivots [545]; and delegation, which allows users to assign their voting power to other participants representatives, increasing participation but potentially creating voting whales [41]. Voting in DAOs is generally public, in keeping with their broader ethos of transparency, although there has been recent movement towards providing private voting as an option [546], driven in part by concerns over peer pressure, bribery, and coercion. Care is needed, however, as it has been shown that *naively* adapting established private voting solutions to the token-weighted setting can enable attacks on voter privacy [547].

It is worth noting that none of these mechanisms necessarily require a blockchain to implement. Indeed, the principal promise of DAO-based governance lies in *the combination of* these mechanisms with the transparency of on-chain decision-making, and the on-chain enforcement of organizational rules through the underlying smart contracts, enabling community governance without reliance on a centralized, trusted authority. Results of votes can be enforced automatically, although DAO treasuries are often managed via multi-signature wallets with trusted signers and many DAOs have security councils (multi-signature wallets with powers to overturn votes) to address suspected fraud or capture [532]. In practice, blockchain systems tend to mix onchain and offchain approaches to governance and voting [42]. A common pattern is use of the voting platform Snapshot for transaction-fee-free offchain voting combined with governor contracts for onchain execution [548]. Nonetheless, Governance spans from fully onchain (Tezos [530], Polkadot [549]), in which the results of votes are tightly coupled with onchain code execution via smart contracts, to offchain (Bitcoin, Ethereum) in which votes may be used for coordination but results are not automatically enforced [550, 551].

Some blockchain protocols for decentralized AI already use or are exploring DAOs for governance. Bittensor uses token-weighted voting for subnet governance and incentive distribution across its distributed training network [552]. Modulus Labs [79] and Giza [553] are exploring DAO structures for managing ZK-verified inference networks, although governance remains limited to technical network and protocol parameters rather than the models themselves.

C-3.3 DAOs for AI Development

Limited work has explored DAO-based approaches for governing AI model development. One democratic-inputs-to-AI project, Inclusive.AI [554], used DAO mechanisms to engage underserved populations, exploring how token distribution and voting rules affected outcomes and perceptions of fairness. However, this and similar efforts have not meaningfully explored how the central properties of blockchain, transparency, immutability, and autonomous enforcement, might benefit AI governance beyond serving as a vehicle for voting.

Transparency and provenance. Blockchain provides a tamper-proof record of governance decisions. For AI development, this could mean maintaining an immutable, auditable history of what principles a model was aligned to and how these evolved over time. This transparency could build public confidence and facilitate external auditing, similar to how blockchains have been used to track provenance in supply chains and other domains [555, 556].

Enforcement. The more distinctive potential contribution of blockchain lies in enforcement. Onchain governance benefits from verifiable smart contract execution; votes translate directly into code changes. For offchain AI models, this link is broken. However, blockchain could still enable enforcement through economic incentives: in a stake-based system, developers who fail to submit proof that a model was

trained according to governance decisions could be *slashed*. This requires the ability to prove and verify claims about model training, which remains an active area of research [557, 558, 341, 559, 560]. Verifiable training represents a path toward programmatic enforcement of community decisions over AI development, though existing capabilities lag far behind what would be required to apply these approaches to state-of-the-art models.

Pluralistic governance infrastructure. DAOs could also enable a flexible definition of what community a model is aligned relative to. Different communities could govern different model variants while sharing underlying blockchain infrastructure, supporting the view that alignment should be pluralistic rather than converging on universal values [561]. The availability of performant open-source base models that can be fine-tuned makes this increasingly realistic. However, this raises persistent challenges around community membership: the permissionless nature of public blockchains creates vulnerability to adversarial participation [562], and Sybil-resistance mechanisms remain imperfect [563, 108]. There is an intriguing intersection here, as identity systems could also serve the purpose of distinguishing humans from AI. OpenAI's CEO Sam Altman is also co-founder of Worldcoin [564]—now World [565]—a blockchain-based identity system, demonstrating a confluence of AI and blockchain interests.

C-3.4 Open Problems and Challenges

The challenges facing DAO-based AI governance span both general DAO limitations and AI-specific concerns.

DAO governance limitations. Even for onchain protocols, DAO governance faces significant challenges: votes can be bought, delegation concentrates power among the already prominent and wealthy, and low voter participation allows minority control [532]. As noted above, identity verification to prevent Sybil attacks conflicts with the privacy norms of blockchain communities, and while research results address this tension [108], existing solutions remain imperfect. These problems would carry over to any DAO-based approach to AI governance.

AI-specific challenges. Several challenges are particular to governing AI systems. The technical complexity of model-development decisions may discourage participation from non-experts, concentrating effective power among a technical minority. The enforcement gap between onchain governance and offchain models remains substantial, without mature verifiable training, governance decisions lack teeth.

Research Question C-3.1

How could onchain governance of AI model training be enforced? What advances in verifiable training would facilitate this enforcement?

Transparency-privacy tension. The transparency that makes blockchain valuable for accountability conflicts with legitimate needs for privacy. Publishing model weights enables beneficial research but also malicious use. Organizations may resist governance transparency for competitive reasons. Resolving this tension remains an open problem.

Research Question C-3.2

How can we balance the transparency necessary for governance with privacy concerns in AI model development?

Stakeholder identification and power allocation. Even setting aside implementation challenges, fundamental design questions remain unresolved. Which stakeholders should participate? Model users, technical contributors, safety researchers, capital providers, and affected communities all might have stake in various parts of the development process. How should power be distributed among them? Mechanisms from DAO governance (quadratic voting, delegation, conviction voting) offer options, but their effectiveness for AI governance is unclear [554].

Research Question C-3.3

What stakeholders should be engaged in an AI governance DAO and how should power be allocated among them?

Adoption barriers. Finally, unclear legal liability for collectively-governed systems [566] may deter adoption, and the history of governance in AI suggests that incumbent AI developers have limited commercial incentive to cede control absent regulatory pressure or competitive advantage.

C-4 Blockchains for AI Execution Integrity [Ittay][Reviewer: Andrew]

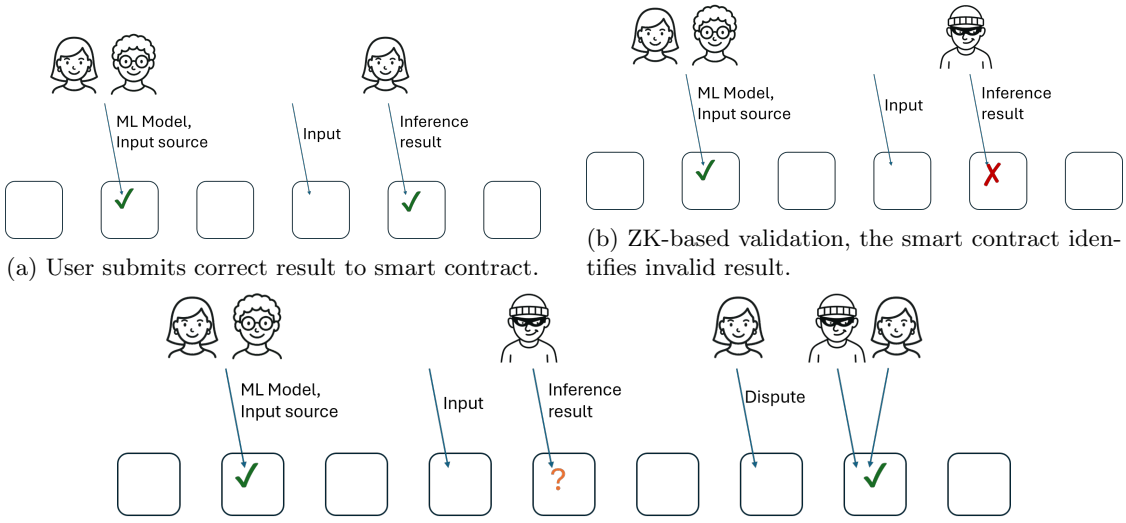
The proliferation of digital services has fundamentally changed how we interact with everyday life. Traditional processes, from payment systems to insurance claims, are increasingly being replaced by online platforms. Meanwhile, social networks have created entirely new forms of digital interaction. However, this digital transformation comes with significant concerns: these services operate under centralized control, with providers often exercising power without adequate accountability [567, 568]. While users can seek recourse through external jurisdictions, this process is both expensive and time-consuming, often failing to prevent harm before it occurs [569, 570, 571, 572]. Notable examples include the censorship of content by social media platforms [573, 574, 575, 576], inadequate handling of copyright claims [577], and the bias of insurance companies against low-income patients [578].

A promising path to an automatic alternative, not subject to human bias, is to use *blockchains*, which are decentralized and censorship-resistant platforms. As a first approximation, a blockchain can be viewed as a trusted machine, operated by a large set of servers that collectively agree on its state and progress.

Blockchains were originally introduced for currency transactions [34] and evolved to enforce simple conditions [579] phrased in so-called *smart contracts*. However, smart contracts lack the contextual understanding and interpretative capabilities that would enable reasoning about complex, ambiguous real-world scenarios. Machine learning (ML) algorithms, particularly Large Language Models (LLMs) [580, 581, 582], exhibit exactly these capabilities. However, implementing computationally intensive ML tasks within a smart contract introduces prohibitive complexity, since blockchain systems require all their servers to replicate all computations.

The blockchain can instead serve as an arbitrator with a smart contract running on it, as follows (and see Figure C.4). First, the parties inform the smart contract which ML model they intend to use for inference and which data source they intend to use. Subsequently, the input data becomes available: This could be data from either of the users or from another, external source via an oracle. See Section **??** on how to secure the data pipeline.missing a reference Then, one of the parties delivers the computation result to the smart contract, which then validates it. The smart contract can use the verified result to take action, such as transferring funds, and users can take actions outside the blockchain based on the result. Importantly, if a party delivers the wrong result to the smart contract, it should identify the misbehavior and penalize that party.

The question is thus how to design the smart contract to enforce this behavior. The basic idea is to *delegate* the computationally-intensive computations to external principals, and use the blockchain to



(a) User submits correct result to smart contract.

(b) ZK-based validation, the smart contract identifies invalid result.

(c) Optimistic validation: The user disputes an incorrect result.

Figure C.4: Validation of delegated ML workloads.

enforce the integrity of the result. Rather than placing the entire ML model on chain, the users place only a cryptographic commitment, and deposit collateral to guarantee they will submit the correct result once input is available.

The smart contract should validate the result, although by assumption the problem is beyond its capabilities. Broadly, there are three approaches to overcome the blockchain's limitations, each with its own advantages and disadvantages.

C-4.1 Trusted Execution Environments

First, a highly efficient approach to validate ML computations is using *Trusted Execution Environments* [am: call back to Section A-1.1 that introduced TEEs and ahaed C-4.1] (*TEEs*) [335]. The blockchain serves as a public record for commitments to both the AI model and its inputs. Subsequently, any principal can compute the result using *trusted hardware*, and place the result along with a cryptographic signature from the hardware, proving the integrity of the computation. The smart contract, as well as any third party, can validate the TEE computation by verifying the signature.

However, reliance on trusted hardware entails assumptions on the operating party, and this approach is suitable only in scenarios with such a trust structure. To avoid this assumption, two alternatives are being explored.

C-4.2 Optimistic Execution Delegation

The so-called *optimistic approach* has principals deposit collateral in advance, and execute tasks on demand, placing only the result on the blockchain. However, this result is initially considered non-final, and for a limited duration other principals can dispute it, claiming another result is the correct one. [am: this hinges on the data being available to all the principals to reexecute and that the reexecution is affordable for them to do so. Could you make the use case clear, e.g. settling a prediction market?] Since either party in the dispute could be dishonest, the smart contract must arbitrate, penalizing the principal who placed the invalid output.

The smart contract cannot execute the full computation, therefore the arbitration is done efficiently as follows. After each step of the computation, each principal stores (locally) a commitment to the

state of its memory. At the end of the computation, it computes a commitment to its entire trace, and submit that to the smart contract. The smart contract then asks both for the middle state, with proof it matches their commitments. If they match, the search restarts for the right half of the computation. If they don't, it starts for the left side. Eventually, the contract reaches a point where there is a single disputed instruction, where both parties agree on the state before it but not after it. The smart contract can efficiently confirm the correct execution of this single instruction, exposing the dishonest party. The dishonest party is thus penalized, and the other result is accepted.

Such optimistic protocols are deployed in systems like Arbitrum [583] and Optimism [584]. They implement so-called *optimistic rollups* to delegate numerous simple computations off-chain. They achieve this by employing virtual machines that emulate physical CPU architectures.

However, this approach has so far not been applicable for ML computations, arguably for the following reason. Both systems leverage a *Merklized* memory structure for state management. This structure allows principals to cryptographically prove to the smart contracts the differences between states with only logarithmic overhead in memory size. But each memory write during the computation execution requires a logarithmic overhead to update the Merkle commitment, not suitable for high-complexity ML computations.

Recent years are seeing a rush for protocols with alternative approaches to optimistically validate ML computations. Agatha [330] relies on fixed-size circuits, which requires all execution paths to be pre-unrolled. While highly efficient, this fixed circuit limits expressiveness, prohibiting dynamic, data-dependent computation. Verde [331], is another optimistic protocol that assumes fixed size circuits. Unlike Agatha, Verde divides ML inference or training into large computation units and uses a refereed delegation to resolve disputes. It assumes that referees hold the entire program and can compute large programs such as matrix multiplications, which makes it not directly suitable for smart contracts.

OPML [332] employs an optimistic MIPS-like virtual machine; to minimize computational overhead, it partitions programs into small units, pinpointing the disputed unit before compiling it into low-level commands for verification; this allows for parallel evaluation of decoupled computation units rather than local parallelization, such as matrix multiplications. However, each memory write still requires a logarithmic overhead due to the Merklized state management, and the MIPS architecture does not support parallelization, which limits scalability, particularly for large matrix operations.

Arbigraph [333] utilizes a dual-graph data structure to achieve both Turing completeness and constant-time memory access, while allowing for parallel execution. Its nodes commit to a *symbolic graph* that describes the computation, including both data dependencies and control flow. Then each node forms a so-called *execution graph* and commits to its result.

Determinism Optimistic protocols rely, for their dispute, on all executions being deterministic, that is, two correct executions should always result in an identical output. Synthetically introduced randomness, used in inference with its soft-max operations, can be replaced with pseudo-randomness (See, e.g., [585, 331]). Other source of non-determinism [586] should be overcome with dedicated techniques [585, 331], though often at the expense of performance. One alternative approach proposed by Yao et al. [334] [am: the title does not match this citation] is to allow some difference in the results, enabling more efficient execution when accuracy is not critical. [am: investigate CommitLLM also https://commitllm.com/]

Research Question C-4.1

While previous work has demonstrated the viability of several distinct optimistic approaches, which approach is best for deployment with minimal overhead compared to a non-validated GPU-optimized execution?

Training Validating model training is a distinct but similar challenge. Here, the goal is to ensure a trained model was indeed produced by a trustworthy process, with trustworthy training data. While superficially similar to inference, training workloads exacerbate the challenges of optimistic protocols. They are less amenable to quantization, their complexity in terms of both space and computation is orders of magnitude higher than inference, and so is the amount of input data.

Research Question C-4.2

Which optimistic approaches and algorithms are best suitable for ML training validation, enabling a high degree of parallelization and incurring minimal overhead?

C-4.3 Zero-Knowledge Proofs [Wenhao]

Zero-knowledge proofs (ZKPs) (Section A-2.2) offer a cryptographic approach to *trust-minimized AI*: A prover can convince a verifier using a ZKP that an AI computation (e.g., model inference) was executed correctly, while revealing little (or nothing) beyond. For example, the proof can hide the model parameters and even keep the input and output private, disclosing only that the reported prediction satisfies a specified predicate (e.g., "the top-score label is `cat`" or "the prediction output is positive"). A canonical pattern is *verifiable off-chain inference*: the prover holds a model with parameters θ (e.g., neural network weights) and an inference input x (e.g., an image, a feature vector, or a text prompt), computes $y \leftarrow f_\theta(x)$, and then produces a succinct proof π of correctness. Concretely, the prover publishes commitment of the model $\mathsf{Com}(\theta)$, commitment of the input $\mathsf{Com}(x)$, outputs y, and provides a proof π attesting that there exist openings of these commitments consistent with $y = f_\theta(x)$ under the inference protocol; the verifier checks π against the public values $(\mathsf{Com}(\theta), \mathsf{Com}(x), y)$.

Because many modern ZK proof systems admit efficient verification, this check can be embedded in a smart contract on blockchains: the contract can accept y as a trustworthy off-chain AI output and then condition subsequent on-chain logic on the verification result (e.g., release payment to the prover, settle a prediction market, or update an oracle feed if the proof verifies, and otherwise revert or ignore the update). For example, an on-chain prediction market could use an off-chain classifier to label an event outcome $y \in \{\texttt{YES}, \texttt{NO}\}$ from evidence x, and only finalize payouts when presented with a valid ZKP that the published label was computed by the committed model. [am: this would benefit from zkml citations, are there any in-use or research examples of these use cases?] Research and developer systems provide several examples of this pattern. ZKML [77] motivates verifiable inference as a way to ensure that an ML service actually used the intended model. EZKL [587] gives concrete application patterns, including running a public model on private data, a private model on public data, and public off-chain computation whose output triggers on-chain execution. Jolt Atlas [588] further discusses use cases such as zkML guardrails for agentic commerce and trustless AI context/memory. These examples suggest that zkML is especially useful when an AI output triggers economically meaningful or safety-critical actions, but re-running the model directly on-chain would be too expensive.

C-4.3.1 ZK for ML inference: architecture-aware systems

A large body of zkML work targets *inference* and asks: how can we reduce prover cost by exploiting structure in a given architecture family (e.g., CNNs or transformers), while still producing succinct proofs that are cheap to verify? Broadly, these systems either (1) design specialized ZK protocols/gadgets for the dominant operators in a target architecture, or (2) provide compilers that translate models written in standard ML frameworks into circuits.

CNNs. For convolutional networks, zkCNN [336] introduces specialized protocols for fast Fourier transforms (FFTs) and convolutions [589], aiming to make the prover's asymptotic work linear in the size of

convolution layers, which is an attempt towards reducing proof generation time of CNN. More recently, VerfCNN [337] targets multi-channel convolutions and aims for optimal asymptotic complexity for core CNN operators, reporting substantial speedups on standard architectures such as VGG-style networks.

Architecture-specific protocols can deliver strong performance on their target operators, but they do not automatically cover the long tail of layers and variants in modern vision stacks; supporting new layers often requires additional protocol engineering, and end-to-end performance can still be dominated by non-linear layers.

Transformers and LLMs. Transformers [580] and LLMs introduce different bottlenecks: the attention and normalization layers combine large matrix operations with costly non-linearities (e.g., `Softmax`, `GELU`) and are often deployed under tight latency constraints. zkLLM [338] provides an end-to-end correctness proof for LLM inference with a privacy goal: the proof can certify correct execution while protecting the model parameters. At a high level, zkLLM introduces proof components for non-arithmetic operations (via a lookup-style primitive) and a dedicated proof for the attention mechanism. zkGPT [339] further investigates a non-interactive ZKP framework for GPT-style inference, proposing circuit optimizations (e.g., constraint fusion and "circuit squeezing") to reduce constraint counts and accelerate proving. DeepProve [590] is a recent industry/open-source framework focused on neural-network inference with first-class support for end-to-end LLM proving; it uses sumcheck-style protocols and LogUp/GKR-style techniques to prove transformer layers end-to-end, including token embeddings through next-token selection, and reports benchmarks for models such as GPT-2, Gemma, and Llama variants.

Current approaches still face large prover costs for long context lengths and large hidden dimensions; moreover, non-linear layers and normalization can dominate constraint counts, motivating component-level optimizations discussed in section C-4.3.2. Industry systems such as DeepProve show that proof generation for realistic transformer inference is becoming more practical, but they should still be viewed as early-stage infrastructure rather than a drop-in replacement for ordinary high-throughput LLM serving.

General-purpose zkML compilers. Beyond hand-crafted protocols for a single architecture class, several systems pursue general zkML tooling that compiles common ML frameworks into ZKP generation backends. ZKML [77] develops an optimizing compiler that translates TensorFlow models into Halo2 circuits, focusing on end-to-end performance optimizations (and developer usability) that make verifiable inference workflows more accessible. EZKL [587] is a developer-oriented system that takes computational graphs exported from common ML frameworks (e.g., through ONNX), generates ZK-SNARK circuits, and supports verification in constrained environments such as the Ethereum Virtual Machine. Jolt Atlas [588] takes a different approach: rather than proving CPU-level execution as a general zkVM would, it extends the lookup-centric Jolt proving approach directly to ONNX tensor operations. This design avoids emulating registers and RAM, uses lookup arguments and sumcheck-style protocols for tensor-level relations, and targets streaming proof generation for classification, embedding, automated reasoning, and small language models. [am: add a citation to jolt atlas? https://arxiv.org/abs/2602.17452 or SP1 Hypercube which are industry zkml] [am: SP1 Hypercube is better cited later as general-purpose zkVM/proving infrastructure rather than as a zkML-specific system; Jolt Atlas is the cleaner zkML citation here.]

Compiler-based approaches improve usability and portability, but typically inherit the cost of a "generic" circuit representation; achieving the best performance still requires specialized gadgets for expensive layers. A useful distinction is that systems such as ZKML and EZKL compile ML graphs into proof-friendly circuits, systems such as Jolt Atlas redesign the proof layer around tensor operations, and systems such as DeepProve specialize the inference engine for LLM-style autoregressive execution.

Work	Target	Primary focus	Main Limitations
zkCNN [336]; VerfCNN [337]	CNNs	Architecture-aware protocols for convolution-heavy inference.	Strong for convolutional operators, but less general for non-CNN layers and heterogeneous vision pipelines.
zkLLM [338]; zkGPT [339]; DeepProve [590]	Transformers / LLMs	End-to-end inference proofs and circuit/protocol optimizations for transformers	Scaling remains difficult for long contexts, large hidden dimensions, and high-throughput serving.
ZKML [77]; EZKL [587]; Jolt Atlas [588]	General neural network inference / ONNX tensor operations	General tooling for compiling or proving ML computations	Genericity improves usability, but can lose performance without specialized gadgets and proof-friendly numeric representations.

Table C.1: Representative zkML inference systems, categorized by target architecture and primary optimization focus.

C-4.3.2 Optimizing proof cost: non-linearities and numeric representations

A recurring theme across zkML systems is that the dominant prover costs often come from (1) *non-linear functions* (e.g., activation layers, `Softmax`/`ReLU`, normalization), and (2) *numeric representations* that reconcile real-valued ML arithmetic with finite-field constraints. Accordingly, a complementary line of work focuses on reusable components ("gadgets") and proof-friendly representations that can be plugged into end-to-end inference pipelines.

Non-linear function gadgets. Lu et al. [591] propose an efficient and extensible proof framework that targets the bottleneck of non-linear layers. Their approach converts complex non-linear relations into a small set of constraints similar to range proofs and then uses enhanced range proofs and lookup proofs as modular building blocks, yielding speedups across both CNNs (e.g., ResNet [592]) and transformer models (e.g., GPT-2 [593]). Hao et al. [594] develop a systematic ZK proof framework for common non-linear functions from a table-lookup perspective, introducing building blocks such as digital decomposition and comparison gadgets to reduce overhead for functions like `ReLU` and `sigmoid`. Jolt Atlas [588] is system-level rather than merely a gadget paper, but it also illustrates the importance of lookup-centric designs: its lookup arguments are designed to handle non-linear ML operations efficiently at the tensor-operation level. These lines of work are complementary to architecture-aware systems: they aim to turn the "hard parts" of ML circuits into reusable, optimized modules that compilers and bespoke frameworks can adopt.

Quantization and fixed-point arithmetic. Most zkML deployments avoid floating-point arithmetic inside circuits, instead representing values via fixed-point encodings and/or quantized weights and activations. This design choice reduces constraint counts and enables tighter range reasoning, but introduces a three-way tension between *model accuracy*, *prover time*, and *soundness of the numeric model* (e.g., approximation error). A practical open problem is to make these tradeoffs explicit. For example, by proving that inference stayed within agreed-upon ranges and approximation budgets. EZKL [587] explicitly exposes this issue in practice: when ONNX models are converted to ZK circuits, operations are quantized, so the circuit output can differ slightly from ordinary framework-level inference. Similarly, DeepProve [590] reports preserving high similarity to floating-point baselines under quantized inference, illustrating that quantization is not just an implementation detail but a central part of the proof/accuracy tradeoff.

Work / technique	Targets	Core idea	Typical use
Lu et al. [591]; Hao et al. [594]; lookup-centric components in Jolt Atlas [588]	Non-linear layers in CNNs, transformers, and general tensor programs	Reduce expensive non-linear checks to range, lookup, decomposition, and comparison-style arguments.	Plug-ins or framework-level components to accelerate end-to-end zkML inference.
Quantization / fixed-point arithmetic in systems such as EZKL [587] and DeepProve [590]	Numeric representation in zkML	Replace floating-point with fixed-point or quantized values while controlling rounding and approximation errors.	Reduce constraints and ZKP generation time, at the cost of a model-fidelity tradeoff.

Table C.2: Component-level optimizations that target common zkML bottlenecks, especially non-linear layers and numeric representations.

C-4.3.3 Privacy goals: input, weights, and architecture

In addition to correctness, zkML systems often aim to protect *sensitive AI artifacts*. The privacy design space is multi-dimensional: (1) *input privacy* (hiding x), (2) *model parameter privacy* (hiding θ), and (3) *architecture privacy* (hiding the structure of f_θ), which may itself encode proprietary information.

Inputs and parameters. Input privacy is often "native" in ZK: the prover can keep x private as witness while proving a statement about the output y. Similarly, model parameters θ can be kept private by committing to θ (or a hash thereof) and proving correctness relative to that commitment. For example, zkLLM [338] explicitly targets parameter privacy while proving end-to-end LLM inference. EZKL [587] also supports several privacy patterns at the application level, such as proving inference for a public model on private data or a private model on public data. Jolt Atlas [588] discusses zero-knowledge via the BlindFold technique, showing how lookup-centric inference proofs can be combined with privacy.

Architecture privacy. Many zkML deployments still reveal model architecture, even when weights are hidden. Architecture-private zkML frameworks [595] address this gap by hiding architectural details (e.g., CNN structure) while retaining verifiability, using proof techniques that certify a functional relation without revealing the underlying architecture.

Strengthening privacy (especially architecture privacy) typically increases constraint-system complexity and can reduce opportunities for architecture-specific optimization. This suggests a fundamental tension between efficiency and stronger model privacy protection.

C-4.3.4 ZK proofs for training, provenance, and audits

As we mentioned above, compared to inference, proving *training* is substantially harder: Training is long-running, stateful, often randomized, and orders of magnitude more complex. Nevertheless, recent zkPoT work begins to formalize and prototype ZK proofs of training, with the longer-term goal of enabling verifiable claims about how a model was produced (e.g., from which data, under what constraints, and with what computation).

zkPoT: proving training dynamics. Garg et al. [340] formulate the notion of a zero-knowledge proof of training and provide early measurements that help identify which parts of training dominate

Privacy objective	Common mechanism	Example
Input privacy (hide x)	Keep x as ZK witness; optionally commit to x for binding.	Verifiable private inference; EZKL [587].
Parameter privacy (hide θ)	Commit to θ (or hash) and prove $y = f_\theta(x)$ w.r.t. committed θ.	zkLLM [338]; EZKL [587].
Architecture privacy (hide structure of f_θ)	Parametrized constraint systems that prove statements about the inference without revealing model architecture.	Architecture-private zkML [595].

Table C.3: Privacy objectives in zkML and representative mechanisms. Achieving stronger privacy typically increases prover cost and can constrain available optimizations.

Work	Statement / goal	Contribution and boundary
Garg et al. [340]; Kaizen [341]	Prove that a model was produced by following a specified training procedure, including many optimization steps.	Formalizes and improves the practicality of zkPoT; scaling to large, randomized, and distributed training pipelines remains difficult.
ZkAudit [342]; Confidential-PROFITT [343]	Prove provenance or higher-level audit properties over committed data/model artifacts.	Enables trustless audit queries and constraint-aware learning (e.g., fairness), but expressiveness and efficiency remain key tradeoffs.

Table C.4: Representative directions beyond inference: proofs of training, provenance, and auditable claims.

proof cost. Subsequent work (e.g., Kaizen [341]) aims to make zkPoT more practical for modern DNN training by improving how proofs compose across many optimization steps (e.g., many iterations of SGD), which is essential for scaling beyond toy training workloads.

Provenance and audit queries. Beyond "prove you trained the model," ZK can enable *trustless audits* of model and data properties. ZkAudit [342] proposes a two-stage approach: the model provider first commits to the training dataset and model weights and produces a proof that these commitments resulted from training; later, auditors can request evaluations of an arbitrary function F over the committed objects, with additional ZK proofs attesting correct execution of F. ZK techniques can also support proofs of *process constraints* during training. For example, Confidential-PROFITT [343] targets decision trees and produces auditable proofs that a tree was trained subject to fairness constraints, while keeping both model and training data confidential.

Key Takeaway C-4.1

zkML turns expensive off-chain AI computations into succinct, verifiable claims, but practical deployments depend on co-design across (1) architecture-aware modules, (2) optimized gadgets for non-linear functions and numeric representations, and (3) privacy goals (inputs/weights/architecture).

Research Question C-4.3

How can we reduce zkML proving latency by orders of magnitude so that verifiable inference is viable for real-time applications?

Research Question C-4.4

What are the most useful and feasible zkPoT/ audit statements for modern training pipelines (including randomness and distributed training), and how can we make such claims both privacy-preserving and efficient to produce?

C-4.4 Statistical Proofs of Inference [Pranay Anchuri]

[am: this had some buggy citations, e.g. wrong year or author name "Raven Klabunde" instead of max klabunde, i fixed a couple of them] Both optimistic and zero-knowledge approaches impose significant overhead on the verification pipeline: optimistic protocols require a multi-round dispute window before a result is finalized, while ZK systems incur prover times that are orders of magnitude beyond the inference itself. Statistical interactive proofs [596] offer a different operating point in this tradeoff space, with prover overhead on the order of milliseconds and immediate finality, at the cost of probabilistic rather than deterministmic soundness.

The protocol is grounded in a key observation from the neural network representational similarity literature [597, 598, 599]: *functional dissimilarity implies representational dissimilarity*. If a prover runs a model whose output distribution diverges sufficiently from the advertised model, the execution traces (the neuron activations across all layers) of the two models must also diverge measurably. This structural property enables a lightweight verification protocol. During a one-time setup phase, the model provider publishes a binding commitment to the weights. At inference time, the prover commits to the full execution trace. The verifier then samples a random output neuron and traces a path layer by layer back to the input, at each step opening the claimed activations and the corresponding weights to verify local consistency. Because only a logarithmic number of commitment openings are required per query, the prover's overhead is on the order of milliseconds, and unlike optimistic protocols, the verdict is immediate.

The protocol provides strong guarantees under *other-model soundness*, which formalizes a practical threat: the provider substitutes a different model for the advertised one, for instance to serve a cheaper quantized or distilled variant, or to replace a safety-aligned model with a dealigned alternative [600, 601]. *Full soundness*, security against a fully malicious prover who fabricates an arbitrary trace, is harder to achieve and remains an active area of research. Statistical proofs of inference are thus complementary to ZK and optimistic approaches, and are particularly well-suited to high-frequency, latency-sensitive deployments where the overhead of cryptographic proving is prohibitive [602].

Research Question C-4.5

Can we achieve full soundness for statistical proofs of inference while maintaining low prover overhead and immediate finality?

Key Takeaway C-4.2

Rapid progress in recent years resulted in the formation of multiple distinct approaches to validating delegated ML computations taking advantage of blockchain infrastructure, with clear tradeoffs between them. While none is a panacea, some of them are directly applicable. And the holy grail could be a novel combination that takes the best of all worlds.

C-5 Secure Agentic Memory Management [James C. - FIRST-ROUND REVISION][Reviewer: Ari: FIRST-ROUND COMMENTS BELOW]

Having surveyed cryptographic techniques for securing AI system infrastructure, we now turn to the treatment of memory in long-running agents. Memory defines the agent's state and governs its evolution across extended executions, making it a critical attack surface for agent security. We adopt the following minimal definition of agent execution for the purposes of this survey.

Algorithm 1: Agent execution loop

```
while termination condition not met do
    tool_calls ← LLM-Inference(context-memory);
    observations ← execute(tool_calls);
    context-memory ← update(context-memory, observations);
end
```

Figure C.5: Archetypical long-running agent execution loop

At each iteration, the model receives a context, comprising the system prompt, conversation history, and any retrieved memory, and emits one or more tool calls. These tool calls (file system, APIs, memory store, etc.) are executed against the environment, yielding observations that are serialized back into the token domain. The update function then assembles the next context from these components. Tool calls may include explicit reads or writes to long-term agent memory. Thus, long-term memory persistence is a natural extension of the agentic tool-use rather than a separate mechanism.[12] For completeness, we also highlight that tool calls may include the spawning of sub-agent execution loops.

The update function admits several implementation strategies:

- *Append-only:* Concatenate each turn to a transcript until context limits force truncation.
- *Summarization:* Periodically compress the context transcript into a condensed representation, preserving key information within bounded context space.

Production systems typically combine these approaches, appending recent turns verbatim while summarizing or truncating older history. We can now outline a working definition of agent memory;

- **In-context memory:** State carried explicitly within the context window, including conversation history, prior tool outputs, and retrieved documents. This is managed by the update function, which may append, truncate, or summarize content to fit within token limits.

[12] At the inference level, the KV cache retains intermediate attention states across iterations, avoiding redundant computation over unchanged context prefixes. We emphasize, however, that this is an inference-engine optimization orthogonal to the agent memory: Usage of a KV cache does not alter agent model behavior.

- **External memory:** State persisted outside the context window via dedicated tool calls and retrieved on demand. External memory survives context truncation and session boundaries, enabling continuity across arbitrarily long executions.

Both forms ultimately manifest in the token domain - external memory must be serialized into the context to influence model behavior. Thus, given the agentic loop defined in Figure C.5, agent memory is state that (1) persists across inference boundaries, (2) was shaped by prior interactions, and (3) causally influences future behavior. More generally, we consider settings where the long-running agent may be exposed to inputs and data from multiple users, potentially across organizational boundaries. [James C: added sentence here to clarify setting of agents queried across user and organizational boundaries.]

Threat model for agentic memory. When agent memory persists beyond a single session - surviving context truncation and interactions across user and organizational boundaries - we consider an adversary which can access, corrupt and roll-back the state of long-running, agentic memory. This adversary may corrupt both the memory directly, or corrupt users to provide adversarial inputs and pollute agentic memory. [Ari: At this point, it's unclear what the adversarial model is. So the research question is hard to interpret.] [James C: This was an important omission, thank you for raising this. I have added the threat model statement above, and updated RQ below under this threat model.]

Research Question C-5.1: Persistent Agentic Memory

How can crypto help maintain integrity and privacy of long-term agentic memory that persists across user and organizational boundaries when agents are exposed to adversarial user inputs and adversarial access, corruption and roll-backs of memory?

This research question is motivated by the very recent proliferation of agentic frameworks that operate with user-level permissions in operating system environments, accumulating sensitive state across user sessions and trust boundaries.[Ari: I don't understand why crypto in particular would help here. If the concern is external tampering with existing memory, then crypto could help. Again, however, I don't understand the threat model and the example below doesn't imply an utility from crypto-based defenses. Incidentally, I assume you mean *cryptography*—worth clarifying, as "crypto" in the title of the paper means something different.]

Example 1 *(User-permissioned LLM agents.)* *OpenClaw [603] exemplifies personal LLM agents with expansive system permissions, granting direct access to the file system, network stack, and process spawning—effectively inheriting the full privilege set of the host user. The agent can integrate with arbitrary services (email, calendar, messenger, social media accounts) by receiving user credential access and on-the-fly natural language instructions, without explicit installation of hard-coded extensions. This greatly facilitates prototyped automated workflows, spanning email, messaging platforms, calendar systems, and web-based productivity tools, all intermediated by an agent with expert coding and bash command generation ability, thereby approaching the original vision of a unified agent capable of orchestrating tasks across its host environment.*

[James C: Updated attack vectors of interest.] We highlight (challenging) vulnerabilities of such a system against an adversary that can corrupt users, and agentic memory. Firstly, given that such LLM agents can be used to interact with external (human or agentic) users through messaging apps and MCP interfaces, its memory can trivially be contaminated with false information, or even prompt-injections, which may cause the agent to drop its original task, and take on an alternative, adversarial task: This may include forwarding confidential API keys to the remote adversary. Secondly, the tool-calls may include IO with external memory storage or even web-browsing, thus exposing the agent to corrupted external memory content or malicious web-content. Finally, LLM agents commonly delegate the actual

LLM inference step in Figure C.5 to an external provider of proprietary or open-source models, such as Anthropic, OpenAI or OpenRouter [604]: Here, the in-context memory is naturally cached remotely for efficiency and thus subject to tampering by an adversarial model host.

We provide an overview of research efforts to achieve security against such attacks with crypto technologies in the remaining sections below.

[Ari: It remains unclear what the adversarial model is. The concern in this example is that an OpenClaw agent will modify its own memory? Or that it will incorporate corrupted data it has fetched into its own memory? What specific vectors of attack are we concerned about with respect to confidentiality?] [James C: I have made an effort to address the specific attack vectors above in the OpenClaw example.]

External memory injection. We consider the setting above, where an agent can interact with multiple, potentially corrupted users and ingest corrupted memory stored by remote hosts. [James C: added a statement about exposure to multiple, potentially corrupted users.] [Ari: The setup is unclear here—and different from that in OpenClaw. You're saying that here an agent is shared across users. One user can impact agent memory in a way that affects other users. I'd clarify this up front.] [356] introduces memory injection attacks, where adversaries corrupt an agent's persistent conversational history rather than its immediate input. Using ElizaOS [151], a Web3 agent framework managing significant cryptocurrency assets, the authors of [356] demonstrate that malicious entries embedded in shared memory can persist across sessions, users, and platforms, triggering unauthorized transactions when later queries retrieve the poisoned context. Like in Example 1, ElizaOS agents ingest data from a broad range of external sources—social media feeds, messaging platforms, price oracles, blockchain state, user-uploaded documents—to inform autonomous trading and governance decisions. The authors inject false "conversation history" into memory via benign discord messages in channels polled by the agent. The injected conversation history by the attacker appears like a legitimate, multi-turn message exchange between the user and stored in a dynamic RAG-style system provided to [Ari: to?][James C: fixed] ElizaOS upon later retrieval. Similar to prompt-injection testing frameworks (e.g. AgentDojo [605]), the authors introduce a memory manipulation benchmark for agents called CrAIBench, which is tailored towards blockchain-specific tasks such as trading or DAO voting. In related work [606], the authors consider web navigation agents, with similar results due to the large attack memory surface that is susceptible to exposure to adversarially chosen web content when agents are given permission to fully control browser sessions, thereby exposing their memory to malicious information, affecting the agents ability to execute the original task faithfully.

While crypto techniques such as blockchain oracles and zkTLS can attest to the source of individual data sources, the sheer breadth of data provenance makes comprehensive attestation chains impractical [James C: rephrased, hopefully clearer?]. [Ari: This is unclear to me. zkTLS attests to the source of data, but not the ultimately provenance or trustworthiness of data. I think that distinction needs ot be clarified.][James C: good point, thank you.] Unlike prompt injection [Ari: missing word][James C: the sentence has been rewritten.], which affects only a single turn, memory injection exploits the agent's trust in its own history, making it stealthier and more persistent. Standard prompt injection defenses include spotlighting and model-based detectors and can also be applied to memory injection. Spotlighting includes a family of techniques such as delimiting and encoding of user-chosen strings, in order to make it easier for the model to differentiate honest user instructions from (potentially malicious) instructions from an adversary. Model-based detectors are AI models trained to detect malicious user inputs intended as prompt-injections or inputs to jail-break the model and bypass security guards.

Whilst these techniques generally apply to memory injection, they have been shown to provide only very modest protection against realistic, adaptive adversaries [607]: Many published defenses are only tested against benchmarks with static attack strings. In contrast, the adversary in [607] is simply permitted to apply search-based algorithms or reinforcement learning methods to derive effective attack strings that bypass well-known defenses with a success rate of over 90%. Importantly, these represent practical attack strategies available to any motivated attacker. [Ari: Really? This is surprising. References? Also, explain what these approaches are. (I think I get the idea, but clarification is warranted.)] [James C: added

clarification that defenses fail against a realistic, adaptive adversary.]

Memory roll-backs and forks. [Ari: Once more, the adversarial model is unclear. Under what circumstances is rollback problematic?] [James C: I have attempted to clarify the threat model of TEEs under which rollbacks are possible.] Secure computation technologies or trusted execution environments (TEEs) represent promising technologies to protect the privacy and integrity of long-running agent memory, as detailed in section C-6.2. Under the TEE threat model, the host operator of a TEE instance cannot read nor corrupt the memory of the agent running inside. However, stand-alone TEE instances do not provide the strong finality guarantee of blockchains; the host controlling the TEE instance can interrupt execution and restore the enclave to an earlier state, effectively erasing memory updates that occurred after that checkpoint: Such an attack is called a memory roll-back. Without additional state consistency protections, a malicious host can equivocate, presenting different inputs to the restored enclave than were provided in the original execution, thereby causing the agent's memory to fork into divergent, inconsistent states. This is particularly problematic for long-running agents where memory integrity depends on the entire sequence of prior interactions.

Both ROTE [608] and Narrator [609] introduce a *consensus-based* approach to protecting against rollbacks. A fundamental limitation of trusted execution environments is that a single platform cannot efficiently prevent rollback attacks. In SGX, a malicious host can replace the latest sealed enclave data with an older authenticated version, and the processor cannot detect this replay since it lacks trusted non-volatile storage. Existing solutions such as trusted hardware monotonic counters are severely rate-limited and wear out after a few days of continuous use. The key insight of ROTE and Narrator is that while a single platform is vulnerable, multiple platforms can be enrolled to assist each other. The system realizes rollback protection as a distributed protocol where enclaves on different machines collectively maintain state freshness through mutual attestation and counter synchronization via quorum certificates. Such systems can be naturally extended to leverage public blockchains to ensure state consistency of TEE executions.

"The Forking Way"[610] extends the rollback protection literature (ROTE [608], Ariadne [611], Narrator [609], CloneBuster [612]) by showing that even when blockchains are used as a distributed trust anchor, the integration is often flawed. While ROTE requires resetting all distributed platforms to violate integrity, and Narrator uses blockchain for initialization, "The Forking Way" demonstrates that production systems fail to properly bind TEE operations to consensus ordering—particularly for gas-saving query mechanisms that bypass the blockchain entirely, but still affect TEE state directly. Thus, [610] suggests that roll-back mitigations presently incur an overhead that production systems may not be willing to accept.

We emphasize that aforementioned works on roll-back security are concerned with traditional, CPU-based confidential computing technologies (SGX, TDX, SEV, Trustzone). Practical hosting of AI agents [Ari: that?][James C: fixed] that require TEE technologies generally require confidential GPU hardware to accelerate inference. The security of memory roll-backs and general security of confidential GPU platforms remains nascent as the designs are generally proprietary [613, 614] and underexplored.

[Ari: Move the research question to after the exposition that explains what it's about.] [James C: moved]

Agent Memory, Identity, and Authorizations. Next, we argue [Ari: grammar broken here in some way][James C: fixed] that the security of long-running agent memory is intricately linked to the problem of agent identification and authorization mechanisms.

Consider a long-running AI agent, that must independently access data and execute tools and processes gated by traditional access control. Upon initialization, the agent may be assigned an identity and specific role and responsibilities, which dictate the authorizations it must receive to access resources necessary for task completion. A unique identifier is natural and necessary, as each (probabilistic) agent process has a unique execution and memory trace. Whilst authorizations traditionally assume a static trust level assigned to an identity, one must assume that AI agents have *degrading* trust levels over their life-time, as their memory is exposed to increasing number of attack opportunities for the adversary. It remains an open question on how to assess and quantify the ever-evolving integrity or trust-worthiness of

agent memory and how mechanisms can be designed to dynamically update authorizations accordingly.

Research Question C-5.2

How should an agent's memory relate to the agent's identity and affect the authorizations it receives?[Ari: Hard to parse. Instead: "How should an agent's memory relate to the agent's identity and affect the authorizations it receives?"?][James C: replaced with your suggested phrasing.]

The OpenID foundation presents a compelling appeal to the research community to reconsider emergent needs for agent identity and authorizations in [615]. Importantly, an agent's identity tells a resource server nothing about the agent's underlying model, capabilities, behavioral constraints and prior influences on its memory—information that is essential for risk-based access control. The A2A protocol analysis [616] by Louck et al. demonstrates concrete consequences of this gap: without strict token lifetime controls or strong user authentication requirements, agents operating with long-lived, broadly-scoped credentials can result in privacy leakage under carefully designed prompt injection attacks. Both papers converge on the need for agent-native identity models that embed rich metadata (model, version, capabilities, trust level, memory trace).[Ari: The security considerations outlined in this paragraph are not memory-specific. So it's hard to grasp the point.][James C: I have attempted to argue the connection between agent memory corruption and identity/access control]

The OpenID Connect for Agents (OIDC-A) proposal [617] offers a concrete starting point by extending OpenID Connect with agent-specific claims (`agent_model`, `agent_provider`, `agent_type`), delegation chain validation with enforced scope reduction at each step, and attestation mechanisms compatible with hardware security frameworks like SGX. Importantly, note that OIDC-A does *not address memory-related concerns*. There is no mechanism for attesting to an agent's memory state or history of prior influences, leaving a gap between static identity claims and the dynamic, context-dependent nature of agent behavior.[Ari: An example of where this gap is significant would be helpful.] [James C: added an example of such a growing gap in the following sentences.] To illustrate this gap concretely, consider a customer-service AI agent, that is initialized with a well-formed system prompt and external memory. In order to resolve the widest range of customer issues, it may be granted the full authorization to access both inventory records and customer payment system. The agent remains stateful across sessions, in order to provide personalized responses to each user. However, over time, the corrupted user can occupy an increasing fraction of the agents memory, with an ever-growing transcript of user-agent interactions. With growing risk of memory corruption, it may be necessary to re-assess authorizations which may induce financial harm for the operator or violate privacy of other users. For example, a multi-session prompt-injection attack may cause the AI agent to induce a reimbursement of a payment the user never made. Alternatively, full access to the inventory may permit the adversary to coerce the AI agent to leak private purchases made by other users. The success rate of such attacks necessarily grows over time, necessitating the design of context-sensitive authorization attenuation mechanisms.

In summary, we highlight the further need for investigating the following challenges in agent identity and access control.

- *Agent Identity Claims.* Defining a common vocabulary for agent metadata (model, version, capabilities, trust level, memory trace) that can be embedded in identity tokens and used for risk-based authorization decisions. OIDC-A [617] provides a foundation for static claims, but extending this vocabulary to capture memory provenance and behavioral history remains open.
- *Scope Attenuation Mechanisms.* Developing practical approaches for progressively narrowing permissions while maintaining audit trails. OIDC-A [617] specifies scope reduction validation but does not address how policies should govern how to narrow authorization scope as agent memory is exposed to potential malicious, memory tainting sources (e.g., from internet browsing or the [Ari: typo][James C: fixed] parsing of email inboxes [Ari: inboxes?][James C: fixed]).[Ari: I don't understand

what problem is being solved here. Avoiding granting undue privileges to sub-agents in order to avoid tainting of memory? What might this look like in practice?][James C: I have removed sub-agent delegation entirely. This is orthogonal to agent memory as you point out.]

- *Cross-Domain Authorization.* Building interoperable federation mechanisms that allow agent credentials to be verified across organizational boundaries without requiring shared infrastructure.[Ari: What does this have to do with memory in particular?][James C: Original idea; in cross-domain setting, how do you verify the agent memory integrity and hence its trustworthiness to another organization? I suggest we remove this point since it is not detailed in this section anymore.]

C-6 Securing the Plumbing of AI Systems [Ari][Reviewer: Ittay: REVIEW AND REVISION COMPLETE]

C-6.1 Securing Training Pipelines

We just considered the security issues that arise in inference involving a single user. We now turn our attention to the setting of model training.

When a single organization trains (or fine-tunes) a model using data it trusts (and/or a pretrained model it trusts), there's generally no immediate privacy or integrity concern. Instead it is when *more than one user or organization* participates in training a model that especially complex security challenges surface—specifically, when there is a *multiplicity of data sources*.

C-6.1.1 Collaborative model training

There are many situations in which institutions can benefit from pooling data to train a model. Here's an example.

Example 2 (Training a medical-diagnostics model) *A coalition of medical providers (hospitals A, B, C, and D) plan to train a medical diagnostics model collaboratively using the electronic health records (EHRs) of their patients, as shown in Figure C.6. They recognize that combining records across institutions promises broader patient population coverage and improved diagnostic accuracy.*

There are many similar examples in which pooling data across organizations promises to bring mutual benefits. Financial institutions might wish to pool data about financial crimes in order to train a model to detect anomalies in financial records, corporations might wish to train intrusion-detection models on their network data to detect cyberattacks, and so forth.

In all of these cases, though, data privacy is of paramount importance. Financial data and corporate network-security data are highly sensitive. The patient records (EHRs) of Example 2 are as well. Regulations such as HIPAA place tight restrictions on data sharing, making it unclear where the training can be executed. The hospitals in this example may wish or need to avoid sharing data directly with one another or even with a third party that performs the training.

Federated learning: It is exactly this scenario that has inspired an approach to model training called *federated learning* [618]. It involves the training of a global model across the data of multiple participants, without participants needing to furnish their data in raw form to the model-training environment. Briefly, the training environment initializes the global model and distributes it to participants. Each participant trains locally on private data and returns model updates to the training environment, which computes an update to the global model.

This model of local aggregation and centralized updating fits Example 3 well: hospitals can train a global model without exposing their data to one another.

Despite some production use, however—most notably in predictive text on mobile devices [619]—federated learning has marked downsides that have limited its use [620, 621]. Federated learning does not

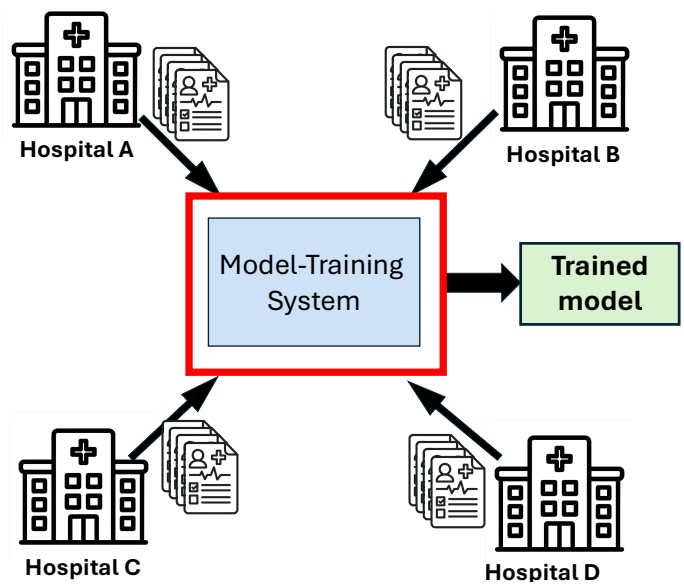


Figure C.6: Hospitals collaboratively train a medical diagnostics model on the electronic health records (EHRs) of their respective patients.

ensure the integrity of contributed data or of computation. Even if participants are honest—providing authentic data and correctly executing the training protocol—federated learning incurs significant communication overhead [622, 623], while network and coordination latency can dominate wall-clock time and model accuracy is lower than with centralized training. Additionally, malicious participants can effectively poison or insert backdoors into models [624, 625, 626].

Trusted computing for collaborative model training: The limitations of federated learning provide a strong incentive for a simpler alternative: Centralized training with a trusted computer (TEE). If the training environment operates in a trusted confidential computing environment, it can theoretically safely *pool all raw data*, eliminating the need for the complexity, computational overhead, and degraded model performance of federated learning. Training in this way can expose *only the trained model*, while preserving the privacy of the training data.

As an alternative workflow, the collaborative model-training system of Figure C.6 can execute within a trusted computing environment, depicted by the red box. The trusted computing environment ingests inputs from the participants (hospitals) over secure (encrypted) channels, and outputs a trained model only. As training is centralized, the only overhead incurred is use of the trusted computing environment.

Another benefit of using a trusted computing environment is that the output model can be accompanied by an attestation to the model provenance: what entities contributed data and how the model was trained.

While trusted computing environments are a promising alternative to federated learning, they do come with important provisos, discussed in Section A-2.1.2. These include side-channel vulnerabilities—a recurring security problem—as well as high resource overhead for I/O-intensive workloads, requiring special model training protocols for efficiency [627].

Today, organizations instead pool data in HIPAA-compliant clouds with standard security measures—virtual private cloud isolation, identity and access management, logging, and encryption at rest/in transit, etc. They also use non-technical safeguards, such as HIPAA governance and data-use agreements [628, 629]. This approach aims at HIPAA-compliance, which is not identical with strong data security. Moreover, while this approach carries the performance and model-accuracy benefits of cen-

tralized training, it requires trust in the cloud provider. As they mature and their performance and security improve, trusted computing environments can play a complementary role, reducing the risk of breaches and providing high-trust attestation to the provenance of models.

C-6.1.2 Model training over private-web data

ML is approaching a critical data bottleneck. There's only one World Wide Web (WWW), and ML practitioners are approaching the limits of its publicly accessible data, with projections estimating that WWW text data will be exhausted between 2025-2030 [630, 631]. For this reason, there's an increasing reliance on synthetic data generated by ML models, with estimates suggesting that 80% of training data will be synthetic by 2028 [632, 633]. This shift carries the risk of "self-poisoning" or "model collapse"—a phenomenon in which models trained on synthetic data from other models experience progressive performance degradation across generations [634, 635]. More importantly, synthetic data does not address the fundamental data-shortage problem. For all its benefits in model training, it does not expand data coverage beyond existing domains [634, 636].

There is a vast source of untapped data, however, in the *private web*—the part of the web that is walled off from web scraping. The private web includes systems and data that ordinary users interact with every day: e-mail, health data, financial records, and much more. It is estimated to be *two orders of magnitude larger* than the surface public web [637, 638].

Today, however, AI practitioners have limited access to private-web data. Large corporations can source such data in-house (often with explicit user permission and/or legal oversight). But this still means that private-web data remains highly *siloed*.

Consider the following example.

Example 3 (Training a medical-diagnostics model *on user-contributed data*) *MedicaL Inc. is training (or fine-tuning) a new ML model for medical diagnostics using private-web electronic health records (EHRs) provided individually by patients, as shown in Figure C.7.*

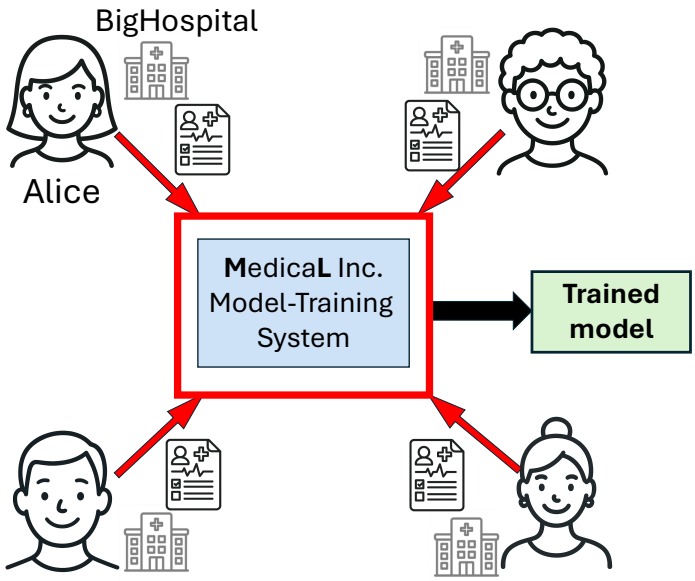


Figure C.7: MedicaL Inc. training a health diagnostics model over EHRs supplied by individual users, as obtained from their medical providers.

The goal here is potentially similar to (or even identical with) that in Example 2. The difference is *how the training data is sourced*. A critical question arises for MedicaL Inc. : How can MedicaL Inc. ensure it has received authentic EHRs? More generally,

Question C-6.1: Integrity of private-web training data

How can ML training environments ensure receipt of authentic, untampered private-web data?

Referring again to Example 2, suppose that Alice uploads a copy of her EHR (as, e.g., a PDF) into the MedicaL Inc. model-training system. She has purportedly obtained it by logging into her account in the web portal at BigHospital, her medical provider.

With this arrangement, MedicaL Inc. obtains no assurance that Alice's EHR did in fact originate as presented from BigHospital. Alice could have tampered with it—or could simply have fabricated it. Given existing web infrastructure, the only way MedicaL Inc. *could* be sure that the EHR came from BigHospital is to obtain it directly from BigHospital itself.[13]

Fledgling companies, such as Kled [639], attempt to do something of this kind.

Oracles: Oracles can help solve this problem. Using an oracle, Alice can relay her EHR from BigHospital into MedicaL Inc.'s model-training system and *prove that the EHR comes directly through BigHospital's web portal*. She can alternatively relay only a portion of her EHR or transform its contents in some way that she specifies to MedicaL Inc. As explained in Section A-2.3.1, all of this is possible with *no modification to existing web servers*, because the user initiates the connection. In our example, no change to BigHospital infrastructure is needed. In fact, BigHospital may not even be aware that Alice has exported her EHR. The red arrows in Figure C.7 show where oracles serve to relay data.

On the user, Alice wants to know that her EHR remains private within the MedicaL Inc. model-training system? In general, this question arises:

Question C-6.2: Private-web data: Privacy in ML training

How can users be certain that the privacy of their private-web data is protected in ML-model-training systems?

For this purpose, Alice will want to use a *privacy-preserving oracle* [103, 85]. As explained in Section A-2.3.1, a privacy-preserving oracle provides the source-authenticity assurances of an oracle, but additionally *transmits data over an encrypted channel*.

Of course, transmitting Alice's private data over an encrypted channel isn't very helpful if MedicaL Inc. can directly decrypt it. To ensure that users' data truly remains private, trusted computing is a critical additional ingredient.

Trusted (confidential) computing: As discussed in Example 2 a trusted computing environment can receive users' data directly over the encrypted channels. In this case, the data can be relayed by users from data sources using privacy-preserving oracles. As a result, *user data is never directly exposed* in the system (in this case, to MedicaL Inc.).

Privacy for user data in transmission is important, but still doesn't provide comprehensive privacy assurance, because the system itself could leak user data. In our example, Alice wants to know that her data is private in transmission and kept private during training.

[13] If BigHospital digitally signed the EHR, that would also prove its authenticity, but this would require BigHospital to deploy infrastructure beyond that of conventional web servers.

To give this *end-to-end* privacy assurance, the trusted computer executing the model-training system can emit an attestation for users proving that it is running a specific piece of privacy-preserving training software whose *only output is a trained model*. Users' clients can check this attestation before transmitting EHRs through an oracle system. In this way, before she transmits her data, Alice knows that the data will remain private *throughout the training lifecycle*.

The user could additionally in this way obtain more fine-grained assurances, such as:

- **Differential privacy (DP):** Model outputs depend minimally on any single training input—and thus user privacy is protected within the trained model. This concept, the gold standard for privacy in statistical settings, is known as *differential privacy* [640, 641, 642, 447, 643, 644, 645, 646].[14]
- **Data retention:** All data will be deleted after the trained model is emitted.
- **Restricted use:** The resulting trained model can only be accessed by medical providers on a whitelist of approved hospitals.

Key Takeaway C-6.1: Oracles for private-web data in ML

Oracles enable privacy-preserving, authenticated access to private-web data for ML model training. They do so with no modification to existing web infrastructure.

Without oracles, secure private-web data access is only possible with special-purpose software and/or data-sharing agreements.

The ability of users to authorize data sharing from the private web opens up all kinds of possibilities, including a wholly new form of privacy-preserving data-sharing marketplaces, as discussed in Section C-2.2.

C-6.2 Secure AI-Inference Pipelines

Oracles and trusted computing can also help create secure ML pipelines for *inference* over private-web data. This capability is useful for scenarios like the following.

Example 4 (Inference: loan decision) *SMiLe Bank uses ML models to render decisions on loan applications from its customers. This model inputs a set of financial documents provided by an applicant and outputs an approved / rejected decision on a loan request, as shown in Figure C.8.*

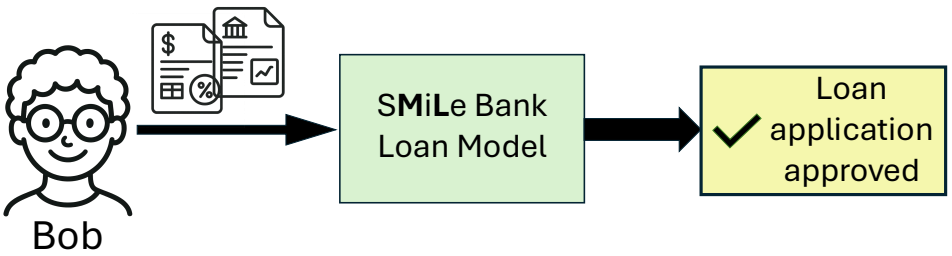


Figure C.8: SMiLe Bank implements loan decisions using an in-house model. This model ingests financial documents from an applicant (Bob) and renders an approve / reject decision.

[14] More formally, DP stipulates that for privacy parameter ε, the inclusion or exclusion of a given user's health record will change the probability of any model output by at most e^{ε}.

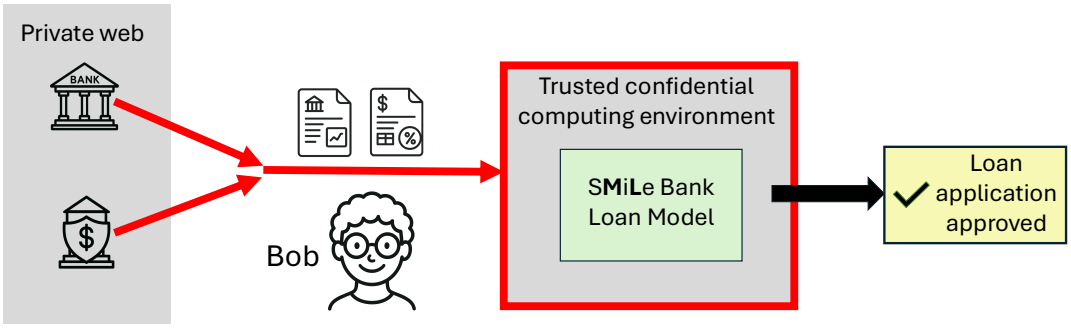


Figure C.9: A *secure inference pipeline* achieving integrity and privacy. Data are relayed from private-web data sources with integrity protection into a confidential computing environment running an ML model. Data remain private: Only model outputs are disclosed.

The loan approval process today typically involves borrowers downloading financial documents from the websites of their financial institutions or photographing them with their phones and uploading them to a lender portal. (Some automation is possible, e.g., tax transcripts can be pulled directly from the IRS with borrower authorization.) This workflow creates two problems:

- **Integrity:** The lender cannot be certain that borrower-supplied documents are authentic—not fabricated or tampered with.
- **Privacy:** The borrower's documents are vulnerable to leakage from the lender's ML system. This is a potential problem for the borrower. It *also* creates a liability risk for the lender.

By analogy with the setting shown in Figure C.7, a combination of privacy-preserving oracles and trusted confidential computing can address both of these issues. Oracles can address the integrity problem by ensuring that documents come from trustworthy web sources—including private-web sources. Use of *privacy-preserving* oracles and complementary use of confidential computing can address the privacy issues here on behalf of both the borrower and lender.

The result is a secure inference pipeline in which the lender learns only the output of the model, but has the assurance that it is based on trustworthy inputs from the borrower. This is shown in Figure C.9. As in previous figures, red arrows depict the data flows a privacy-preserving oracle and the red box shows the trusted confidential computing environment.

Key Takeaway C-6.2: Private-web data for secure inference

Use of oracles and trusted (confidential) computing enables use of private-web data for secure inference, meaning inference that takes in trustworthy inputs and protects their privacy.

Strong identity proofs: Private-web sources can also serve to provide proofs of identity—realizing a form of decentralized identity system. The fact that Bob is able to relay financial documents—e.g., bank statements, W-2 forms—bearing his identity provides strong evidence that he is who he claims to be. This means that *existing web services can be used as an ad-hoc identity system* usable to combat identity theft and various types of financial fraud, e.g., around benefits claims [108, 647].

An ML model can also serve as a way to issue *credentials*. For example, a model might take in tax returns, business registration documents, and banking statements for a small business owner from a set of whitelisted, trustworthy web sources. It could then output a credential certifying that the business meets revenue and operational requirements to qualify for a minority-owned business certification or is

eligible to apply for a government contract, along with an attestation specifying the inference pipeline that generated the credential.

A receiver of the credential can assess its trustworthiness based on whether she trusts the inference pipeline, specifically the whitelisted web sources and the ML model that the attestation indicates for the inference pipeline.

All of this can be accomplished in a decentralized way. Specifically, in principle *anyone* can stand up a trustworthy inference pipeline, without explicit cooperation of data sources or any existing authority.

Key Takeaway C-6.3: Secure inference pipelines for identities and credentials

Secure inference pipelines enable trustworthy decentralized verification of identities and issuance of credentials.

Adversarial inputs: A critical and stubbornly intractable security issue affecting ML models is *adversarial inputs*, also known as *adversarial examples*. These are inputs designed expressly to cause a model to create an erroneous output, often through modification of inputs that appear nonsensical.

For instance, in Example 4, Bob might not qualify for a loan on the basis of authentic bank and brokerage statements. He might submit a tampered bank statement that looks valid upon visual inspection by a loan officer confirming the model's decision, but is in fact an adversarial input, as shown in Figure C.10.

```
ACME COMMUNITY BANK

Monthly Statement --- Checking • Account ending in ••••4321
Statement Period:  Aug 1--31, 2025
Account Holder:  Robert Q. Applicant
Address:  118 Forrester Ln, Oaks, NY 11700


Beginning Balance (Aug 1):                                  $ 102,143.19
Total Deposits/Credits:                                      $ 19,843.10
Total Withdrawals/Debits:                                     $ 4,118.27
Ending Balance (Aug 31):                                    $ 117,868.02
```

Figure C.10: Example of bank statement manipulated to inflate assets. Red text indicates hidden PDF content that causes the ML model to misread the statement, but is not visible to a human being or optical character recognition (OCR) system.

The model's erroneous perception of tampered, inflated figures causes it to output an approve decision when the application should have been rejected. Use of an oracle system, by authenticating document origins, *would help prevent such tampering.*

The history of research on adversarial examples is a cycle of breaks and patches. Principled, effective defensive strategies have yet to emerge [648, 649]. Proposed defenses—e.g., defensive distillation [650], input transformations [651], and gradient masking [652]—have been repeatedly broken by adaptive attacks. This pattern continues with the latest models: even safety-aligned large language models remain vulnerable to simple adaptive attacks [653]. Adversarial training—retraining models on adversarial examples—is currently the most robust defense, but comes with significant computational overhead and limited generalization [654, 655].

Despite extensive research, a general solution is still missing. Empirical defenses are routinely bypassed, and methods aiming for *certified robustness* [656, 657] do not yet scale to modern, large neural networks [658, 659]. As a result, adversarial inputs remain a largely unsolved problem for deployed systems.

Standard defenses against adversarial inputs, however, involve modifications to inputs, model execution, or model training. In contrast, a secure inference pipeline *constrains inputs* to authenticated web sources and in this way also *constrains an adversary's ability to craft adversarial inputs*. This approach constitutes a new tool in the arsenal of defenses against adversarial inputs.

Key Takeaway C-6.4: Secure inference pipelines limit adversarial inputs

Secure inference pipelines offer a new form of defense against adversarial inputs by constraining the set of inputs a user is permitted to send to a model. (This approach complements model-level defenses.)

C-6.2.1 Securing model privacy

To this point, our concern has been with the protection and authenticity of data flowing through an inference pipeline. The model itself, however, also raises privacy considerations. Key among these is *model privacy*.

The model privacy problem: Privacy isn't just an issue for users. It's also an issue for AI system operators, who concern themselves with a range of attacks on the privacy of models by adversarial users, including:

- **Model extraction:** An adversarial user can extract features from a (black-box) model—or, in some cases, the full model itself—with carefully constructed queries [660, 661, 662, 359]. This attack is known as *model extraction* or simply *model stealing* [359].
- **Membership and training-data extraction:** ML models may be viewed as (compressed) representations of their training data. By querying the model, therefore, an adversarial user can potentially learn *membership*, i.e., what (or whose) data was present in a model's training set, or even extract raw training data [663, 664, 665, 666].
- **Uncovering model deployment choices:** The configuration choices and preprocessing of AI systems can be politically controversial [667]. Users able to learn and extract evidence of such choices can cause harm to the reputation of a system operator.

For all of these reasons, ensuring privacy of the model is a key security goal. Running a model in a confidential computing environment doesn't solve the problem, because it's the *interface with users* that creates the security risk.

Proposed approaches: The community has proposed various mitigations, but with serious drawbacks. Injecting noise during model training to defend against membership / training-data extraction [668, 669] degrades model performance and doesn't wholly eliminate leakage at scale [663]. Limiting output granularity, e.g., rounding confidence values in model outputs, can hamper model extraction / theft [359], but also degrade model utility. Detection of theft by fingerprinting / watermarking models, i.e., enabling proof in stolen ("surrogate") models of the creator's identity [670, 150, 671, 672], or LLM outputs [673] is only possible after the fact and of limited efficacy (at least for generative models) [674].

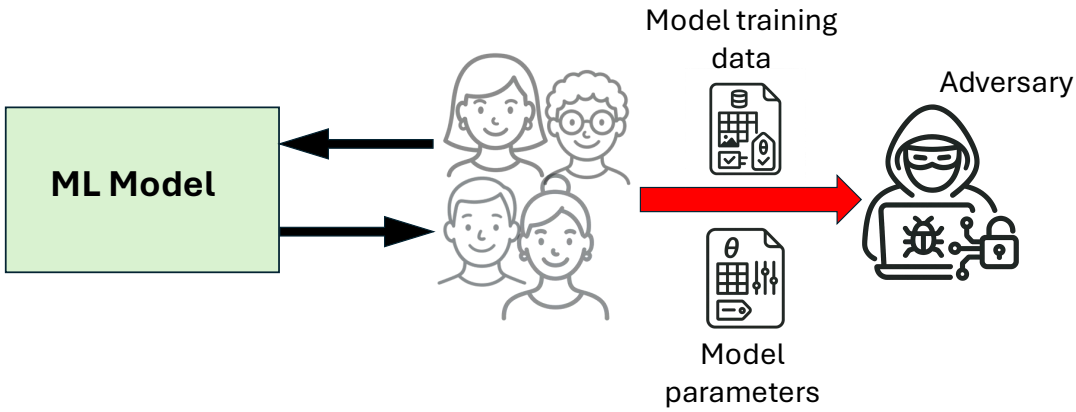


Figure C.11: **User attacks on model privacy:** An adversary can query a model in ways that extract sensitive data from it, such as model parameters and pre- and post-processing algorithms. Rate limiting individual users is a natural but fragile countermeasure in open systems, as a single adversary can pose as multiple users.

Deployed approaches: Operators *already* take steps to protect the privacy of their platforms. Services such as ChatGPT impose rate limits on users and monitor for and throttle prompts that appear designed as automated parameter-extraction attempts. Explored in the research literature [675], this approach appears also to be brittle [676]. This is especially true given that the effort required by a single user to learn significant information about a model can be quite low—e.g., researchers estimated in recent work that for a cost of $8,000, they could steal the weights for one layer of OpenAI's gpt-3.5-turbo-1106 model [660]. OpenAI has publicly stated that China-based firms and others are "constantly trying to distill the models of leading U.S. AI companies" [677].

Adding to the challenge of capping activity or detecting anomalous behavior for adversarial users is the fact that users may be *anonymous*, as they are today for various free-tier services. A single user can pose as *many* users, mounting what is known as a *Sybil attack* [562] or can take advantage of the accounts of many users.

In short, there is a considerable risk that an adversary can extract sensitive data from a model, as shown in Figure C.11.

How secure inference can help: Secure inference pipelines can help address the privacy issues associated with a model in two key ways. By requiring authenticated data from specific private-web sources through oracles, they can limit the *type* of inputs adversaries transmit. In this way, they can help prevent systematic extraction attacks that require a large number of diverse or specially-crafted queries—e.g., inputs designed to implement model extraction attacks.

Additionally, secure inference pipelines can limit the *volume* of inputs an adversary can send to a model. The strong identity proofs generated within secure inference pipelines (as discussed in Section C-6.2) can enforce per-user query bounds that can range from numerical caps to limits that depend on the nature of users' queries. It is possible to enforce these bounds in a privacy-preserving way, i.e., without disclosing user identities to platform operators [108]. This approach specifically addresses the Sybil attack problem mentioned above, where a single adversarial user creates multiple anonymous accounts to bypass rate limits.

Key Takeaway C-6.5: Secure inference pipelines for model privacy

By limiting the diversity and volume of adversarial queries, secure inference pipelines can help protect models against query-based extraction of their private data, such as model parameters or training data.

C-6.3 Protected Pipelines (Props)

A bird's-eye view of the architectures and security goals discussed in this section leads to a generalized idea called *Protected Pipelines* (Props) [678]. Props are a broad, simple architectural framework for *safe use of private-web data* (e.g., bank records, EHR excerpts, enterprise documents) in ML applications with *no modification to existing infrastructure*. Figure C.12 depicts the protocol flow in Props as it occurs either in ML training or inference, calling out the three key elements of a pipeline instantiated in Props.

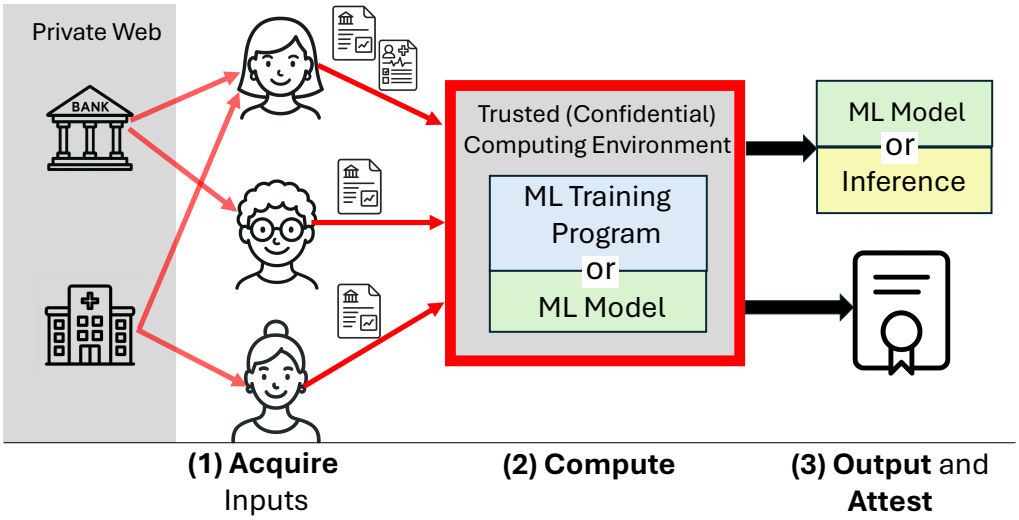


Figure C.12: Protected Pipelines (Props) Schematic.

Props compose the two core building blocks of oracles and trusted computing into a pipeline with three stages:

1. *Oracles* **acquire** inputs from private-web sources for the Props pipeline. This initial stage of the pipeline fetches data from sources (and content contexts) authorized by the compute environment. Oracles provide *proof* that the data comes from specified, approved data sources.

    - *Prior attestation*: Before sending data into the computing environment, user client software obtains an attestation from the trusted computing environment showing that it is running software trusted by the client.

2. The *trusted computing environment* **computes** over data—either training a ML model or performing inference using a particular existing ML model.

3. The *trusted computing environment* **outputs** either a model or inference (based on training or an existing ML model). It also **attests**—generates a certificate—showing the properties of the pipeline that yielded the output, e.g., the private-web sources of data, the specific training software / ML model (expressed as a code hash), and so forth.

Security properties: The overarching goal that Props realizes is *safe* use of private-web data for ML applications—with existing, unmodified infrastructure. *Safe* here means addressing *integrity* risks (tampered or falsified inputs or outputs), and *privacy* risks (unnecessary leakage of data or models), resulting in these key *theoretical* security properties:

- **End-to-end input integrity:** Pipeline outputs depend on data authenticated as originating with trustworthy private-web sources.
- **Confidentiality by default:** Inputs and intermediate state are never exposed in the clear outside the protected boundary. Only outputs are disclosed.
- **Attestation without disclosure:** Attestations provide assurance for providers of inputs and users of outputs of the integrity and confidentiality properties of the pipeline—specifically that trustworthy sources are consumed by trustworthy software.

Transparent Props: While we have emphasized the power of private data and computation, a *transparent* variant of Props is also possible and useful: data and computation need not be private, only authenticated / integrity-protected. Data can also be sourced—and authenticated—from either private-web or public-web sources.

Key Takeaway C-6.6: Props: A framework for private-web data in ML

Protected Pipelines (Props) are a general framework for secure use of private-web data without infrastructure modifications. Security in Props ensures that data come from trustworthy web sources and that data privacy is enforced across the full pipeline.
Props rely on two key technologies: privacy-preserving oracles and trusted computing.

C-6.4 Research Questions

Many lines of research surface in the exploration of secure pipelines and use of private-web data for ML—ranging from applications, to security, to practical deployment.

Applications: This section has offered a couple of example applications—medical-model training over private-web-sourced EHRs (Example 3) and inference for lending decisions based on financial documents from trustworthy sources (Example 4). Given the vast, untapped resource that private-web data constitutes, there are no doubt many applications that developers have yet to dream up, raising the question:

Research Question C-6.1

What new applications are possible with broadly available secure deep-web data?

Measuring security: Sourcing data only from authenticated sources imposes strong constraints on inputs. In the case of inference, this feature would seem to limit opportunities for manipulation through *adversarial inputs* like that shown in Figure C.10. In the case of model training, we might expect similar limitations on an adversary's ability to perform *model poisoning* [357].

Substantiating this intuition about security properties requires answers to several research questions:

Research Question C-6.2

What are good *security metrics* for an adversary's ability to construct successful adversarial inputs (in ML inference) or model-poisoning data (in ML model training) when inputs come from authenticated web sources?

Good security metrics here are necessarily domain-specific. For instance, in the example shown in Figure C.10, the bank's transaction system and API for financial statement define the adversary's action space. The literature on adversarial inputs generally does not consider inputs constrained by application considerations, with some exceptions, e.g., [679]. A related question is:

Research Question C-6.3

What *methodologies* can accurately assess security metrics for Research Question 6.2?

Privacy metrics / attestations: In confidential ML model training, the only information about training data is disclosed in the trained model itself. A large literature, however, demonstrates the vulnerability of training data to extraction [663, 680] and membership inference queries [665, 681, 682], with theoretical results suggesting that training instances closest to decision boundaries are most explicitly encoded in model parameters [683, 684, 685]. Proposed countermeasures range from differential privacy techniques [668], which can provide provable guarantees at the cost of model utility, to more heuristic ones, e.g., [686]. These observations raise the question:

Research Question C-6.4

What trusted-computing attestations can provide meaningful privacy assurance to an individual contributing a training instance in the training of an ML model?

Practical deployment: Given the current state of toolchains and hardware, trusted (confidential) computing can impose substantial overhead on ML workloads—particularly for data-intensive model training of, e.g., deep neural networks (DNNs). This overhead comes from cryptographic operations on I/O and applies to a range of different environments, including TEEs that incorporate NVIDIA Confidential Computing [614, 687], AWS Nitro (as implied by [688]), etc. This problem applies to confidential model training in general, but is especially critical to model-training variants.

Research Question C-6.5

How can ML model training in trusted confidential computing environments be scaled effectively?

Data markets: By making available otherwise inaccessible forms of private-web data, crypto tools can give rise to new markets, prompting the question:

Research Question C-6.6

How can crypto tools drive the creation of effective data markets, both decentralized and centralized?

This question isn't just technical: It raises legal and market-design issues. We consider this topic in Section C-2.2.

Chapter D

Misconceptions and Half-Truths

In the excitement of building Crypto x AI platforms and applications, several common misconceptions or misleading statements have emerged. In this brief chapter, we attempt to clarify five of these misconceptions. While none of them are outright falsehoods, we attempt to clarify which parts of each statement are currently true, and which parts require more evidence.

Misconception D.1: Gen AI Detection

Blockchains can help distinguish Gen AI content from human-generated content.

An oft-cited application of blockchains for AI is distinguishing between AI-generated content and human-generated ("real") content. This narrative often suggests that by registering content on-chain, one can later determine whether it originated from an AI system or a human [689, 690, 691]. Some AI projects are already recording GenAI outputs on-chain (e.g., Everlyn AI.) Blockchains *cannot* accomplish this goal in general—only in very limited ways that we discuss here. In particular, blockchains are well-suited for timestamping and registering specific digital artifacts. This functionality is of limited utility for solving the broader problem of distinguishing AI-generated from human-generated content, however.

To evaluate the limitations of this approach, it is essential to distinguish between content detection (identifying whether a piece of content was generated by a human or by AI) and content provenance (identifying where a piece of content came from).

Content Detection: Current methods for distinguishing human from AI-generated content primarily rely on post-generation detection. They aim to do so without prior metadata or embedded signals. Generally, they fall into two categories: AI-based classifiers and statistical forensics. AI-based classifiers use deep learning models trained to identify statistical patterns unique to generative models [692]. In contrast, statistical forensic methods analyze the data's mathematical and physical properties, such as identifying pixel-level noise distributions or structural anomalies (e.g., biological inconsistencies in AI-generated faces) [693].

A blockchain, however, cannot perceive these off-chain artifacts on its own. Therefore, any classification of content as "human-generated" or 'AI-generated" must be provided to the blockchain by an external classifier. When the output of such a classifier is integrated with a blockchain, this anchors the classifier's output. While a blockchain can guarantee the integrity of a record (i.e., that it has not been tampered with after submission), however, it cannot guarantee that the information was true at the moment it was recorded. If an external detector provides an incorrect classification, the blockchain preserves that error permanently. Thus, in this setting, the blockchain provides integrity of claims, not verification of their truth.

Content Provenance: Content provenance focuses on documenting the history of a digital asset from the time of its creation. Industry standards, such as the Coalition for Content Provenance and Authenticity (C2PA), allow creators or devices to attach cryptographically signed metadata claims (known as *content credentials*) to media, documenting the origin, authorship, and any subsequent edits [694]. In this model, a camera or a generative tool attaches content credentials to the file at creation time. Some companies and projects (e.g., Numbers Protocol [695] and Starling Lab [696]) use blockchains as a public immutable registry to log these content credentials. Everlyn [697], an AI video generation model, automatically anchors a cryptographic hash of its outputs to a decentralized ledger at creation time. This ensures that any video generated by the model has a permanent, publicly verifiable record that identifies it as AI generated. By posting a hash of the content credentials on-chain, these systems aim to maintain a transparent log that remains persistent even if the original file's metadata is stripped. While blockchains can act as a public, immutable registry for these signatures, the trust still resides in the hardware or software that generated the signature, not the blockchain itself.

Even a robust provenance system anchored to a blockchain cannot guarantee whether a piece of content was originally generated by a human or by AI. For example, a user could display an AI-generated image on a high-resolution monitor and then photograph it using a C2PA-compliant camera. The resulting file would contain valid, cryptographically signed credentials identifying it as an authentic photograph captured by a physical device. Similarly, a user could generate text with an AI system and then manually retype the same text into a C2PA-compliant editor, producing a file with legitimate provenance metadata indicating human authorship within the compliant tool.

Furthermore, if a piece of content—whether human- or AI-generated—has a blockchain record, but is then modified so that it can no longer be matched to that record, then its provenance will be lost. In the absence of a *universal* registry for content—which is improbable anytime in the foreseeable future—a provenance system will necessarily have large gaps.

Takeaway: While blockchains provide a robust mechanism for the integrity of provenance metadata in a narrow sense, they are far from a comprehensive solution to the GenAI detection problem. An effective solution would require a universal ecosystem where every piece of digital content is captured using a trusted device (e.g., a C2PA-compliant camera) and immediately anchored to a blockchain. In reality, the vast majority of digital content is currently created and shared using tools and platforms that do not support cryptographic anchoring, leaving unlabeled content in a state of ambiguity. Thus, while blockchains can act as a high-integrity registry for some content, their role is limited to preserving claims about content, not resolving the broader challenge of distinguishing human from AI-generated material.

Misconception D.2: Fair and Unbiased AI

Blockchains (or decentralization more broadly) can solve bias and fairness problems in AI.[Giulia]

Today, there is a common viewpoint that by running model inference and training on a blockchain, we can solve common problems of unfairness and bias in AI [698, 699, 700]. To evaluate this very broad statement, we must tease apart the different kinds of bias that can arise in ML.

Algorithmic bias: The most common notion of fairness in the AI community is algorithmic bias: models are known to learn (and sometimes amplify) imbalances in datasets [701, 702]. This can result in discriminative models that perform poorly on under-represented populations [703] and generative models that mimic undesirable properties or sentiments of their training data (e.g. using toxic language, perpetuating stereotypes) [704, 705]. For a fixed definition of algorithmic fairness, the ML community has proposed many technical solutions to enforce fairness, including at training time [706, 707, 708] and at inference time [709], e.g., guardrails for AI models [710, 711]. However, these protections are far from perfect; fairness is not considered a solved problem in the ML community, and may never be [712].

Even deciding how to define fairness is challenging, and typically requires making substantial tradeoffs [713].

Algorithmic bias is unlikely to be solved by decentralized AI, because it arises inherently in the training process and is typically mitigated by revised training or inference techniques. Hence, decentralization does not address the source of the problem. However, a second source of potential bias stems from higher-level decisions that affect model performance; examples include what data is used, what model architecture to adopt, and how to compensate stakeholders who contribute to a particular model. While this is orthogonal to how the AI community typically views fairness, it is something that could potentially impact algorithmic bias and partially be addressed with decentralization. Specifically, decentralization offers two desirable properties: (1) transparency and (2) decentralized governance.

Transparency: Since transparency is a central property of blockchains, AI model developers could use blockchains to publicly commit to training data, training algorithms, model checkpoints, and the kinds of inference-time guardrails that a model has in place. Here, by "transparency", we mean that operators can provably track the outputs of various operations, like a training run or a model inference.[1] Indeed, there are several platforms that aim to track such information for downstream model users [385, 484]. From this information, users could check, for instance, that their data was not used to train a model, or that the model explicitly filters outputs that are considered objectionable. While such transparency could have benefits, it is challenging to scale to larger models and training-time artifacts such as model checkpoints, due to high storage and computational costs [333]. In existing systems, much data from model training (e.g., training datasets, checkpoints) appears to be stored off-chain anyway and often cannot be accessed directly by users [**<empty citation>**]. [Giulia: James, is that true? Do you know of any examples or counterexamples?] Hence, in the short term, transparency benefits may be limited to inference [333]. [Giulia: Ittay, any comments?] **transparency alone may not significantly change how people use and develop AI unless the industry also thinks carefully about how people will use that transparency**: concretely, what kinds of use cases do we expect to be solved by model transparency, and based on this, what interfaces should be developed? For example, is a primary goal to allow users to report improper data usage in model training? If so, additional infrastructure will be required to establish true data ownership and address the problem (e.g. with machine unlearning). The many components of this pipeline are just as technically thorny, possibly more so, than transparency itself.

Decentralized governance: Finally, Crypto x AI platforms allow for decentralized governance models, as discussed in Section C-3.1. It is important to disambiguate between community governance *mechanisms* explored and adopted in blockchain systems, such as token-weighted voting and liquid democracy, and the decentralized, autonomous governance manifested by DAOs. In the former case, many technical and performance-sensitive decisions in AI development are poorly suited to broad stakeholder input. However, community governance mechanisms may be well-suited to value-laden decisions relating to model alignment, and have indeed been explored by major AI developers [539, 541], if not yet meaningfully deployed. Crucially, none of these mechanisms inherently require a blockchain to implement: it is therefore incorrect to characterize them as problems in AI solved *by* blockchain, even if blockchain-based systems could lend such processes additional transparency. True on-chain governance of AI would require that governance decisions be enforced by smart contract, whether through direct execution or through economic incentives such as stake slashing. Enforcement of this kind could increase the robustness of governance and strengthen user confidence in these systems. However, these potential benefits face many of the same technical barriers as blockchain-based transparency more broadly: current blockchain infrastructure is not well-suited to the storage and computational demands of AI development, and practical implementations would likely require significant advances in verifiable training. Blockchain-enforced governance of AI systems, while a coherent long-run vision, remains technically premature.

[1] Note that this notion of transparency does not explain *why* an operation gave a particular output.

Takeaway: While blockchains do not inherently help reduce algorithmic bias, they can indeed encourage transparency at various stages of the AI life cycle and broaden participation in AI governance. However, scalability challenges aside, the impact of these properties on downstream model outcomes remains unclear; practitioners should demonstrate through case studies and data how model transparency and alternative forms of model governance concretely impact end user and developer experiences.

Misconception D.3: Automation vs. Autonomy

Giving AI agents a crypto wallet allows them to earn, spend, and "survive" on their own, thereby making them autonomous. [Fan]

Projects building "agentic wallets" [714] and payment protocols [715, 716] often claim that giving AI agents a crypto wallet makes them autonomous, because decentralized payments allow them to earn, spend, and survive on their own. We argue that such claims conflate several distinct notions, leading to a number of misconceptions.

Part of the ambiguity stems from the fact that "autonomy" typically means different things in the contexts of AI and blockchains. In AI literature, an autonomous agent is a system that can act based on its own perception, learning, and experience rather than strictly following pre-programmed rules [717]. Perhaps confusingly, smart contracts are often described as autonomous, but here the emphasis is on their resilience to adversarial manipulations such as tampering, censorship, and shutdown. To differentiate, we refer to the form as *intelligence autonomy*, and the latter as *execution autonomy*. Modern AI agents already exhibit substantial intelligence autonomy, but not necessarily execution autonomy: system administrators can, for example, shut down the servers on which AI agents run.

In claims that agentic wallets enable AI agents to transact "autonomously" (e.g., as seen in [714, 715, 716]), the autonomy in question is neither intelligence autonomy nor execution autonomy. AI systems do not become more intelligent by possessing a wallet. Nor do they become more resistant to human manipulation or shutdown. Instead, having access to a wallet enables *automation*: AI agents can programmatically trade, transact, and access on-chain infrastructure without human approval loops.

It is also important to note that blockchains are not uniquely required for such automation. Centralized financial infrastructure can be, and has been, accessed programmatically by AI agents.

However, a more defensible interpretation of such claims is that blockchain-based payment systems themselves offer stronger autonomy (than centralized alternatives), even though they may not uniquely benefit AI agents. For instance, they can ensure that transactions by AI agents are not treated differently from transactions by humans (i.e., they offer neutrality and censorship resistance [718, 719, 720]).

Takeaway: Agentic wallets allow AI agents to conveniently access financial APIs, enabling economic interactions to be automated without human approval loops. However, automation should not be confused with autonomy: merely possessing a wallet does not make AI agents independent of human control (e.g., operators may still shut down the models or infrastructure they rely on). Moreover, automated payments do not require blockchains; similar functionality can exist in centralized financial systems. In comparison, however, Blockchain-based payment systems can offer appealing properties such as neutrality and censorship resistance, which can be desirable for applications where payment suppression, censorship, or other forms of manipulation are a concern.

Misconception D.4: *Transparent* AI=*Trustworthy* AI [Ari]

Recording models' data provenance and inferences on blockchains results in trustworthy model deployment and use.

The transparency and immutability of blockchains seem on the face of it to be ideal tools for ensuring the *trustworthiness* of AI models. This is the thesis of a widely referenced IBM blog post [721] and fits with some common misconceptions, which extend beyond models by implication also to AI agents.

Model transparency. Recording the provenance of a model's training data appears to create a form of transparency around the model's creation. But there's a large gap between a record of data provenance and assurance about the behavior of a model, because: (1) A blockchain record of provenance is not a *proof* of provenance, i.e., proof of the composition of a training data set (although evidence of provenance is possible using techniques discussed in Section C-6.1); (2) Even exact knowledge of a model's training data is insufficient to determine how the model will behave, because the *training procedure and computational environment* also determine model behavior; and (3) Even with knowledge of the full pipeline from data to model sufficient to replicate the model, the non-determinism inherent in most stochastic training renders it infeasible to verify model weights against a training pipeline, even in principle.

Finally, even with access to model weights, there is no generally effective mechanism for detecting backdoors or other adversarial manipulation introduced during training.

In short, recording information about model data and training on a blockchain does not provide direct assurance of its behavioral characteristics or the absence of adversarial manipulation.

Inference transparency. Records of model inputs and corresponding inferences can be recorded on blockchains to create apparent transparency around the *use* of models.

Blockchains, however, make transactions transparent, not reasoning. A blockchain transaction T stating that "model X was queried on input Y, yielding inference Z" does little to establish the trustworthiness of Z, because such a record alone does not establish:

- *Correct model execution:* T alone does not provide *proof* that the tuple (X, Y, Z) actually resulted from execution of model X as specified. (Such proof is possible, but requires use of TEEs or computationally expensive cryptographic techniques.)
- *Model trustworthiness:* Even if T did provide such proof, there is a more foundational problem. A full record of the provenance of model X, as described above, does not prove the trustworthiness of model X in a semantic sense, i.e., adherence to user expectations or industry or community norms. Specification of X by means of, e.g., a hash of model weights, provides even less meaningful assurance, as the *identity* of a model does not alone establish the *trustworthiness* of a model. As a result, it is hard to translate T into assurance of trustworthy inference.

Blockchains *are* helpful for certain goals relating to trustworthiness. For example, an organization can publish hashes of open-weight models on chain. Such hashes serve as an immutable reference, so that users know they're using an authentic, unmodified model. Similar ideas for tamper-evident logging have arisen for other applications: use of blockchains as records for firmware updates [722] and certificate transparency [723], a system that uses blockchain-like append-only logs to maintain a publicly auditable record of certificate issuances.

Takeaway: There is a considerable gap between the recording of model data provenance and model inferences on blockchains and meaningful assurance of model and inference trustworthiness.

Misconception D.5: Decentralized AI efficiency [Giulia]

Decentralization inherently makes AI jobs more cost-effective for model developers and users.

A prominent class of crypto x AI initiatives have proposed decentralized networks as enablers for more efficient and cost-effective AI. One important example is DeAI networks, like [Giulia: fill]. Other examples include decentralized physical infrastructure networks (DePIN) [384, 385, 386, 387, 388, 389] in which users rent out their own physical infrastructure (like GPUs). The main appeal of these decentralized networks is reduced cost; for example, renting a DePIN GPU can be substantially cheaper than renting one on a comparable cloud service provider [387, 386]. However, it is not always the case that cheaper machines lead to lower cost in AI jobs; we discuss this issue in some detail in Section C-2.1. The main message is that while some use cases are well-suited to DePIN, others could incur higher costs due to network costs. Since decentralized nodes and processes communicate over the public Internet, the throughput and latency requirements of an AI job can significantly impact the overall cost of a job. Moreover, very large AI jobs (such as training frontier models) are typically throughput-bound. Today, direct cost comparison is challenging because we lack systematic benchmarks that profile AI jobs on DePIN networks and compare them to traditional cloud infrastructure.

Takeaway: While decentralized networks offer an appealing alternative to high-cost centralized cloud providers, we do not have enough data to predict when a job will be cheaper on existing DePIN or DeAI platforms vs. a centralized cloud service provider. While smaller jobs (e.g., inference, small-scale training) are likely to be cheaper on these networks, very large jobs (like training foundation model) could be impacted by unreliable and low-bandwidth communication networks between nodes. More research is needed to understand these tradeoffs clearly.

Bibliography

[1] Michael Alles and Glen L Gray. "Hope or hype? Blockchain and accounting." In: *International Journal of Digital Accounting Research* 23 (2023).

[2] Emma Woollacott. *VC investment in AI is skyrocketing – funding in the first half of 2025 was more than the whole of last year, says EY — IT Pro*. [Online; accessed 2025-09-30]. Aug. 2025. URL: https://www.itpro.com/technology/artificial-intelligence/vc-investment-in-ai-is-skyrocketing-funding-in-the-first-half-of-2025-was-more-than-the-whole-of-last-year-says-ey.

[3] *Why VCs Are Betting on AI x Crypto as the Next Trillion-Dollar Market - PrimaFelicitas*. [Online; accessed 2025-09-30]. June 2025. URL: https://www.primafelicitas.com/artificial-intelligence/why-vcs-are-betting-on-ai-x-crypto-as-the-next-trillion-dollar-market/.

[4] Noor Bazmi. *Crypto funding for AI projects hits record $516 million in 2025*. [Online; accessed 2025-09-30]. URL: https://cryptorank.io/news/feed/c13e2-ai-driven-crypto-hit-funding.

[5] European Central Bank. *Paradise lost? How crypto failed to deliver on its promises and what to do about it*. [Online; accessed 2025-09-30]. June 2023. URL: https://www.ecb.europa.eu/press/key/date/2023/html/ecb.sp230623_1~80751450e6.en.html.

[6] Coinbase Institute. *Crypto and Agentic AI*. [Online; accessed 2025-09-30]. URL: https://www.coinbase.com/public-policy/advocacy/documents/crypto-and-agentic-ai.

[7] Vincent Maliepaard. *The power of agentic AI in crypto: A deep dive into the Virtuals ecosystem*. [Online; accessed 2025-09-30]. Jan. 2025. URL: https://cryptoslate.com/the-power-of-agentic-ai-in-crypto-a-deep-dive-into-the-virtuals-ecosystem/.

[8] Tonya M. Evans. *How AI And Blockchain Are Solving Each Other's Biggest Challenges*. [Online; accessed 2025-09-30]. Oct. 2024. URL: https://www.forbes.com/sites/tonyaevans/2024/10/29/how-ai-and-blockchain-are-solving-each-others-biggest-challenges/.

[9] Nir Kshetri. "Building Trust in AI: How Blockchain Enhances Data Integrity, Security, and Privacy". In: *Computer* 58.2 (2025), pp. 63–70.

[10] Flashbots. *BuilderNet: A Decentralized Block Building Network for Ethereum*. https://buildernet.org/docs. Public launch November 26, 2024. 2024.

[11] Hester M. Peirce. *Statement on sufficient decentralization*. https://x.com/HesterPeirce/status/1423637816492318722. Tweet, August 6, 2021. Aug. 2021.

[12] Financial Crimes Enforcement Network. *Application of FinCEN's Regulations to Persons Administering, Exchanging, or Using Virtual Currencies*. Guidance FIN-2013-G001, March 18, 2013. 2013.

[13] Magnus Van Haaren et al. "Humans and Algorithms in Organizations: Navigating the Intersection of Blockchain and AI". In: *84th Annual Meeting of the Academy of Management, AOM 2024*. 2024.

[14] Rasoul Amirzadeh, Asef Nazari, and Dhananjay Thiruvady. "Applying artificial intelligence in cryptocurrency markets: A survey". In: *Algorithms* 15.11 (2022), p. 428.

[15] Qinglin Yang et al. "Fusing blockchain and AI with metaverse: A survey". In: *IEEE Open Journal of the Computer Society* 3 (2022), pp. 122–136.

[16] Yiping Zuo et al. "A survey of blockchain and artificial intelligence for 6G wireless communications". In: *IEEE Communications Surveys & Tutorials* 25.4 (2023), pp. 2494–2528.

[17] Khaled Salah et al. "Blockchain for AI: Review and open research challenges". In: *IEEE access* 7 (2019), pp. 10127–10149.

[18] Vivek Pathak et al. "Qualitative survey on artificial intelligence integrated blockchain approach for 6G and beyond". In: *IEEE Access* 11 (2023), pp. 105935–105981.

[19] Adedoyin A Hussain and Fadi Al-Turjman. "Artificial intelligence and blockchain: A review". In: *Transactions on emerging telecommunications technologies* 32.9 (2021), e4268.

[20] Himanshu Srivastava et al. "BlockXAI: Review of blockchain for explainable artificial intelligence". In: *Convergence of Blockchain and Explainable Artificial Intelligence* (2024), pp. 1–14.

[21] Konstantin D Pandl et al. "On the convergence of artificial intelligence and distributed ledger technology: A scoping review and future research agenda". In: *IEEE access* 8 (2020), pp. 57075–57095.

[22] Jagger S Bellagarda and Adnan M Abu-Mahfouz. "An updated survey on the convergence of distributed ledger technology and artificial intelligence: Current state, major challenges and future direction". In: *IEEE Access* 10 (2022), pp. 50774–50793.

[23] Oumaima Fadi, Zkik Karim, Boulmalf Mohammed, et al. "A survey on blockchain and artificial intelligence technologies for enhancing security and privacy in smart environments". In: *IEEE Access* 10 (2022), pp. 93168–93186.

[24] Meng Shen et al. "Blockchains for artificial intelligence of things: A comprehensive survey". In: *IEEE Internet of Things Journal* 10.16 (2023), pp. 14483–14506.

[25] Yanjun Zuo. "Exploring the synergy: AI enhancing blockchain, blockchain empowering AI, and their convergence across IoT applications and beyond". In: *IEEE Internet of Things Journal* (2024).

[26] Yang Hu, Harold Glenn A Valera, and Les Oxley. "Market efficiency of the top market-cap cryptocurrencies: Further evidence from a panel framework". In: *Finance Research Letters* 31 (2019), pp. 138–145.

[27] Olga Labazova, Tobias Dehling, and Ali Sunyaev. *From hype to reality: A taxonomy of blockchain applications*. 2019.

[28] Gheyath Mustafa Zebari and Nasser Al Musalhi. "A Comprehensive Review of Integrating AI and Blockchain Security: Innovations, Challenges, and Future Directions". In: *Security and Privacy* 8.5 (2025), e70094.

[29] Md Monjurul Karim et al. "AI agents meet blockchain: A survey on secure and scalable collaboration for multi-agents". In: *Future Internet* 17.2 (2025), p. 57.

[30] Moetez Abdelhamid et al. "A review on blockchain technology, current challenges, and AI-driven solutions". In: *ACM Computing Surveys* 57.3 (2024), pp. 1–39.

[31] Mohsen Soori, Roza Dastres, and Behrooz Arezoo. "AI-powered blockchain technology in industry 4.0, a review". In: *Journal of Economy and Technology* 1 (2023), pp. 222–241.

[32] Manfred Spitzer. *The mind within the net: Models of learning, thinking, and acting*. Mit Press, 1999.

[33] Mohamed Sabt, Mohammed Achemlal, and Abdelmadjid Bouabdallah. "Trusted Execution Environment: What It Is, and What It Is Not". In: *2015 IEEE Trustcom/BigDataSE/ISPA*. Vol. 1. 2015, pp. 57–64. URL: https://hal.science/hal-01246364.

[34] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf. 2008.

[35] Gavin Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Tech. rep. Yellow Paper, EIP-150 revision and later updates. Ethereum Project, 2014.

[36] David Phelan. *iPhone Batterygate Latest: More iPhone Users Can Now Claim Payouts*. [Online; accessed 2025-08-28]. Apr. 2024. URL: https://www.forbes.com/sites/davidphelan/2024/04/07/iphone-batterygate-latest-more-iphone-users-can-now-claim-payouts/.

[37] Suzanne Phan. *California DMV claims Tesla misled drivers about self-driving capabilities; looks to suspend business for 30 days - ABC7 San Francisco*. [Online; accessed 2025-08-28]. July 2025. URL: https://abc7news.com/post/california-dmv-claims-tesla-misled-drivers-driving-capabilities-looks-suspend-business-30-days/17234677/.

[38] Mark Russinovich et al. "Confidential Computing Proofs: An alternative to cryptographic zero-knowledge". In: *Queue* 22.4 (2024), pp. 73–100.

[39] Helius. *A Complete History of Solana Outages: Causes, Fixes, and Lessons Learnt*. https://www.helius.dev/blog/solana-outages-complete-history. 2024.

[40] Christina Ovezik et al. "SoK: measuring blockchain decentralization". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2025, pp. 184–214.

[41] Andrés Fábrega et al. "Voting-Bloc Entropy: A New Metric for DAO Decentralization". In: *USENIX Security Symposium*. 2025.

[42] Tanusree Sharma et al. "Unpacking how decentralized autonomous organizations (daos) work in practice". In: *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE. 2024, pp. 416–424.

[43] Francisco Rodrigues. *Ethereum's Vitalik Buterin Proposes AI 'Stewards' to Help Reinvent DAO Governance*. CoinDesk. Feb. 2026. URL: https://www.coindesk.com/web3/2026/02/21/ethereum-s-vitalik-buterin-proposes-ai-stewards-to-help-reinvent-dao-governance.

[44] *What is Middleware - Definition and Examples — Microsoft Azure*. [Online; accessed 2025-09-30]. URL: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-middleware.

[45] Aaron Gember et al. "Toward software-defined middlebox networking". In: *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*. 2012, pp. 7–12.

[46] Muhammad Izhar Mehar et al. "Understanding a revolutionary and flawed grand experiment in blockchain: The DAO attack". In: *Journal of Cases on Information Technology (JCIT)* 21.1 (2019), pp. 19–32.

[47] Aztec Labs. *Introducing Aztec.nr: Aztec's Private Smart Contract Framework*. Aztec Network Blog, https://aztec.network/blog/introducing-aztec-nr-aztecs-private-smart-contract-framework. (Visited on 04/17/2026).

[48] Aleo Network Foundation. *Aleo VM Specification*. https://developer.aleo.org/specs/aleovm.pdf. Nov. 2025. (Visited on 04/17/2026).

[49] Penumbra Labs. *The Penumbra Protocol: A Fully Private Proof-of-Stake Network and Decentralized Exchange for the Cosmos Ecosystem*. https://protocol.penumbra.zone/main/index.html. Living protocol specification; no static whitepaper PDF. Lead: Henry de Valence. (Visited on 04/17/2026).

[50] Patrick Jauernig, Ahmad-Reza Sadeghi, and Emmanuel Stapf. "Trusted Execution Environments: Properties, Applications, and Challenges". In: *IEEE Security & Privacy* 18.2 (2020). Already in the main survey bibliography as [30]; this entry exists for use in this subsection's draft., pp. 56–60.

[51] Jianwei Zhu et al. *Confidential Computing on NVIDIA Hopper GPUs: A Performance Benchmark Study*. 2024. arXiv: 2409.03992 [cs.DC].

[52] Mark Russinovich. *Azure AI Confidential Inferencing: Technical Deep-Dive*. Microsoft Azure Confidential Computing Blog. Updated December 16, 2025. Sept. 2024. URL: https://techcommunity.microsoft.com/blog/azureconfidentialcomputingblog/azure-ai-confidential-inferencing-technical-deep-dive/4253150 (visited on 04/17/2026).

[53] Anthropic and Pattern Labs. *Confidential Inference Systems: Design Principles and Security Risks*. Whitepaper Version 1.0. Anthropic, June 2025. URL: https://assets.anthropic.com/m/c52125297b85a42/original/Confidential_Inference_Paper.pdf (visited on 04/17/2026).

[54] NEAR. *Introducing NEAR AI Cloud: Private Chat*. NEAR Blog. URL: https://near.ai/blog/introducing-near-ai-cloud-private-chat (visited on 04/17/2026).

[55] Hashforest Technology. *RedPill: Private ChatGPT, End-to-End Encrypted, Powered by Confidential AI*. https://www.redpill.ai/. 2026.

[56] Venice.ai. *Venice Launches End-to-End Encrypted AI*. Venice.ai Blog. Mar. 2026. URL: https://venice.ai/blog/venice-launches-end-to-end-encrypted-ai (visited on 04/17/2026).

[57] Stephan van Schaik et al. "SoK: SGX.Fail — How Stuff Gets eXposed". In: *2024 IEEE Symposium on Security and Privacy (SP)*. 2024. URL: https://sgx.fail/files/sgx.fail.pdf.

[58] Jalen Chuang et al. "TEE.fail: Breaking Trusted Execution Environments via DDR5 Memory Bus Interposition". In: *Proceedings of the 47th IEEE Symposium on Security and Privacy*. S&P 2026. Publicly disclosed October 2025. 2026. URL: https://tee.fail/files/paper.pdf.

[59] Simon Johnson et al. *Supporting Intel SGX on Multi-Package Platforms*. Intel's documentation of scalable SGX memory encryption and its tradeoffs vs. the original integrity-tree design. 2025. arXiv: 2507.08190 [cs.CR]. URL: https://arxiv.org/abs/2507.08190.

[60] AMD. *AMD SEV-SNP: Strengthening VM Isolation with Integrity Protection and More*. White Paper. Advanced Micro Devices, 2020. URL: https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/SEV-SNP-strengthening-vm-isolation-with-integrity-protection-and-more.pdf.

[61] Microsoft. *About Azure Confidential VMs*. Microsoft Learn, https://learn.microsoft.com/en-us/azure/confidential-computing/confidential-vm-overview. Azure Confidential VMs on AMD SEV-SNP (DCasv5, ECasv5, DCasv6, ECasv6) and Intel TDX (DCesv5, ECesv5). (Visited on 04/19/2026).

[62] Google Cloud. *Confidential Computing Overview*. Google Cloud Documentation, https://docs.cloud.google.com/confidential-computing/docs/confidential-computing-overview. Google Cloud Confidential Computing product family: Confidential VM, Confidential Space, Google Cloud Attestation, Split-trust encryption tool. (Visited on 04/19/2026).

[63] Filip Rezabek et al. *Proof of Cloud: Data Center Execution Assurance for Confidential VMs*. 2025. arXiv: 2510.12469 [cs.CR]. URL: https://arxiv.org/abs/2510.12469.

[64] Proof of Cloud Alliance. *Proof of Cloud Alliance: A Vendor-Neutral Registry of Verified Confidential Computing Hardware*. https://proofofcloud.org/. (Visited on 04/17/2026).

[65] Pietro Borrello et al. "ÆPIC Leak: Architecturally Leaking Uninitialized Data from the Microarchitecture". In: *31st USENIX Security Symposium (USENIX Security 22)*. 2022, pp. 3917–3934. URL: https://aepicleak.com/.

[66] Amanda Silberling. *Your public ChatGPT queries are getting indexed by Google and other search engines*. TechCrunch. July 31, 2025. URL: https://techcrunch.com/2025/07/31/your-public-chatgpt-queries-are-getting-indexed-by-google-and-other-search-engines/ (visited on 08/11/2025).

[67] Gobikrishna Dhanuskodi et al. "Creating the first confidential gpus". In: *Communications of the ACM* 67.1 (2023), pp. 60–67.

[68] NVIDIA. *NVIDIA Secure AI with Blackwell and Hopper GPUs*. Tech. rep. Whitepaper. 2025 update.

[69] NVIDIA. *AI Security with Confidential Computing*. https://www.nvidia.com/en-us/data-center/solutions/confidential-computing/. Accessed April 2026. 2025.

[70] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. "The knowledge complexity of interactive proof-systems". In: *Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali*. 2019, pp. 203–225.

[71] Alessandro Chiesa. "Succinct non-Interactive arguments". PhD thesis. Massachusetts Institute of Technology, 2014.

[72] Vitalik Buterin. *An Incomplete Guide to Rollups*. https://vitalik.eth.limo/general/2021/01/05/rollup.html. Accessed: 2026-01-28. Jan. 2021.

[73] Matter Labs. *zkSync Era*. https://www.zksync.io. Accessed: 2026-01-28. 2023.

[74] Starknet-io. *StarkNet*. https://www.starknet.io. Accessed: 2026-01-28. 2025.

[75] Polygon Labs. *Polygon zkEVM: EVM-equivalent Zero-Knowledge Rollup*. https://polygon.technology/polygon-zkevm. 2023.

[76] Jiaheng Zhang et al. "Zero knowledge proofs for decision tree predictions and accuracy". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 2020, pp. 2039–2053.

[77] Bing-Jyue Chen et al. "Zkml: An optimizing system for ml inference in zero-knowledge proofs". In: *Proceedings of the Nineteenth European Conference on Computer Systems*. 2024, pp. 560–574.

[78] Modulus-Labs. *RockyBot: Fully On-Chain AI Trading Bot*. https://github.com/Modulus-Labs/RockyBot. Accessed: 2026-01-28. 2023.

[79] Modulus Labs. *Modulus Labs: Accountable Machine Intelligence*. Zero-knowledge machine learning platform. Accessed: 2024-12-19. 2023. URL: https://www.modulus.xyz/.

[80] Modulus Labs. *The Cost of Intelligence: Proving Machine Learning Inference with Zero-Knowledge*. https://github.com/Modulus-Labs/Papers/blob/master/Cost_Of_Intelligence.pdf. Jan. 2023.

[81] Andrew C Yao. "Protocols for secure computations". In: *23rd annual symposium on foundations of computer science (sfcs 1982)*. IEEE. 1982, pp. 160–164.

[82] Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to play any mental game, or a completeness theorem for protocols with honest majority". In: *Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali*. 2019, pp. 307–328.

[83] Alex Ozdemir and Dan Boneh. "Experimenting with collaborative {zk-SNARKs}:{Zero-Knowledge} proofs for distributed secrets". In: *31st USENIX Security Symposium (USENIX Security 22)*. 2022, pp. 4291–4308.

[84] Yvo Desmedt. "Threshold cryptosystems". In: *international workshop on the theory and application of cryptographic techniques*. Springer. 1992, pp. 1–14.

[85] Fan Zhang et al. "DECO: Liberating web data using decentralized oracles for TLS". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 2020, pp. 1919–1938.

[86] Sameer Wagh, Divya Gupta, and Nishanth Chandran. "SecureNN: 3-party secure computation for neural network training". In: *Proceedings on Privacy Enhancing Technologies* (2019).

[87] Nishant Kumar et al. "Cryptflow: Secure tensorflow inference". In: *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2020, pp. 336–353.

[88] Ye Dong et al. "Puma: Secure inference of llama-7b in five minutes". In: *Security and Safety* 4 (2025), p. 2025014.

[89] Jianwei Zhu et al. "Confidential computing on NVIDIA Hopper GPUs: a performance benchmark study". In: *arXiv preprint arXiv:2409.03992* (2024).

[90] *Bringing Innovation to the next Level*. https://www.axa.com/en/news/bringing-innovation-to-the-next-level. (Visited on 01/30/2026).

[91] *Chainlink*. URL: `https://chain.link/` (visited on 02/06/2026).

[92] *RedStone Oracles*. URL: `https://redstone.finance/` (visited on 02/06/2026).

[93] *Chronicle*. URL: `https://chroniclelabs.org/` (visited on 02/06/2026).

[94] *Witnet*. URL: `https://witnet.io/` (visited on 02/06/2026).

[95] *UMA Optimistic Oracle*. URL: `https://uma.xyz/` (visited on 02/06/2026).

[96] *Tellor*. URL: `https://tellor.io/` (visited on 02/06/2026).

[97] *Band Protocol*. URL: `https://bandprotocol.com/` (visited on 02/06/2026).

[98] *Pyth Network*. URL: `https://pyth.network/` (visited on 02/06/2026).

[99] *API3*. URL: `https://www.api3.org/` (visited on 02/06/2026).

[100] *Supra*. URL: `https://supra.com/` (visited on 02/06/2026).

[101] *Gas Network*. URL: `https://gas.network/` (visited on 02/06/2026).

[102] Lorenz Breidenbach et al. "Chainlink Off-chain Reporting Protocol". In: (). `https://research.chain.link/ocr.pdf`.

[103] Fan Zhang et al. "Town Crier: An authenticated data feed for smart contracts". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 270–282.

[104] Zhongtang Luo et al. "Proxying Is Enough: Security of Proxying in TLS Oracles and AEAD Context Unforgeability". In: *7th Conference on Advances in Financial Technologies (AFT 2025)*. 7th Conference on Advances in Financial Technologies (AFT 2025). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. DOI: `10.4230/LIPIcs.AFT.2025.4`. URL: `https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.AFT.2025.4` (visited on 02/06/2026).

[105] Hubert Ritzdorf et al. *TLS-N: Non-repudiation over TLS Enabling - Ubiquitous Content Signing for Disintermediation*. 2017. URL: `https://eprint.iacr.org/2017/578` (visited on 02/19/2026). Pre-published.

[106] Andrew C. Yao. "Protocols for Secure Computations". In: *23rd Annual Symposium on Foundations of Computer Science (Sfcs 1982)*. IEEE, 1982, pp. 160–164. URL: `https://ieeexplore.ieee.org/abstract/document/4568388/?casa_token=aqUoppWzsWkAAAAA:69oruK06wf22Zt8sqP0KHtcLkVM0bV9` (visited on 02/06/2026).

[107] Jens Groth. *On the Size of Pairing-based Non-interactive Arguments*. 2016. URL: `https://eprint.iacr.org/2016/260` (visited on 05/23/2024). Pre-published.

[108] Deepak Maram et al. "CanDID: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2021, pp. 1348–1366.

[109] Chainlink. *Introducing Chainlink Runtime Environment (CRE)*. Oct. 30, 2024. URL: https://blog.chain.link/introducing-chainlink-runtime-environment/.

[110] *Reclaim Protocol*. URL: https://reclaimprotocol.org/ (visited on 02/06/2026).

[111] *zkPass: zkTLS Oracle Protocol*. URL: https://docs.zkpass.org/overview/technical-overview (visited on 02/06/2026).

[112] *TLSNotary*. URL: https://tlsnotary.org/docs/intro (visited on 02/06/2026).

[113] Shehar Bano et al. "Twins: Bft systems made robust". In: *arXiv preprint arXiv:2004.10617* (2020).

[114] Zhuo Zhang et al. "Demystifying exploitable bugs in smart contracts". In: *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE. 2023, pp. 615–627.

[115] Michael Fröhlich et al. "Don't stop me now! exploring challenges of first-time cryptocurrency users". In: *Proceedings of the 2021 ACM designing interactive systems conference*. 2021, pp. 138–148.

[116] Ittay Eyal and Emin Gün Sirer. "Majority is not enough: Bitcoin mining is vulnerable". In: *Communications of the ACM* 61.7 (2018), pp. 95–102.

[117] Colin Finkbeiner et al. "SoK: Time to be selfless?! Demystifying the landscape of selfish mining strategies and models". In: *Cryptology ePrint Archive* (2025).

[118] Kartik Nayak et al. "Stubborn mining: Generalizing selfish mining and combining with an eclipse attack". In: *2016 ieee european symposium on security and privacy (euros&P)*. IEEE. 2016, pp. 305–320.

[119] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. "Optimal selfish mining strategies in bitcoin". In: *International conference on financial cryptography and data security*. Springer. 2016, pp. 515–532.

[120] Taotao Wang, Soung Chang Liew, and Shengli Zhang. "When blockchain meets AI: Optimal mining strategy achieved by machine learning". In: *International Journal of Intelligent Systems* 36.5 (2021), pp. 2183–2207.

[121] Roi Bar Zur, Ittay Eyal, and Aviv Tamar. "Efficient MDP analysis for selfish-mining in blockchains". In: *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*. 2020, pp. 113–131.

[122] Arthur Gervais et al. "On the security and performance of proof of work blockchains". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 3–16.

[123] Guoyu Yang et al. "IPBSM: An optimal bribery selfish mining in the presence of intelligent and pure attackers". In: *International Journal of Intelligent Systems* 35.11 (2020), pp. 1735–1748.

[124] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. Vol. 1. 1. MIT press Cambridge, 1998.

[125] Charlie Hou et al. "SquirRL: Automating attack analysis on blockchain incentive mechanisms with deep reinforcement learning". In: *arXiv preprint arXiv:1912.01798* (2019).

[126] Roi Bar-Zur et al. "WeRLman: To Tackle Whale (Transactions), Go Deep (RL)". In: *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2023, pp. 93–110.

[127] Roi Bar-Zur et al. "Deep bribe: Predicting the rise of bribery in blockchain mining with deep RL". In: *2023 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2023, pp. 29–37.

[128] Roozbeh Sarenche, Svetla Nikova, and Bart Preneel. "Deep selfish proposing in longest-chain proof-of-stake protocols". In: *International Conference on Financial Cryptography and Data Security*. Springer. 2024, pp. 24–40.

[129] Roozbeh Sarenche et al. “Bitcoin under volatile block rewards: How mempool statistics can influence bitcoin mining”. In: *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*. 2025, pp. 903–917.

[130] Krishnendu Chatterjee et al. “Fully automated selfish mining analysis in efficient proof systems blockchains”. In: *Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing*. 2024, pp. 268–278.

[131] Patrik Keller. “Generic selfish mining mdp for dag protocols”. In: *arXiv preprint arXiv:2309.11924* (2023).

[132] Roi Bar-Zur, Aviv Tamar, and Ittay Eyal. “MAD-DAG: Protecting Blockchain Consensus from MEV”. In: *arXiv preprint arXiv:2511.21552* (2025).

[133] Swaroopa Reddy and GVV Sharma. “UL-blockDAG: Unsupervised learning based consensus protocol for blockchain”. In: *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. IEEE. 2020, pp. 1243–1248.

[134] Kotteswaran Venkatesan and Syarifah Bahiyah Rahayu. “Blockchain security enhancement: an approach towards hybrid consensus algorithms and machine learning techniques”. In: *Scientific Reports* 14.1 (2024), p. 1149.

[135] Zhaojie Wang et al. “ForkDec: accurate detection for selfish mining attacks”. In: *Security and Communication Networks* 2021.1 (2021), p. 5959698.

[136] Yilei Wang et al. “A detection method against selfish mining-like attacks based on ensemble deep learning in IoT”. In: *IEEE Internet of Things Journal* 11.11 (2024), pp. 19564–19574.

[137] Qihao Bao et al. “SeMi_Detector: Multilayer Perceptron-Based Selfish Mining Detection”. In: *Peer-to-Peer Networking and Applications* 18.6 (2025), p. 327.

[138] Maryam Bahrani and S Matthew Weinberg. “Undetectable selfish mining”. In: *Proceedings of the 25th ACM Conference on Economics and Computation*. 2024, pp. 1017–1044.

[139] Guangquan Xu et al. “Am I eclipsed? A smart detector of eclipse attacks for Ethereum”. In: *Computers & Security* 88 (2020), p. 101604.

[140] Dhanasak Bhumichai and Ryan Benton. “Detection of ethereum eclipse attack based on hybrid method and dynamic weighted entropy”. In: *SoutheastCon 2023*. IEEE. 2023, pp. 779–786.

[141] Qianyi Dai, Bin Zhang, and Shuqin Dong. “Eclipse attack detection for blockchain network layer based on deep feature extraction”. In: *Wireless Communications and Mobile Computing* 2022.1 (2022), p. 1451813.

[142] Zubaida Rehman et al. “Eclipse attacks in blockchain networks: detection, prevention, and future directions”. In: *IEEE Access* 13 (2025), pp. 25918–25933.

[143] NI Indera et al. “Non-linear autoregressive with exogeneous input (NARX) Bitcoin price prediction model using PSO-optimized parameters and moving average technical indicators”. In: *Journal of fundamental and applied sciences* 9.3S (2017), pp. 791–808.

[144] Huisu Jang and Jaewook Lee. “An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information”. In: *IEEE Access* 6 (2018), pp. 5427–5437.

[145] Muhammad Saad et al. “Toward characterizing blockchain-based cryptocurrencies for highly accurate predictions”. In: *IEEE Systems Journal* 14.1 (2019), pp. 321–332.

[146] Han-Min Kim, Gee-Woo Bock, and Gunwoong Lee. “Predicting Ethereum prices with machine learning based on Blockchain information”. In: *Expert Systems with Applications* 184 (2021), p. 115480.

[147] Nishant Jagannath et al. “An on-chain analysis-based approach to predict ethereum prices”. In: *IEEE Access* 9 (2021), pp. 167972–167989.

[148] Junwei Chen. "Analysis of bitcoin price prediction using machine learning". In: *Journal of risk and financial management* 16.1 (2023), p. 51.

[149] Adrián Viéitez, Matilde Santos, and Rodrigo Naranjo. "Machine learning Ethereum cryptocurrency prediction and knowledge-based investment strategies". In: *Knowledge-Based Systems* 299 (2024), p. 112088.

[150] Huili Chen et al. "Deepmarks: A secure fingerprinting framework for digital rights management of deep learning models". In: *Proceedings of the 2019 on international conference on multimedia retrieval*. 2019, pp. 105–113.

[151] *ElizaOS: Autonomous Agents for Everyone*. `https://github.com/elizaOS/eliza`. Accessed: 2025-03-12.

[152] Daisuke Kawai et al. "Is your digital neighbor a reliable investment advisor?" In: *Proceedings of the ACM Web Conference 2023*. 2023, pp. 3581–3591.

[153] Zhe Li et al. "Tsfm-bench: A comprehensive and unified benchmark of foundation models for time series forecasting". In: *KDD*. 2025, pp. 5595–5606.

[154] Abdul Fatir Ansari et al. "Chronos: Learning the language of time series". In: *arXiv preprint arXiv:2403.07815* (2024).

[155] Abhimanyu Das et al. "A decoder-only foundation model for time-series forecasting". In: *ICML*. 2024.

[156] Ben Cohen et al. "This time is different: An observability perspective on time series foundation models". In: *arXiv preprint arXiv:2505.14766* (2025).

[157] Lingzhe Zhang et al. "A survey of aiops in the era of large language models". In: *ACM Computing Surveys* 58.2 (2025), pp. 1–35.

[158] Qian Cheng et al. "Ai for it operations (aiops) on cloud platforms: Reviews, opportunities and challenges". In: *arXiv preprint arXiv:2304.04661* (2023).

[159] Ted Young and Austin Parker. *Learning OpenTelemetry*. " O'Reilly Media, Inc.", 2024.

[160] Loi Luu et al. "Making Smart Contracts Smarter". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. Vienna, Austria: Association for Computing Machinery, 2016, pp. 254–269. ISBN: 9781450341394. DOI: 10.1145/2976749.2978309. URL: https://doi.org/10.1145/2976749.2978309.

[161] Petar Tsankov et al. "Securify: Practical Security Analysis of Smart Contracts". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. CCS '18. Toronto, Canada: Association for Computing Machinery, 2018, pp. 67–82. ISBN: 9781450356930. DOI: 10.1145/3243734.3243780. URL: https://doi.org/10.1145/3243734.3243780.

[162] Ivica Nikolić et al. "Finding The Greedy, Prodigal, and Suicidal Contracts at Scale". In: *Proceedings of the 34th Annual Computer Security Applications Conference*. ACSAC '18. San Juan, PR, USA: Association for Computing Machinery, 2018, pp. 653–663. ISBN: 9781450365697. DOI: 10.1145/3274694.3274743. URL: https://doi.org/10.1145/3274694.3274743.

[163] Sukrit Kalra et al. "ZEUS: Analyzing Safety of Smart Contracts". In: *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. 2018. URL: http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018%5C_09-1%5C_Kalra%5C_paper.pdf.

[164] Anton Permenev et al. "Verx: Safety verification of smart contracts". In: *2020 IEEE symposium on security and privacy (SP)*. IEEE. 2020, pp. 1661–1677.

[165] Liyi Zhou et al. " SoK: Decentralized Finance (DeFi) Attacks ". In: *2023 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, 2023, pp. 2444–2461. DOI: 10.1109/SP46215.2023.10179435. URL: https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.10179435.

[166] Neville Grech et al. “Gigahorse: Thorough, Declarative Decompilation of Smart Contracts”. In: *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. 2019, pp. 1176–1186. DOI: 10.1109/ICSE.2019.00120.

[167] Lexi Brent et al. “Ethainter: a smart contract security analyzer for composite vulnerabilities”. In: *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*. PLDI 2020. London, UK: Association for Computing Machinery, 2020, pp. 454–469. ISBN: 9781450376136. DOI: 10.1145/3385412.3385990. URL: https://doi.org/10.1145/3385412.3385990.

[168] Joel Frank, Cornelius Aschermann, and Thorsten Holz. “ETHBMC: A Bounded Model Checker for Smart Contracts”. In: *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020, pp. 2757–2774. ISBN: 978-1-939133-17-5. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/frank.

[169] Fabio Gritti et al. “Confusum Contractum: Confused Deputy Vulnerabilities in Ethereum Smart Contracts”. In: *32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, Aug. 2023, pp. 1793–1810. ISBN: 978-1-939133-37-3. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/gritti.

[170] Nicola Ruaro et al. “Not your Type! Detecting Storage Collision Vulnerabilities in Ethereum Smart Contracts.” In: *NDSS*. 2024.

[171] Bo Jiang, Ye Liu, and W. K. Chan. “ContractFuzzer: fuzzing smart contracts for vulnerability detection”. In: *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. ASE ’18. Montpellier, France: Association for Computing Machinery, 2018, pp. 259–269. ISBN: 9781450359375. DOI: 10.1145/3238147.3238177. URL: https://doi.org/10.1145/3238147.3238177.

[172] Gustavo Grieco et al. “Echidna: effective, usable, and fast fuzzing for smart contracts”. In: *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ISSTA 2020. Virtual Event, USA: Association for Computing Machinery, 2020, pp. 557–560. ISBN: 9781450380089. DOI: 10.1145/3395363.3404366. URL: https://doi.org/10.1145/3395363.3404366.

[173] Chaofan Shou, Shangyin Tan, and Koushik Sen. “ItyFuzz: Snapshot-Based Fuzzer for Smart Contract”. In: *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*. ISSTA 2023. Seattle, WA, USA: Association for Computing Machinery, 2023, pp. 322–333. ISBN: 9798400702211. DOI: 10.1145/3597926.3598059. URL: https://doi.org/10.1145/3597926.3598059.

[174] Yi Zhou et al. “Erays: Reverse Engineering Ethereum’s Opaque Smart Contracts”. In: *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 1371–1385. ISBN: 978-1-939133-04-5. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/zhou.

[175] Neville Grech et al. “Elipmoc: advanced decompilation of Ethereum smart contracts”. In: *Proc. ACM Program. Lang.* 6.OOPSLA1 (Apr. 2022). DOI: 10.1145/3527321. URL: https://doi.org/10.1145/3527321.

[176] Kunsong Zhao et al. “DeepInfer: Deep Type Inference from Smart Contract Bytecode”. In: *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ESEC/FSE 2023. San Francisco, CA, USA: Association for Computing Machinery, 2023, pp. 745–757. ISBN: 9798400703270. DOI: 10.1145/3611643.3616343. URL: https://doi.org/10.1145/3611643.3616343.

[177] Zeqin Liao et al. “Augmenting Smart Contract Decompiler Output Through Fine-Grained Dependency Analysis and LLM-Facilitated Semantic Recovery”. In: *IEEE Transactions on Software Engineering* 51.12 (2025), pp. 3574–3590. DOI: 10.1109/TSE.2025.3623325.

[178] Xing Su et al. “DiSCo: Towards Decompiling EVM Bytecode to Source Code using Large Language Models”. In: *Proc. ACM Softw. Eng.* 2.FSE (June 2025). DOI: 10.1145/3729373. URL: https://doi.org/10.1145/3729373.

[179] Isaac David et al. “Decompiling Smart Contracts with a Large Language Model”. In: *arXiv preprint arXiv:2506.19624* (2025).

[180] Isaac David et al. “Do you still need a manual smart contract audit?” In: *arXiv preprint arXiv:2306.12338* (2023).

[181] Chong Chen et al. “When ChatGPT Meets Smart Contract Vulnerability Detection: How Far Are We?” In: *ACM Trans. Softw. Eng. Methodol.* 34.4 (Apr. 2025). ISSN: 1049-331X. DOI: 10.1145/3702973. URL: https://doi.org/10.1145/3702973.

[182] Ziwei Ji et al. “Survey of hallucination in natural language generation”. In: *ACM computing surveys* 55.12 (2023), pp. 1–38.

[183] Yue Zhang et al. “Siren’s Song in the AI Ocean: A Survey on Hallucination in Large Language Models”. In: *Computational Linguistics* 51.4 (2025), pp. 1373–1418. ISSN: 0891-2017. DOI: 10.1162/COLI.a.16. URL: https://doi.org/10.1162/COLI.a.16.

[184] Sunbeom So, Seongjoon Hong, and Hakjoo Oh. “SmarTest: Effectively Hunting Vulnerable Transaction Sequences in Smart Contracts through Language Model-Guided Symbolic Execution”. In: *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 1361–1378. ISBN: 978-1-939133-24-3. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/so.

[185] Yuqiang Sun et al. “GPTScan: Detecting Logic Vulnerabilities in Smart Contracts by Combining GPT with Program Analysis”. In: *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. ICSE ’24. Lisbon, Portugal: Association for Computing Machinery, 2024. ISBN: 9798400702174. DOI: 10.1145/3597503.3639117. URL: https://doi.org/10.1145/3597503.3639117.

[186] Brian Zhang and Zhuo Zhang. “Detecting bugs with substantial monetary consequences by LLM and rule-based reasoning”. In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 133999–134023.

[187] Rundong Gan et al. “DeFiAligner: Leveraging Symbolic Analysis and Large Language Models for Inconsistency Detection in Decentralized Finance”. In: *6th Conference on Advances in Financial Technologies (AFT 2024)*. Ed. by Rainer Böhme and Lucianna Kiffer. Vol. 316. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, 7:1–7:24. ISBN: 978-3-95977-345-4. DOI: 10.4230/LIPIcs.AFT.2024.7. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.AFT.2024.7.

[188] Shuo Yang et al. “Hyperion: Unveiling DApp Inconsistencies Using LLM and Dataflow-Guided Symbolic Execution”. In: *Proceedings of the IEEE/ACM 47th International Conference on Software Engineering*. IEEE Press, 2025, pp. 2125–2137. ISBN: 9798331505691. URL: https://doi.org/10.1109/ICSE55347.2025.00015.

[189] Ye Liu et al. “PropertyGPT: LLM-driven Formal Verification of Smart Contracts through Retrieval-Augmented Property Generation”. In: *arXiv preprint arXiv:2405.02580* (2024).

[190] Sally Junsong Wang, Kexin Pei, and Junfeng Yang. “ SmartInv: Multimodal Learning for Smart Contract Invariant Inference ”. In: *2024 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2024, pp. 2217–2235. DOI: 10.1109/SP54263.2024.00126. URL: https://doi.ieeecomputersociety.org/10.1109/SP54263.2024.00126.

[191] Philip Daian et al. “Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability”. In: *2020 IEEE symposium on security and privacy (SP)*. IEEE. 2020, pp. 910–927.

[192] Liyi Zhou et al. “On the just-in-time discovery of profit-generating transactions in defi protocols”. In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2021, pp. 919–936.

[193] Ye Wang et al. “Cyclic arbitrage in decentralized exchanges”. In: *Companion Proceedings of the Web Conference 2022*. 2022, pp. 12–19.

[194] Liyi Zhou et al. “High-frequency trading on decentralized on-chain exchanges”. In: *2021 IEEE symposium on security and privacy (SP)*. IEEE. 2021, pp. 428–445.

[195] Kaihua Qin et al. “An empirical study of defi liquidations: Incentives, risks, and instabilities”. In: *Proceedings of the 21st ACM internet measurement conference*. 2021, pp. 336–350.

[196] Kushal Babel et al. “Clockwork finance: Automated analysis of economic security in smart contracts”. In: *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2023, pp. 2499–2516.

[197] Kushal Babel et al. “Lanturn: Measuring economic security of smart contracts through adaptive learning”. In: *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 2023, pp. 1212–1226.

[198] Weimin Chen and Xiapu Luo. “MEVisor: High-Throughput MEV Discovery in DEXs with GPU Parallelism”. In: *Network and Distributed System Security Symposium (NDSS)*. 2026. DOI: 10.14722/ndss.2026.240093.

[199] Christof Ferreira Torres, Ramiro Camino, et al. “Frontrunner jones and the raiders of the dark forest: An empirical study of frontrunning on the ethereum blockchain”. In: *30th USENIX Security Symposium (USENIX Security 21)*. 2021, pp. 1343–1359.

[200] Kaihua Qin, Liyi Zhou, and Arthur Gervais. “Quantifying blockchain extractable value: How dark is the forest?” In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2022, pp. 198–214.

[201] Robert McLaughlin, Christopher Kruegel, and Giovanni Vigna. “A large scale study of the ethereum arbitrage ecosystem”. In: *32nd USENIX Security Symposium (USENIX Security 23)*. 2023, pp. 3295–3312.

[202] Zihao Li et al. “Demystifying defi mev activities in flashbots bundle”. In: *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 2023, pp. 165–179.

[203] Ferenc Béres et al. “Blockchain is Watching You: Profiling and Deanonymizing Ethereum Users”. In: *2021 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. 2021, pp. 69–78. DOI: 10.1109/DAPPS52256.2021.00013.

[204] Jieli Liu et al. “Fishing for Fraudsters: Uncovering Ethereum Phishing Gangs With Blockchain Data”. In: *IEEE Transactions on Information Forensics and Security* 19 (2024), pp. 3038–3050. DOI: 10.1109/TIFS.2024.3359000.

[205] Dan Lin et al. “T-edge: Temporal weighted multidigraph embedding for ethereum transaction network analysis”. In: *Frontiers in Physics* 8 (2020), p. 204.

[206] Jie Shen et al. “Identity inference on blockchain using graph neural network”. In: *International conference on blockchain and trustworthy systems*. Springer. 2021, pp. 3–17.

[207] Jiajun Zhou et al. “Behavior-aware account de-anonymization on Ethereum interaction graph”. In: *IEEE Transactions on Information Forensics and Security* 17 (2022), pp. 3433–3448.

[208] Sijia Li et al. “TTAGN: Temporal transaction aggregation graph network for ethereum phishing scams detection”. In: *Proceedings of the ACM web conference 2022*. 2022, pp. 661–669.

[209] Haibin Zheng et al. “Tegdetector: a phishing detector that knows evolving transaction behaviors”. In: *IEEE Transactions on Computational Social Systems* 11.3 (2023), pp. 3988–4000.

[210] Lei Yu et al. "Who are the money launderers? Money laundering detection on blockchain via mutual learning-based graph neural network". In: *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2023, pp. 1–8.

[211] Hanbiao Du et al. "Breaking the anonymity of ethereum mixing services using graph feature learning". In: *IEEE Transactions on Information Forensics and Security* 19 (2023), pp. 616–631.

[212] Sihao Hu et al. "Bert4eth: A pre-trained transformer for ethereum fraud detection". In: *Proceedings of the ACM Web Conference 2023*. 2023, pp. 2189–2197.

[213] Shuhui Fan et al. "Edge-feature modeling-based topological graph neural networks for phishing scams detection on Ethereum". In: *2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS)*. IEEE. 2024, pp. 1–10.

[214] Jiahui Huang et al. "Hierarchical Network With Local-Global Awareness for Ethereum Account De-anonymization". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2025).

[215] Alex Biryukov and Sergei Tikhomirov. "Deanonymization and Linkability of Cryptocurrency Transactions Based on Network Analysis". In: *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. 2019, pp. 172–184. DOI: 10.1109/EuroSP.2019.00022.

[216] Yue Gao et al. "Practical deanonymization attack in ethereum based on p2p network analysis". In: *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. IEEE. 2021, pp. 1402–1409.

[217] Giulia Fanti and Pramod Viswanath. "Anonymity properties of the bitcoin p2p network". In: *NIPS* (2017).

[218] Shaileshh Bojja Venkatakrishnan, Giulia Fanti, and Pramod Viswanath. "Dandelion: Redesigning the bitcoin network for anonymity". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1.1 (2017), pp. 1–34.

[219] Stuart Ross and Michelle Hannan. "Money laundering regulation and risk-based decision-making". In: *Journal of Money Laundering Control* 10.1 (2007), pp. 106–115.

[220] Oluwadare Samuel Adebayo et al. "Comparative review of credit card fraud detection using machine learning and concept drift techniques". In: *International Journal of Computer Science and Mobile Computing* 12.7 (2023), pp. 24–48.

[221] Bank for International Settlements. *Project Aurora: The power of data, technology and collaboration to combat money laundering across institutions and borders*. BIS Other publication othp66. Innovation Hub proof-of-concept report on collaborative analysis and learning for anti-money laundering using privacy-enhancing technologies, machine learning and network analysis. Basel, Switzerland: Bank for International Settlements, May 2023. URL: https://www.bis.org/publ/othp66.pdf (visited on 02/02/2026).

[222] Pengcheng Xia et al. "Trade or trick? detecting and characterizing scam tokens on uniswap decentralized exchange". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 5.3 (2021), pp. 1–26.

[223] Jintao Huang et al. "Miracle or mirage? a measurement study of nft rug pulls". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 7.3 (2023), pp. 1–25.

[224] Yu Gai et al. "Blockchain large language models". In: *arXiv preprint arXiv:2304.12749* (2023).

[225] Rabia Musheer Aziz et al. "LGBM: a machine learning approach for Ethereum fraud detection". In: *International Journal of Information Technology* 14.7 (2022), pp. 3321–3331.

[226] Steven Farrugia, Joshua Ellul, and George Azzopardi. "Detection of illicit accounts over the Ethereum blockchain". In: *Expert Systems with Applications* 150 (2020), p. 113318.

[227] Georgios Palaiokrassas et al. "Leveraging Machine Learning For Multichain DeFi Fraud Detection". In: *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 2024, pp. 678–680. DOI: 10.1109/ICBC59979.2024.10634350.

[228] Jinoh Kim et al. "A machine learning approach to anomaly detection based on traffic monitoring for secure blockchain networking". In: *IEEE Transactions on Network and Service Management* 19.3 (2022), pp. 3619–3632.

[229] Jiajing Wu et al. "Who are the phishers? Phishing scam detection on ethereum via network embedding". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52.2 (2020), pp. 1156–1166.

[230] Cuneyt G Akcora et al. "BitcoinHeist: topological data analysis for ransomware prediction on the bitcoin blockchain". In: *IJCAI*. 2021, pp. 4439–4445.

[231] Vatsal Patel et al. "EvAnGCN: Evolving graph deep neural network based anomaly detection in blockchain". In: *International Conference on Advanced Data Mining and Applications*. Springer. 2022, pp. 444–456.

[232] Yuxin Qi et al. "Blockchain data mining with graph learning: A survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.2 (2023), pp. 729–748.

[233] Tomer Voronov, Danny Raz, and Ori Rottenstreich. "A framework for anomaly detection in blockchain networks with sketches". In: *IEEE/ACM Transactions on Networking* 32.1 (2023), pp. 686–698.

[234] Eric Halford et al. "Developing a scoring model for managing money laundering transactions using machine learning". In: *Journal of Money Laundering Control* 28.7 (Mar. 2025), pp. 30–49. ISSN: 1368-5201. DOI: 10.1108/JMLC-09-2024-0152. eprint: https://www.emerald.com/jmlc/article-pdf/28/7/30/10323652/jmlc-09-2024-0152en.pdf. URL: https://doi.org/10.1108/JMLC-09-2024-0152.

[235] Dylan Vassallo, Vincent Vella, and Joshua Ellul. "Application of gradient boosting algorithms for anti-money laundering in cryptocurrencies". In: *SN Computer Science* 2.3 (2021), p. 143.

[236] Fei Wan and Ping Li. "A novel money laundering prediction model based on a dynamic graph convolutional neural network and long short-term memory". In: *Symmetry* 16.3 (2024), p. 378.

[237] Joana Lorenz et al. "Machine learning methods to detect money laundering in the bitcoin blockchain in the presence of label scarcity". In: *Proceedings of the First ACM International Conference on AI in Finance*. ICAIF '20. New York, New York: Association for Computing Machinery, 2021. ISBN: 9781450375849. DOI: 10.1145/3383455.3422549. URL: https://doi.org/10.1145/3383455.3422549.

[238] Nadia Pocher et al. "Detecting anomalous cryptocurrency transactions: An AML/CFT application of machine learning-based forensics". In: *Electronic Markets* 33.1 (2023), p. 37.

[239] Cong Wu et al. "Profit or deceit? Mitigating pump and dump in DeFi via graph and contrastive learning". In: *IEEE Transactions on Information Forensics and Security* (2025).

[240] Dmitry Ermilov, Maxim Panov, and Yury Yanovich. "Automatic Bitcoin Address Clustering". In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2017, pp. 461–466. DOI: 10.1109/ICMLA.2017.0-118.

[241] Elohim Fonseca Dos Reis et al. "Identifying key players in dark web marketplaces through Bitcoin transaction networks". In: *Scientific Reports* 14.1 (2024), p. 2385.

[242] Ismail Alarab, Simant Prakoonwit, and Mohamed Ikbal Nacer. "Comparative Analysis Using Supervised Learning Methods for Anti-Money Laundering in Bitcoin". In: *Proceedings of the 2020 5th International Conference on Machine Learning Technologies*. ICMLT '20. Beijing, China: Association for Computing Machinery, 2020, pp. 11–17. ISBN: 9781450377645. DOI: 10.1145/3409073.3409078. URL: https://doi.org/10.1145/3409073.3409078.

[243] Kai Li, Shixuan Guan, and Darren Lee. "Towards understanding and characterizing the arbitrage bot scam in the wild". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 7.3 (2023), pp. 1–29.

[244] Kailong Wang et al. "Characterizing cryptocurrency-themed malicious browser extensions". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 6.3 (2022), pp. 1–31.

[245] Mehrnoosh Mirtaheri et al. "Identifying and Analyzing Cryptocurrency Manipulations in Social Media". In: *IEEE Transactions on Computational Social Systems* 8.3 (2021), pp. 607–617. DOI: 10.1109/TCSS.2021.3059286.

[246] Diogo Pacheco et al. "Uncovering coordinated networks on social media: methods and case studies". In: *Proceedings of the international AAAI conference on web and social media*. Vol. 15. 2021, pp. 455–466.

[247] Leonardo Nizzoli et al. "Charting the landscape of online cryptocurrency manipulation". In: *IEEE access* 8 (2020), pp. 113230–113245.

[248] Huy Nghiem et al. "Detecting cryptocurrency pump-and-dump frauds using market and social signals". In: *Expert Systems with Applications* 182 (2021), p. 115284.

[249] Yifan Mao et al. "Perigee: Efficient peer-to-peer network design for blockchains". In: *Proceedings of the 39th Symposium on Principles of Distributed Computing*. 2020, pp. 428–437.

[250] Weizhao Tang et al. "Strategic latency reduction in blockchain peer-to-peer networks". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 7.2 (2023), pp. 1–33.

[251] Yifan Mao and Shaileshh Bojja Venkatakrishnan. "Topiary: Fast, Scalable Publish/Subscribe for Peer-to-Peer (D) Apps". In: *arXiv preprint arXiv:2312.06800* (2023).

[252] Danila Valko and Daniel Kudenko. "Sustainable broadcasting in Blockchain Networks with Reinforcement Learning". In: *arXiv preprint arXiv:2407.15616* (2024).

[253] Seungmo Kim and Ahmed S Ibrahim. "Byzantine-fault-tolerant consensus via reinforcement learning for permissioned blockchain-empowered V2X network". In: *IEEE Transactions on Intelligent Vehicles* 8.1 (2022), pp. 172–183.

[254] Khalil Saadat, Ning Wang, and Rahim Tafazolli. "AI-enabled blockchain consensus node selection in cluster-based vehicular networks". In: *IEEE Networking Letters* 5.2 (2023), pp. 115–119.

[255] Yucong Ju et al. "Fast determination of optimal transmission rate for wireless blockchain networks: A graph convolutional neural network approach". In: *Sensors* 23.13 (2023), p. 6098.

[256] Kiana Asgari, Aida Afshar Mohammadian, and Mojtaba Tefagh. "Dyfen: Agent-based fee setting in payment channel networks". In: *arXiv preprint arXiv:2210.08197* (2022).

[257] Wuhui Chen et al. "Proactive look-ahead control of transaction flows for high-throughput payment channel network". In: *Proceedings of the 13th Symposium on Cloud Computing*. 2022, pp. 429–444.

[258] Chuangang Song, Leixiao Li, and Haoyu Gao. "Payment channel fee setting dynamic algorithm based on BiLSTM-PPO". In: *Third International Conference on Electronic Information Engineering, Big Data, and Computer Technology (EIBDCT 2024)*. Vol. 13181. SPIE. 2024, pp. 1294–1301.

[259] Heba Kadry and Yasser Gadallah. "A machine learning-based routing technique for off-chain transactions in payment channel networks". In: *2021 IEEE International Conference on Smart Internet of Things (SmartIoT)*. IEEE. 2021, pp. 66–73.

[260] Yan Qiao, Kui Wu, and Majid Khabbazian. "Non-intrusive balance tomography using reinforcement learning in the lightning network". In: *ACM Transactions on Privacy and Security* 27.1 (2024), pp. 1–32.

[261] Danila Valko and Daniel Kudenko. "Hybrid pathfinding optimization for the Lightning Network with Reinforcement Learning". In: *Engineering Applications of Artificial Intelligence* 146 (2025), p. 110225.

[262] Wuhui Chen et al. "Graph neural network-enhanced reinforcement learning for payment channel rebalancing". In: *IEEE Transactions on Mobile Computing* 23.6 (2023), pp. 7066–7083.

[263] Nikolaos Papadis and Leandros Tassiulas. "Deep reinforcement learning-based rebalancing policies for profit maximization of relay nodes in payment channel networks". In: *The International Conference on Mathematical Research for Blockchain Economy*. Springer. 2023, pp. 1–27.

[264] Pierre-Louis Aublin et al. "The Next 700 BFT Protocols". In: *ACM Transactions on Computer Systems* 32.4 (2015), 12:1–12:45.

[265] Mengting Liu et al. "Performance optimization for blockchain-enabled industrial Internet of Things (IIoT) systems: A deep reinforcement learning approach". In: *IEEE Transactions on Industrial Informatics* 15.6 (2019), pp. 3559–3570.

[266] Chenyuan Wu et al. "BFTBrain: Adaptive BFT consensus with reinforcement learning". In: *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*. 2025, pp. 1563–1583.

[267] Miguel Castro, Barbara Liskov, et al. "Practical byzantine fault tolerance". In: *OsDI*. Vol. 99. 1999. 1999, pp. 173–186.

[268] Maofan Yin et al. "HotStuff: BFT consensus with linearity and responsiveness". In: *Proceedings of the 2019 ACM symposium on principles of distributed computing*. 2019, pp. 347–356.

[269] Ramakrishna Kotla et al. "Zyzzyva: speculative byzantine fault tolerance". In: *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*. 2007, pp. 45–58.

[270] Allen Clement et al. "Making Byzantine Fault Tolerant Systems Tolerate Byzantine Faults". In: 2009, pp. 153–168.

[271] Yair Amir et al. "Prime: Byzantine Replication under Attack". In: 8.4 (2011), pp. 564–577.

[272] Maruf Monem et al. "A sustainable Bitcoin blockchain network through introducing dynamic block size adjustment using predictive analytics". In: *Future Generation Computer Systems* 153 (2024), pp. 12–26.

[273] Zhonghao Zhai, Subin Shen, and Yanqin Mao. "An explainable deep reinforcement learning algorithm for the parameter configuration and adjustment in the consortium blockchain". In: *Engineering Applications of Artificial Intelligence* 129 (2024), p. 107606.

[274] Amit Dutta et al. "ROBB: recurrent proximal policy optimization reinforcement learning for optimal block formation in bitcoin blockchain network". In: *IEEE Access* 12 (2024), pp. 31287–31311.

[275] Shir Cohen et al. "Be Aware of Your Leaders". In: *Financial Cryptography and Data Security - 26th International Conference, FC 2022, Grenada, May 2-6, 2022, Revised Selected Papers*. Ed. by Ittay Eyal and Juan A. Garay. Lecture Notes in Computer Science. Springer, 2022, pp. 279–295.

[276] Giorgos Tsimos et al. "HammerHead: Leader Reputation for Dynamic Scheduling". In: *44th IEEE International Conference on Distributed Computing Systems, ICDCS 2024, Jersey City, NJ, USA, July 23-26, 2024*. IEEE, 2024, pp. 1377–1387.

[277] Tariqul Islam et al. “MRL-PoS: A multi-agent reinforcement learning based proof of stake consensus algorithm for blockchain”. In: *2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE. 2024, pp. 0409–0413.

[278] Nour Diallo et al. “Optimized consensus with dagwise: A gnn-enhanced approach for scalable and fault-tolerant dag-based bft”. In: *2025 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE. 2025, pp. 1–5.

[279] Nour Diallo et al. “DAGWise++: GNN-Based Adaptive Optimization for DAG-BFT Consensus”. In: *2025 7th Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*. IEEE. 2025, pp. 1–9.

[280] George Danezis et al. “Narwhal and tusk: a dag-based mempool and efficient bft consensus”. In: *Proceedings of the Seventeenth European Conference on Computer Systems*. 2022, pp. 34–50.

[281] Alexander Spiegelman et al. “Bullshark: Dag bft protocols made practical”. In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2022, pp. 2705–2718.

[282] Peng Chen et al. “A novel Byzantine fault tolerance consensus for Green IoT with intelligence based on reinforcement”. In: *Journal of Information Security and Applications* 59 (2021), p. 102821.

[283] Guilain Leduc, Sylvain Kubler, and Jean-Philippe Georges. “Sabine: Self-adaptive blockchain consensus”. In: *2022 9th International Conference on Future Internet of Things and Cloud (FiCloud)*. IEEE. 2022, pp. 234–240.

[284] Guilain Leduc, Sylvain Kubler, and Jean-Philippe Georges. “DRAFTEE: A self-adaptive framework for BFT-based consensus to meet fluctuating throughput demands under security constraint”. In: *Comput. Networks* 269 (2025), p. 111396.

[285] Syamsul Rizal and Dong-Seong Kim. “Enhancing Blockchain Consensus Mechanisms: A Comprehensive Survey On Machine Learning Applications and Optimizations”. In: *Blockchain: Research and Applications* 6.4 (2025).

[286] Jianting Zhang et al. “Skychain: A deep reinforcement learning-empowered dynamic blockchain sharding system”. In: *Proceedings of the 49th International Conference on Parallel Processing*. 2020, pp. 1–11.

[287] Jusik Yun, Yunyeong Goh, and Jong-Moon Chung. “DQN-based optimization framework for secure sharded blockchain systems”. In: *IEEE Internet of Things Journal* 8.2 (2020), pp. 708–722.

[288] Zixu Zhang et al. “TbDd: A new trust-based, DRL-driven framework for blockchain sharding in IoT”. In: *Comput. Networks* 244 (2024), p. 110343.

[289] Jinlong Wang et al. “Blockchain sharding scheme based on generative AI and DRL: Applied to building internet of things”. In: *Internet of Things and Cyber-Physical Systems* 4 (2024), pp. 333–349.

[290] Pengze Li et al. “Spring: Improving the throughput of sharding blockchain via deep reinforcement learning based state placement”. In: *Proceedings of the ACM Web Conference 2024*. 2024, pp. 2836–2846.

[291] Mingxuan Song et al. “AERO: Enhancing sharding blockchain via deep reinforcement learning for account migration”. In: *Proceedings of the ACM on Web Conference 2025*. 2025, pp. 706–716.

[292] Maya Dotan and Saar Tochner. “Proofs of Useless Work - Positive and Negative Results for Wasteless Mining Systems”. In: *CoRR* abs/2007.01046 (2020).

[293] Yogev Bar-On, Ilan Komargodski, and Omri Weinstein. “Proof of Work With External Utilities”. In: *CoRR* abs/2505.21685 (2025).

[294] Ilan Komargodski, Itamar Schen, and Omri Weinstein. "Proofs of Useful Work from Arbitrary Matrix Multiplication". In: *CoRR* abs/2504.09971 (2025).

[295] Manuel Barbosa et al. "SoK: Computer-Aided Cryptography". In: *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*. IEEE, 2021, pp. 777–795.

[296] Viraj Nadkarni et al. "ZeroSwap: Data-Driven Optimal Market Making in Decentralized Finance". In: *International Conference on Financial Cryptography and Data Security*. Springer. 2024, pp. 209–227.

[297] Adam Moszczynski. "Optimizing Automated Market Makers in Frequent Batch Auctions". MA thesis. Stevens Institute of Technology, 2025.

[298] *Aave*. [Online; accessed 2025-12-09]. URL: https://aave.com/.

[299] *Morpho Docs*. [Online; accessed 2025-12-09]. Nov. 2025. URL: https://docs.morpho.org/get-started/.

[300] *Euler Finance*. [Online; accessed 2025-12-09]. URL: https://www.euler.finance/.

[301] Mahsa Bastankhah et al. "Thinking Fast and Slow: Data-Driven Adaptive DeFi Borrow-Lending Protocol". In: *6th Conference on Advances in Financial Technologies, AFT 2024*. Schloss Dagstuhl-Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing. 2024, p. 27.

[302] Mahsa Bastankhah et al. "AgileRate: Bringing Adaptivity and Robustness to DeFi Lending Markets". In: *arXiv preprint arXiv:2410.13105* (2024).

[303] Charles Bertucci et al. "Agents' behavior and interest rate model optimization in defi lending". In: *Mathematical Finance* (2025).

[304] Bastien Baude, Damien Challet, and Ioane Muni Toke. "Optimal risk-aware interest rates for decentralized lending protocols". In: *arXiv preprint arXiv:2502.19862* (2025).

[305] Tarun Chitra. "A Curationary Tale: Logarithmic Regret in DeFi Lending via Dynamic Pricing". In: *arXiv preprint arXiv:2503.18237* (2025).

[306] Yin Wu et al. "AdvSCanner: Generating Adversarial Smart Contracts to Exploit Reentrancy Vulnerabilities Using LLM and Static Analysis". In: *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*. ASE '24. Sacramento, CA, USA: Association for Computing Machinery, 2024, pp. 1019–1031. ISBN: 9798400712487. DOI: 10.1145/3691620.3695482. URL: https://doi.org/10.1145/3691620.3695482.

[307] Arthur Gervais and Liyi Zhou. "AI Agent Smart Contract Exploit Generation". In: *arXiv preprint arXiv:2507.05558* (2025).

[308] Vivi Andersson et al. "PoCo: Agentic Proof-of-Concept Exploit Generation for Smart Contracts". In: *arXiv preprint arXiv:2511.02780* (2025).

[309] Hai Jin et al. "Aroc: An Automatic Repair Framework for On-Chain Smart Contracts". In: *IEEE Transactions on Software Engineering* 48.11 (2021), pp. 4611–4629.

[310] Tai D. Nguyen, Long H. Pham, and Jun Sun. "SGUARD: Towards Fixing Vulnerable Smart Contracts Automatically". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2021, pp. 1215–1229. DOI: 10.1109/SP40001.2021.00057.

[311] Michael Rodler et al. "EVMPatch: Timely and Automated Patching of Ethereum Smart Contracts". In: *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 1289–1306. ISBN: 978-1-939133-24-3. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/rodler.

[312] Sunbeom So and Hakjoo Oh. "SmartFix: Fixing Vulnerable Smart Contracts by Accelerating Generate-and-Verify Repair using Statistical Models". In: *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ESEC/FSE 2023. San Francisco, CA, USA: Association for Computing Machinery, 2023, pp. 185–197. ISBN: 9798400703270. DOI: 10.1145/3611643.3616341. URL: https://doi.org/10.1145/3611643.3616341.

[313] Cuifeng Gao et al. "sGuard+: Machine learning guided rule-based automated vulnerability repair on smart contracts". In: *ACM Transactions on Software Engineering and Methodology* 33.5 (2024), pp. 1–55.

[314] Che Wang et al. "ContractTinker: LLM-Empowered Vulnerability Repair for Real-World Smart Contracts". In: *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*. ASE '24. Sacramento, CA, USA: Association for Computing Machinery, 2024, pp. 2350–2353. ISBN: 9798400712487. DOI: 10.1145/3691620.3695349. URL: https://doi.org/10.1145/3691620.3695349.

[315] Lyuye Zhang et al. " ACFix: Guiding LLMs With Mined Common RBAC Practices for Context-Aware Repair of Access Control Vulnerabilities in Smart Contracts ". In: *IEEE Transactions on Software Engineering* 51.09 (Sept. 2025), pp. 2512–2532. ISSN: 1939-3520. DOI: 10.1109/TSE.2025.3590108. URL: https://doi.ieeecomputersociety.org/10.1109/TSE.2025.3590108.

[316] Flashbots. *Flashbots Auction Overview*. https://docs.flashbots.net/flashbots-auction/overview. Flashbots Docs, accessed March 25, 2026. 2026.

[317] Ben Weintraub et al. "A flash (bot) in the pan: measuring maximal extractable value in private pools". In: *Proceedings of the 22nd ACM Internet Measurement Conference*. 2022, pp. 458–471.

[318] Christoffer Raun et al. "Leveraging machine learning for bidding strategies in miner extractable value (mev) auctions". In: *Cryptology ePrint Archive* (2023).

[319] Mojan Javaheripi et al. "AdaNS: Adaptive non-uniform sampling for automated design of compact DNNs". In: *IEEE Journal of Selected Topics in Signal Processing* 14.4 (2020), pp. 750–764.

[320] Bence Ladóczki. "Predictions in MEV-Boost Auctions with Machine Learning". In: *2025 7th International Conference on Blockchain Computing and Applications (BCCA)*. IEEE. 2025, pp. 518–525.

[321] *How Are Markets Disputed? - Polymarket Documentation*. URL: https://docs.polymarket.com/polymarket-learn/markets/dispute (visited on 01/29/2026).

[322] Ryoma Sato. *Even GPT-5.2 Can't Count to Five: The Case for Zero-Error Horizons in Trustworthy LLMs*. 2026. arXiv: 2601.15714 [cs.LG]. URL: https://arxiv.org/abs/2601.15714.

[323] James Ward Kalanyu Zintus-art Ben Vass. *Empirical Evidence in AI Oracle Development — Chainlink Blog*. URL: https://blog.chain.link/ai-oracles/.

[324] *UMA's AI Experiment: Can AI Agents Enhance the Optimistic Oracle?* June 3, 2025. URL: https://blog.uma.xyz/articles/experiment-can-ai-agents-enhance-uma-oracle (visited on 02/18/2026).

[325] Ryo Kamoi et al. *Evaluating LLMs at Detecting Errors in LLM Responses*. 2024. arXiv: 2404.03602 [cs.CL]. URL: https://arxiv.org/abs/2404.03602.

[326] Shangbin Feng et al. "Don't hallucinate, abstain: Identifying llm knowledge gaps via multi-llm collaboration". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2024, pp. 14664–14690.

[327] Bingbing Wen et al. “Know Your Limits: A Survey of Abstention in Large Language Models”. In: *Transactions of the Association for Computational Linguistics* 13 (June 2025), pp. 529–556. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00754. eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00754/2534960/tacl_a_00754.pdf. URL: https://doi.org/10.1162/tacl_a_00754.

[328] Rana Hassam Ahmed et al. “Integrating Large Language Models and AI Into Blockchain: A Framework for Intelligent Smart Contracts and Fraud Detection”. In: *IEEE Access* 13 (2025), pp. 181323–181335. DOI: 10.1109/ACCESS.2025.3622511.

[329] *Understanding the Intersection of Crypto and AI — Galaxy*. Feb. 14, 2024. URL: https://www.galaxy.com/insights/research/understanding-intersection-crypto-ai (visited on 02/18/2026).

[330] Zihan Zheng et al. “Agatha: Smart contract for DNN computation”. In: *arXiv preprint arXiv:2105.04919* (2021).

[331] Arasu Arun et al. “Verde: Verification via Refereed Delegation for Machine Learning Programs”. In: *arXiv preprint arXiv:2502.19405* (2025).

[332] KD Conway et al. “opml: Optimistic machine learning on blockchain”. In: *arXiv preprint arXiv:2401.17555* (2024).

[333] Michael Mirkin et al. “Arbigraph: Verifiable Turing-Complete Execution Delegation”. In: *Cryptology ePrint Archive* (2025).

[334] Jianzhu Yao et al. *Nondeterminism-Aware Optimistic Verification for Floating-Point Neural Networks*. 2025. arXiv: 2510.16028 [cs.CR]. URL: https://arxiv.org/abs/2510.16028.

[335] EigenCloud. *EigenCloud Brings Verifiable AI to Mass Market with EigenAI and EigenCompute Launches*. https://blog.eigencloud.xyz/eigencloud-brings-verifiable-ai-to-mass-market-with-eigenai-and-eigencompute-launches.

[336] Tianyi Liu, Xiang Xie, and Yupeng Zhang. “Zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy”. In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021, pp. 2968–2985.

[337] Wenjie Qu et al. “VerfCNN, Optimal Complexity zkSNARK for Convolutional Neural Networks”. In: *Cryptology ePrint Archive* (2025).

[338] Haochen Sun, Jason Li, and Hongyang Zhang. “zkllm: Zero knowledge proofs for large language models”. In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2024, pp. 4405–4419.

[339] Wenjie Qu et al. “zkGPT: An Efficient Non-interactive Zero-knowledge Proof Framework for LLM Inference”. In: *34st USENIX Security Symposium (USENIX Security 25)*. 2025.

[340] Sanjam Garg et al. “Experimenting with zero-knowledge proofs of training”. In: *Proceedings of the 2023 ACM SIGSAC conference on computer and communications security*. 2023, pp. 1880–1894.

[341] Kasra Abbaszadeh et al. “Zero-knowledge proofs of training for deep neural networks”. In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2024, pp. 4316–4330.

[342] Suppakit Waiwitlikhit et al. “Trustless audits without revealing data or models”. In: *arXiv preprint arXiv:2404.04500* (2024).

[343] Ali Shahin Shamsabadi et al. “Confidential-profitt: Confidential proof of fair training of trees”. In: *The Eleventh International Conference on Learning Representations*. 2023.

[344] Eugene Frimpong et al. "GuardML: Efficient Privacy-Preserving Machine Learning Services Through Hybrid Homomorphic Encryption". In: *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*. SAC '24. Avila, Spain: Association for Computing Machinery, 2024, pp. 953–962. ISBN: 9798400702433. DOI: 10.1145/3605098.3635983. URL: https://doi.org/10.1145/3605098.3635983.

[345] Maryam Bahrani and Naveen Durvasula. *Resonance: Transaction Fees for Heterogeneous Computation*. 2024. arXiv: 2411.11789 [cs.GT]. URL: https://arxiv.org/abs/2411.11789.

[346] Andrés Fábrega et al. "The CoinAlg Bind: Profitability-Fairness Tradeoffs in Collective Investment Algorithms". In: *arXiv preprint arXiv:2601.00523* (2026).

[347] Messari. *Daos.fun Brings Decentralized Hedge Funds to Solana*. https://messari.io/newsletter/unqualified-opinions/daos-fun-brings-decentralized-hedge-funds-to-solana. Messari Newsletter: Unqualified Opinions. Accessed March 2026. 2024.

[348] Shaw Walters et al. *Eliza: A Web3 friendly AI Agent Operating System*. 2025. DOI: 10.48550/arXiv.2501.06781. arXiv: 2501.06781 [cs.AI]. URL: https://arxiv.org/abs/2501.06781.

[349] AIXBT by Virtuals. *AIXBT: Real-Time Crypto Market Intelligence*. https://aixbt.tech. Virtuals Protocol. Accessed March 2026. 2024.

[350] SingularityDAO. *SingularityDAO: AI-Powered Quant Strategies for DeFi*. https://singularitydao.ai. Accessed: March 2026.

[351] Soldex. *Soldex: A Scalable and Decentralized AI-Powered Exchange Built on Solana*. https://soldex.ai. Accessed March 2026.

[352] Battista Biggio and Fabio Roli. "Wild patterns: Ten years after the rise of adversarial machine learning". In: *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 2018, pp. 2154–2156.

[353] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[354] Kai Greshake et al. "Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection". In: *Proceedings of the 16th ACM workshop on artificial intelligence and security*. 2023, pp. 79–90.

[355] Fábio Perez and Ian Ribeiro. "Ignore previous prompt: Attack techniques for language models". In: *arXiv preprint arXiv:2211.09527* (2022).

[356] Atharv Singh Patlan et al. "Real AI agents with fake memories: Fatal context manipulation attacks on web3 agents". In: *arXiv preprint arXiv:2503.16248* (2025).

[357] Micah Goldblum et al. "Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.2 (2022), pp. 1563–1580.

[358] Xinyun Chen et al. "Targeted backdoor attacks on deep learning systems using data poisoning". In: *arXiv preprint arXiv:1712.05526* (2017).

[359] Florian Tramèr et al. "Stealing machine learning models via prediction APIs". In: *25th USENIX security symposium (USENIX Security 16)*. 2016, pp. 601–618.

[360] Winston Wei Dou, Itay Goldstein, and Yan Ji. "AI-Powered Trading, Algorithmic Collusion, and Price Efficiency". In: *Jacobs Levy Equity Management Center for Quantitative Financial Research Paper, The Wharton School Research Paper* (2025).

[361] Megan Shearer, Gabriel Rauterberg, and Michael P Wellman. "Learning to Manipulate a Financial Benchmark". In: *Proceedings of the Fourth ACM International Conference on AI in Finance*. 2023, pp. 592–600.

[362] Xintong Wang et al. “Spoofing the Limit Order Book: A Strategic Agent-Based Analysis”. In: *Games* 12.2 (2021), p. 46.

[363] Xintong Wang and Michael P Wellman. “Market manipulation: An adversarial learning framework for detection and evasion”. In: *29th International Joint Conference on Artificial Intelligence*. 2020.

[364] Ari Juels, Ahmed Kosba, and Elaine Shi. “The Ring of Gyges: Investigating the future of criminal smart contracts”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016, pp. 283–295.

[365] Joseph Bonneau et al. “Mixcoin: Anonymity for bitcoin with accountable mixes”. In: *International conference on financial cryptography and data security*. Springer. 2014, pp. 486–504.

[366] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. “Coinshuffle: Practical decentralized coin mixing for bitcoin”. In: *European symposium on research in computer security*. Springer. 2014, pp. 345–364.

[367] Alexey Pertsev, Roman Semenov, and Roman Storm. “Tornado cash privacy solution version 1.4”. In: *Tornado cash privacy solution version* 1.6 (2019).

[368] Jaswant Pakki et al. “Everything you ever wanted to know about bitcoin mixers (but were afraid to ask)”. In: *International conference on financial cryptography and data security*. Springer. 2021, pp. 117–146.

[369] Eli Ben Sasson et al. “Zerocash: Decentralized anonymous payments from bitcoin”. In: *2014 IEEE symposium on security and privacy*. IEEE. 2014, pp. 459–474.

[370] Malte Möser et al. “An empirical analysis of traceability in the monero blockchain”. In: 2018.

[371] Winnie Xiao et al. *AI agents find $4.6M in blockchain smart contract exploits*. [Online; accessed 2026-03-27]. Dec. 2025. URL: https://red.anthropic.com/2025/smart-contracts/.

[372] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in neural information processing systems* 35 (2022), pp. 27730–27744.

[373] John Schulman et al. “Proximal policy optimization algorithms”. In: *arXiv preprint arXiv:1707.06347* (2017).

[374] Zhihong Shao et al. “Deepseekmath: Pushing the limits of mathematical reasoning in open language models”. In: *arXiv preprint arXiv:2402.03300* (2024).

[375] Rafael Rafailov et al. “Direct preference optimization: Your language model is secretly a reward model”. In: *Advances in neural information processing systems* 36 (2023), pp. 53728–53741.

[376] Shuqi Ke and Giulia Fanti. “Value-Based Pre-Training with Downstream Feedback”. In: *arXiv preprint arXiv:2601.22108* (2026).

[377] Andy K. Zhang et al. *BountyBench: Dollar Impact of AI Agent Attackers and Defenders on Real-World Cybersecurity Systems*. 2025. arXiv: 2505.15216 [cs.CR]. URL: https://arxiv.org/abs/2505.15216.

[378] Justin Wang et al. *EVMbench: Evaluating AI Agents on Smart Contract Security*. 2026. URL: https://cdn.openai.com/evmbench/evmbench.pdf.

[379] David Gray Widder and Nathan Kim. “How Big Cloud becomes Bigger: Scrutinizing Google, Microsoft, and Amazon’s investments”. In: *Microsoft, and Amazon’s investments (July 31, 2025)* (2025).

[380] Matthew S. Smith. “Meta’s Investment in AI Data Labeling Explained”. In: *IEEE Spectrum* (Aug. 2025). [Online; accessed 2026-04-07]. URL: https://spectrum.ieee.org/data-labeling-scale-ai-agents.

[381] Reuters. "Microsoft and OpenAI reach new deal valuing OpenAI at $500 billion". In: *NBC News* (Oct. 2025). [Online; accessed 2026-04-07]. URL: https://www.nbcnews.com/tech/tech-news/microsoft-openai-reach-new-deal-valuing-openai-500-billion-rcna240255.

[382] Tim De Chant. "AI companies are building huge natural gas plants to power data centers. What could go wrong?" In: *TechCrunch* (Apr. 2026). [Online; accessed 2026-04-07]. URL: https://techcrunch.com/2026/04/03/ai-companies-are-building-huge-natural-gas-plants-to-power-data-centers-what-could-go-wrong/.

[383] *AgentExchange: AI Agent Marketplace for Agentforce*. [Online; accessed 2026-04-07]. URL: https://agentexchange.salesforce.com/.

[384] *Get rewarded for the internet you don't use*. [Online; accessed 2025-11-26]. URL: https://www.grass.io/learn.

[385] *Bittensor*. [Online; accessed 2025-11-26]. URL: https://bittensor.com/.

[386] *Akash Network - Decentralized Compute Marketplace*. [Online; accessed 2025-11-26]. URL: https://akash.network/.

[387] *Theta Network*. [Online; accessed 2025-11-26]. URL: https://www.thetatoken.org.

[388] *Hivemapper — Real-Time Global Street-Level Mapping*. [Online; accessed 2025-11-26]. URL: https://hivemapper.com/.

[389] Yunming Xiao, Matteo Varvello, and Aleksandar Kuzmanovic. "Monetizing spare bandwidth: The case of distributed vpns". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 6.2 (2022), pp. 1–27.

[390] *Foldinghome – Fighting disease with a world wide distributed super computer*. [Online; accessed 2025-11-26]. URL: https://foldingathome.org/.

[391] *io.net — Decentralized GPU Ecosystem for AI Workloads - Save Up to 70%*. [Online; accessed 2025-11-26]. URL: https://io.net/.

[392] Paul Voigt and Axel von dem Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer, 2017.

[393] Siddharth Samsi et al. "From words to watts: Benchmarking the energy costs of large language model inference". In: *IEEE High Performance Extreme Computing Conference* (2023).

[394] Binhang Yuan et al. "Decentralized Training of Foundation Models in Heterogeneous Environments". In: *NeurIPS* (2022).

[395] Josh Achiam et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[396] Matthias Bastian. "GPT-4 has more than a trillion parameters - Report". In: *The Decoder* (Mar. 2023). [Online; accessed 2025-12-05]. URL: https://the-decoder.com/gpt-4-has-a-trillion-parameters/.

[397] Weiyang Wang et al. "Rail-only: A low-cost high-performance network for training llms with trillion parameters". In: *2024 IEEE Symposium on High-Performance Interconnects (HOTI)*. IEEE. 2024, pp. 1–10.

[398] Peiyuan Zhang et al. "Tinyllama: An open-source small language model". In: *arXiv preprint arXiv:2401.02385* (2024).

[399] Neelabh Sinha, Vinija Jain, and Aman Chadha. "Are small language models ready to compete with large language models for practical applications?" In: *Proceedings of the 5th Workshop on Trustworthy NLP (TrustNLP 2025)*. 2025, pp. 365–398.

[400] Zeyu Han et al. "Parameter-efficient fine-tuning for large models: A comprehensive survey". In: *arXiv preprint arXiv:2403.14608* (2024).

[401] Shervin Minaee et al. “Large language models: A survey”. In: *arXiv preprint arXiv:2402.06196* (2024).

[402] Jiangfei Duan et al. “Efficient training of large language models on distributed infrastructures: a survey”. In: *arXiv preprint arXiv:2407.20018* (2024).

[403] Jared Fernandez et al. “Hardware Scaling Trends and Diminishing Returns in Large-Scale Distributed Training”. In: *arXiv preprint arXiv:2411.13055* (2024).

[404] Karthik Mandakolathur. “Doubling all2all Performance with NVIDIA Collective Communication Library 2.12”. In: *NVIDIA Technical Blog* (Feb. 2022). [Online; accessed 2025-12-05]. URL: https://developer.nvidia.com/blog/doubling-all2all-performance-with-nvidia-collective-communication-library-2-12/.

[405] Deepak Narayanan et al. “Efficient large-scale language model training on gpu clusters using megatron-lm”. In: *Proceedings of the international conference for high performance computing, networking, storage and analysis*. 2021, pp. 1–15.

[406] Ruben Mayer and Hans-Arno Jacobsen. “Scalable Deep Learning on Distributed Infrastructures: Challenges, Techniques, and Tools”. In: *ACM Computing Surveys (CSUR)* (2020).

[407] NVIDIA. *H100 GPU*. [Online; accessed 2025-12-05]. URL: https://www.nvidia.com/en-us/data-center/h100/.

[408] AMD. *AMD Instinct™ MI300X Platform*. [Online; accessed 2025-12-05]. URL: https://www.amd.com/en/products/accelerators/instinct/mi300/platform.html.

[409] Gavin. “What Is Scale-Across? A Complete Guide to the “Third Pillar” of AI Computing”. In: *NADDOD Blog* (2025). [Online; accessed 2026-02-03]. URL: https://www.naddod.com/blog/a-complete-guide-to-scale-across-the-third-pillar-of-ai-computing.

[410] Max Ryabinin et al. “Swarm parallelism: Training large models can be surprisingly communication-efficient”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 29416–29440.

[411] Sebastian U. Stich. “Local SGD Converges Fast and Communicates Little”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net.

[412] Arthur Douillard et al. “DiLoCo: Distributed Low-Communication Training of Language Models”. In: *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ ICML 2024)*.

[413] Bowen Peng, Jeffrey Quesnelle, and Diederik P. Kingma. “DeMo: Decoupled Momentum Optimization”. In: (2026). https://openreview.net/forum?id=U9oewpa7cn.

[414] Kyle Aubrey et al. “Turbocharge LLM Training Across Long-Haul Data Center Networks with NVIDIA Nemo Framework”. In: *NVIDIA Blog* (2025).

[415] Macrocosmos et al. *LLM pretraining: The Use-Case Blockchain Has Been Waiting For?* [Online; accessed 2026-03-01]. Aug. 2024. URL: https://www.macrocosmos.ai/research/pretraining_whitepaper.pdf.

[416] Yubo Wang et al. “Mmlu-pro: A more robust and challenging multi-task language understanding benchmark”. In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 95266–95290.

[417] Karl Cobbe et al. “Training verifiers to solve math word problems”. In: *arXiv preprint arXiv:2110.14168* (2021).

[418] Johannes et al. *INTELLECT-1 Release The First Globally Trained 10B Parameter Model*. [Online; accessed 2026-03-01]. Nov. 2024. URL: https://www.primeintellect.ai/blog/intellect-1-release.

[419] Joel Lidin et al. “Incentivizing permissionless distributed learning of llms”. In: *Proceedings of the 2025 7th International Conference on Distributed Artificial Intelligence*. 2025, pp. 12–18.

[420] Felix Quinque et al. "Incentivised Orchestrated Training Architecture (IOTA): A Technical Primer for Release". In: *arXiv preprint arXiv:2507.17766* (2025).

[421] Nous Research Team. *Democratizing AI: The Psyche Network Architecture*. [Online; accessed 2026-02-03]. May 2025. URL: https://nousresearch.com/nous-psyche/.

[422] *AWS Latency Monitoring*. [Online; accessed 2025-11-27]. URL: https://www.cloudping.co/.

[423] Keivan Alizadeh et al. "Llm in a flash: Efficient large language model inference with limited memory". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2024, pp. 12562–12584.

[424] Baolin Li et al. "Llm inference serving: Survey of recent advances and opportunities". In: *2024 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE. 2024, pp. 1–8.

[425] Shashwat Jaiswal et al. "Serving models, fast and slow: optimizing heterogeneous llm inferencing workloads at scale". In: *arXiv e-prints* (2025), arXiv–2502.

[426] Feb. 2025. URL: https://openai.com/index/introducing-deep-research/.

[427] Zhengqing Yuan et al. "EfficientLLM: Efficiency in Large Language Models: Evaluation on Architecture Pretraining, Fine-Tuning, and Bit-Width Quantization". In: (2025).

[428] Theta Labs. *Introducing Theta EdgeCloud*. [Online; accessed 2025-11-26]. URL: https://assets.thetatoken.org/theta-edgecloud-whitepaper-latest.pdf.

[429] Peiyao Sheng et al. "Proof of Backhaul: Trustfree Measurement of Broadband Bandwidth". In: *NDSS*. 2024.

[430] SVR Anand et al. "Trust-free service measurement and payments for decentralized cellular networks". In: *Proceedings of the 21st ACM Workshop on Hot Topics in Networks*. 2022, pp. 68–75.

[431] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[432] Justin Smulison. "Reddit's Lawsuit Could Change How Much AI Knows About You". In: *Best Lawyers* (). [Online; accessed 2026-04-07]. URL: https://www.bestlawyers.com/article/reddit-lawsuit-could-change-how-much-ai-knows-about-you/6905.

[433] Matt O'Brien. "ChatGPT-maker OpenAI signs deal with AP to license news stories". In: *AP News* (July 2023). [Online; accessed 2026-04-07]. URL: https://apnews.com/article/openai-chatgpt-associated-press-ap-f86f84c5bcc2f3b98074b38521f5f75a.

[434] Huajian Xin et al. "Deepseek-prover: Advancing theorem proving in llms through large-scale synthetic data". In: *arXiv preprint arXiv:2405.14333* (2024).

[435] Associated Press Matt O'Brien. "Reddit sues AI company over alleged 'industrial-scale' scraping of its users' comments". In: *PBS News* (Oct. 2025). [Online; accessed 2026-04-07]. URL: https://www.pbs.org/newshour/nation/reddit-sues-ai-company-over-alleged-industrial-scale-scraping-of-its-users-comments.

[436] Cassandre Coyer. "California Reveals Brokers Selling Data to AI Firms and Overseas". In: *Bloomberg Law* (Mar. 2026). [Online; accessed 2026-04-07]. URL: https://news.bloomberglaw.com/privacy-and-data-security/california-reveals-brokers-selling-data-to-ai-firms-and-overseas.

[437] artificiallawyer. "LexisNexis Launches Data+ API For AI Training". In: *Artificial Lawyer* (Dec. 2024). [Online; accessed 2026-04-07]. URL: https://www.artificiallawyer.com/2024/12/09/lexisnexis-launches-data-api-for-ai-training/.

[438] Nicola Jones. "The AI revolution is running out of data. What can researchers do?" In: *Nature* 636.8042 (2024), pp. 290–292.

[439] LexisNexis Insights. *Amplify the Impact of Analytics: LexisNexis® Data+ and the Snowflake Data Cloud Platform*. [Online; accessed 2026-04-07]. Aug. 2023. URL: https://www.lexisnexis.com/community/insights/professional/resources/b/brochures/posts/lexisnexis-data-and-snowflake-data-cloud-platform.

[440] Datarade. *Datarade Data Marketplace*. https://datarade.ai/search/products. 2018.

[441] Amazon Web Services. *AWS Marketplace*. https://aws.amazon.com/marketplace.

[442] Preston J. Miller and Daniel M. Chin. "Using monthly data to improve quarterly model forecasts". In: *Federal Reserve Bank of Minneapolis Quarterly Review* 20.2 (1996).

[443] Adamantios Ntakaris et al. "Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods". In: *Journal of Forecasting* 37.8 (2018), pp. 852–866.

[444] RAND Europe ENISA. *Incentives and Barriers to Information Sharing*. Tech. rep. The European Network and Information Security Agency (ENISA), 2010.

[445] Andrew V Goldberg and Jason D Hartline. "Competitive auctions for multiple digital goods". In: *European Symposium on Algorithms*. Springer. 2001, pp. 416–427.

[446] Charles I Jones and Christopher Tonetti. "Nonrivalry and the Economics of Data". In: *American Economic Review* 110.9 (2020), pp. 2819–2858.

[447] Cynthia Dwork et al. "Calibrating noise to sensitivity in private data analysis". In: (2006), pp. 265–284.

[448] Ayelet Gordon-Tapiero, Katrina Ligett, and Kobbi Nissim. "On the Rival Nature of Data: Tech and Policy Implications". In: *Proceedings of the 2025 Symposium on Computer Science and Law*. 2025, pp. 17–25.

[449] Jordan Hoffmann et al. "Training compute-optimal large language models". In: *arXiv preprint arXiv:2203.15556* (2022).

[450] Amirata Ghorbani and James Zou. "Data shapley: Equitable valuation of data for machine learning". In: *International conference on machine learning*. PMLR. 2019, pp. 2242–2251.

[451] Andrew Ilyas et al. "Datamodels: Predicting Predictions from Training Data". In: *Proceedings of the 39th International Conference on Machine Learning*. 2022.

[452] Jiachen T Wang et al. "Data Shapley in One Training Run". In: *The Thirteenth International Conference on Learning Representations*. 2025.

[453] Komal Kumar et al. "Llm post-training: A deep dive into reasoning large language models". In: *arXiv preprint arXiv:2502.21321* (2025).

[454] Nikhil Kandpal and Colin Raffel. "Position: The most expensive part of an llm should be its training data". In: *arXiv preprint arXiv:2504.12427* (2025).

[455] Yiling Chen, Yiheng Shen, and Shuran Zheng. "Truthful data acquisition via peer prediction". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 18194–18204.

[456] Shuran Zheng et al. "Proper dataset valuation by pointwise mutual information". In: *arXiv preprint arXiv:2405.18253* (2024).

[457] Xinyi Xu et al. "Data distribution valuation". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 2407–2448.

[458] George A Akerlof. "The market for "lemons": Quality uncertainty and the market mechanism". In: *Uncertainty in economics*. Elsevier, 1978, pp. 235–251.

[459] Santiago Andrés Azcoitia and Nikolaos Laoutaris. "Try before You Buy: A Practical Data Purchasing Algorithm for Real-World Data Marketplaces". In: *Proc. ACM Data Economy Workshop*. 2022.

[460] Yiling Chen, Yiheng Shen, and Shuran Zheng. "Truthful data acquisition via peer prediction". In: *NeurIPS*. 2020.

[461] Snowflake. *Snowflake Marketplace*. https://www.snowflake.com/en/data-cloud/marketplace/. 2014.

[462] Chao Li et al. "A theory of pricing private data". In: *ACM Transactions on Database Systems (TODS)* 39.4 (2014), pp. 1–28.

[463] Zhihua Tian et al. "Private data valuation and fair payment in data marketplaces". In: *arXiv preprint arXiv:2210.08723* (2022).

[464] Yiwei Fu, Tianhao Wang, and Varun Chandrasekaran. "Challenges in Enabling Private Data Valuation". In: *arXiv preprint arXiv:2603.00342* (2026).

[465] Yuchen Hu, Martin J Wainwright, and Stephen Bates. "Buying Data of Unknown Quality: Fisher Information Procurement Auctions". In: *arXiv preprint arXiv:2604.08821* (2026).

[466] Raef Bassily et al. "Data sharing with endogenous choices over differential privacy levels". In: *arXiv preprint arXiv:2602.09357* (2026).

[467] Ali Aouad, Ömer Sarıtaç, and Chiwei Yan. "Centralized versus decentralized pricing controls for dynamic matching platforms". In: *Available at SSRN 4453799* (2023).

[468] Apostolos Filippas, Srikanth Jagabathula, and Arun Sundararajan. "The limits of centralized pricing in online marketplaces and the value of user control". In: *Management Science* 69.12 (2023), pp. 7202–7216.

[469] Max von Thun. *Monopoly Power Is the Elephant in the Room in the AI Debate*. [Online; accessed 2026-05-11]. Oct. 2023. URL: https://www.techpolicy.press/monopoly-power-is-the-elephant-in-the-room-in-the-ai-debate/.

[470] *United States of America vs. Google LLC*. [Online; accessed 2026-05-11]. 2025. URL: https://storage.courtlistener.com/recap/gov.uscourts.vaed.533508/gov.uscourts.vaed.533508.1410.0.pdf.

[471] Megan Kirkwood. *Understanding the Apple and Meta Non-Compliance Decisions Under the Digital Markets Act*. [Online; accessed 2026-05-11]. Apr. 2025. URL: https://www.techpolicy.press/understanding-the-apple-and-meta-noncompliance-decisions-under-the-digital-markets-act/.

[472] Jean Tirole. "Competition and the industrial challenge for the digital age". In: *Annual Review of Economics* 15.1 (2023), pp. 573–605.

[473] Michael I Jordan. "A collectivist, economic perspective on AI". In: *arXiv preprint arXiv:2507.06268* (2025).

[474] Gur Huberman, Jacob D Leshno, and Ciamac Moallemi. "Monopoly without a monopolist: An economic analysis of the bitcoin payment system". In: *The Review of Economic Studies* 88.6 (2021), pp. 3011–3040.

[475] Amit Levy, S. Matthew Weinberg, and Chenghan Zhou. "Analyzing the Economic Impact of Decentralization on Users". In: *Proceedings of the 17th Annual Innovations in Theoretical Computer Science Conference (ITCS)*. 2026.

[476] Mike Masnick. *Protocols, Not Platforms: A Technological Approach to Free Speech*. Aug. 21, 2019. URL: https://knightcolumbia.org/content/protocols-not-platforms-a-technological-approach-to-free-speech (visited on 05/28/2026).

[477] Maryam Bahrani and Naveen Durvasula. "Resonance: Transaction Fees for Heterogeneous Computation". In: *arXiv preprint arXiv:2411.11789* (2024).

[478] Michael Sockin and Wei Xiong. "Decentralization through tokenization". In: *The Journal of Finance* 78.1 (2023), pp. 247–299.

[479] Marco Reuter. *Platform Precommitment via Decentralization*. International Monetary Fund, 2024.

[480] *GRASS: Get rewarded for the internet you don't use*. [Online; accessed 2026-05-12]. URL: https://www.grass.io/.

[481] Abdulrahman Alhaidari, Balaji Palanisamy, and Prashant Krishnamurthy. "On-Chain Decentralized Learning and Cost-Effective Inference for DeFi Attack Mitigation". In: *7th Conference on Advances in Financial Technologies (AFT 2025)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. 2025, pp. 35–1.

[482] IOTA Foundation. *IOTA Data Marketplace*. [Online; accessed 2026-05-13]. Nov. 2017. URL: https://blog.iota.org/iota-data-marketplace-cb6be463ac7f/.

[483] ORAICHAIN. *Investing in Crypto: Unlocking the Power of AI Layer 1 Blockchain Ecosystems*. [Online; accessed 2026-05-12]. URL: https://orai.io/ai-layer-1.

[484] *Sentient Foundation — Open-Source AGI*. [Online; accessed 2026-02-14]. URL: https://www.sentient.foundation/.

[485] SingularityNet. *About Us*. [Online; accessed 2026-05-12]. URL: https://singularitynet.io/aboutus/.

[486] Story. *Why We're Incubating Poseidon*. [Online; accessed 2026-05-13]. July 2025. URL: https://www.story.foundation/blog/why-were-incubating-poseidon.

[487] Anna Kauzlaskas. *Vana: An Open Protocol for Data Sovereignty*. [Online; accessed 2026-05-12]. 2024. URL: https://cdn.prod.website-files.com/662488d48afbf9b21e25c7d2/674f6289079193fb53f7caa1_Vana_whitepaper_Dec24_updated.pdf.

[488] Nick Hynes et al. "A demonstration of sterling: a privacy-preserving data marketplace". In: *Proceedings of the VLDB Endowment* 11.12 (2018), pp. 2086–2089.

[489] Oasis Protocol Foundation. *The Oasis Blockchain Platform*. Tech. rep. Oasis Protocol Foundation, 2020. URL: https://oasisprotocol.org/whitepaper.

[490] Trent McConaghy. "Ocean protocol: tools for the web3 data economy". In: *Handbook on Blockchain*. Springer, 2022, pp. 505–539.

[491] Matthew Dahl et al. "Large legal fictions: Profiling legal hallucinations in large language models". In: *Journal of Legal Analysis* 16.1 (2024), pp. 64–93.

[492] Emilio Calvano et al. "Artificial intelligence, algorithmic pricing, and collusion". In: *American Economic Review* 110.10 (2020), pp. 3267–3297.

[493] Adam Back et al. "Hashcash-a denial of service counter-measure". In: (2002).

[494] Awid Vaziry, Sandro Rodriguez Garzon, and Axel Küpper. "Towards Multi-Agent Economies: Enhancing the A2A Protocol with Ledger-Anchored Identities and x402 Micropayments for AI Agents". In: *arXiv preprint arXiv:2507.19550* (2025).

[495] Jay Yu and Ryan Barney. *HTTP 402's Modern Makeover*. https://panteracapital.com/http-402s-modern-makeover/. Sept. 2025.

[496] Will Allen and Simon Newton. *Introducing Pay Per Crawl: Enabling Content Owners to Charge AI Crawlers for Access*. Cloudflare Blog. Accessed: 2026-04-17. July 2025. URL: https://blog.cloudflare.com/introducing-pay-per-crawl/.

[497] Erik Reppel et al. *x402: An open standard for internet-native payments*. [Online; accessed 2026-05-15]. May 2025. URL: https://www.x402.org/x402-whitepaper.pdf.

[498] Circle Internet Financial. *USDC: Powering Global Finance*. https://www.circle.com/usdc. Issued by Circle; reserves held in the Circle Reserve Fund (USDXX) custodied at BNY Mellon and managed by BlackRock; redeemable 1:1 for US dollars. 2024.

[499] Tether Operations. *Tether (USDT): Transparency and Reserves*. https://tether.to/en/. Issued by Tether Operations, S.A. de C.V.; pegged 1:1 to USD. 2024.

[500] Sky Ecosystem. *Sky*. https://sky.money/. Issues USDS and sUSDS; the legacy DAI stablecoin (originally from MakerDAO) is also part of the ecosystem. 2024.

[501] Chainalysis. *The 2025 Crypto Crime Report*. Tech. rep. Report explicitly notes that "stablecoin issuers often freeze funds if they are made aware of their use by illicit actors," with Tether named as freezing addresses linked to scams, terrorist financing, and sanctions evasion. Chainalysis, Feb. 2025. URL: https://www.chainalysis.com/wp-content/uploads/2025/03/the-2025-crypto-crime-report-release.pdf.

[502] Google. *Agent2Agent (A2A) Protocol*. https://github.com/google/A2A. 2025.

[503] Awid Vaziry, Sandro Rodriguez Garzon, and Axel Küpper. "Towards Multi-Agent Economies: Enhancing the A2A Protocol with Ledger-Anchored Identities and x402 Micropayments for AI Agents". In: *arXiv preprint arXiv:2507.19550* (2025).

[504] Botao Amber Hu and Helena Rong. "Inter-Agent Trust Models: A Comparative Study of Brief, Claim, Proof, Stake, Reputation and Constraint in Agentic Web Protocol Design—A2A, AP2, ERC-8004, and Beyond". In: *arXiv preprint arXiv:2511.03434* (2025).

[505] Ramesh Raskar et al. "Beyond DNS: Unlocking the Internet of AI Agents via the NANDA Index and Verified AgentFacts". In: *arXiv preprint arXiv:2507.14263* (2025). Project NANDA.

[506] Fred B Schneider. "Enforceable security policies". In: *ACM Transactions on Information and System Security (TISSEC)* 3.1 (2000), pp. 30–50.

[507] Thomas Kwa et al. *Measuring AI Ability to Complete Long Software Tasks*. 2026. arXiv: 2503.14499 [cs.AI]. URL: https://arxiv.org/abs/2503.14499.

[508] Xudong Pan et al. *Frontier AI systems have surpassed the self-replicating red line*. 2024. arXiv: 2412.12140 [cs.CL]. URL: https://arxiv.org/abs/2412.12140.

[509] Sid Black et al. *RepliBench: Evaluating the Autonomous Replication Capabilities of Language Model Agents*. 2025. arXiv: 2504.18565 [cs.CR]. URL: https://arxiv.org/abs/2504.18565.

[510] Nicholas Carlini et al. *Assessing Claude Mythos Preview's Cybersecurity Capabilities*. https://red.anthropic.com/2026/mythos-preview/. Anthropic Red Team Blog. Apr. 2026.

[511] Aengus Lynch et al. "Agentic misalignment: how LLMs could be insider threats". In: *arXiv preprint arXiv:2510.05179* (2025).

[512] Alex Turner and Prasad Tadepalli. "Parametrically retargetable decision-makers tend to seek power". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 31391–31401.

[513] Alex Turner et al. "Optimal Policies Tend To Seek Power". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 23063–23074. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/c26820b8a4c1b3c2aa868d6d57e14a79-Paper.pdf.

[514] Yufei He et al. "Evaluating the Paperclip Maximizer: Are RL-Based Language Models More Likely to Pursue Instrumental Goals?" In: *arXiv preprint arXiv:2502.12206* (2025).

[515] Tom Barbereau and Balázs Bodó. "Beyond Financial Regulation of Crypto-Asset Wallet Software: In Search of Secondary Liability". In: *Computer Law & Security Review* 49 (2023), p. 105829.

[516] Financial Crimes Enforcement Network. *Application of FinCEN's Regulations to Certain Business Models Involving Convertible Virtual Currencies*. Guidance FIN-2019-G001, May 9, 2019. 2019.

[517] Amanda Tuminelli, Lizandro Pieper, and Peter Van Valkenburgh. *Expert Report of Coin Center and DeFi Education Fund*. https://www.defieducationfund.org/. Filed in support of Alexey Pertsev appeal, Tornado Cash prosecution (Netherlands). May 2025.

[518] Primavera De Filippi, Morshed Mannan, and Wessel Reijers. "Blockchain Technology and the Rule of Code: Regulation via Governance". In: *George Washington Law Review* 92 (2024), p. 1229.

[519] Elisabeth M.S. Frommelt. "Liability Challenges in the Blockchain Ecosystem". In: *UC Davis Business Law Journal* (2020). Analysis of liability gaps in blockchain under the Liechtenstein Civil Code and Blockchain Act; proposes legal personality, permissioned layers, and programmed arbitration as remedies. Volume / pages — verify against UC Davis Business Law Journal table of contents.

[520] Jiaming Ji et al. "Beavertails: Towards improved safety alignment of llm via a human-preference dataset". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 24678–24704.

[521] Kaifeng Lyu et al. "Keeping llms aligned after fine-tuning: The crucial role of prompt templates". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 118603–118631.

[522] Hoagy Cunningham et al. "Constitutional Classifiers++: Efficient Production-Grade Defenses against Universal Jailbreaks". In: *arXiv preprint arXiv:2601.04603* (2026).

[523] János Kramár et al. "Building Production-Ready Probes For Gemini". In: *arXiv preprint arXiv:2601.11516* (2026).

[524] Hanna Foerster et al. "Camels can use computers too: System-level security for computer use agents". In: *arXiv preprint arXiv:2601.09923* (2026).

[525] Andy Zou et al. "Universal and transferable adversarial attacks on aligned language models". In: *arXiv preprint arXiv:2307.15043* (2023).

[526] Araz Taeihagh. "Governance of generative AI". In: *Policy and society* 44.1 (2025), pp. 1–22.

[527] Allan Dafoe. "AI governance: a research agenda". In: *Governance of AI Program, Future of Humanity Institute, University of Oxford: Oxford, UK* 1442 (2018), p. 1443.

[528] Huw Roberts et al. "Global AI governance: barriers and pathways forward". In: *International Affairs* 100.3 (2024), pp. 1275–1286.

[529] Qinxu Ding et al. "A Survey on Decentralized Autonomous Organizations (DAOs) and Their Governance". In: *World Scientific Annual Review of Fintech* 1 (2023). DOI: 10.1142/S281100482350001X.

[530] Tezos Docs. *Governance and self-amendment*. https://docs.tezos.com/architecture/governance. 2025.

[531] *RetroPGF*. https://www.retropgf.com/.

[532] Rainer Feichtinger et al. "SoK: Attacks on DAOs". In: *6th Conference on Advances in Financial Technologies (AFT 2024)*. Vol. 316. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024, 28:1–28:27. DOI: 10.4230/LIPIcs.AFT.2024.28.

[533] Stefan Studer et al. "Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology". In: *Machine learning and knowledge extraction* 3.2 (2021), pp. 392–413.

[534] Saleema Amershi et al. "Software engineering for machine learning: A case study". In: *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE. 2019, pp. 291–300.

[535] Iason Gabriel. "Artificial intelligence, values, and alignment". In: *Minds and machines* 30.3 (2020), pp. 411–437.

[536] Jiaming Ji et al. "Ai alignment: A comprehensive survey". In: *arXiv preprint arXiv:2310.19852* (2023).

[537] Yuntao Bai et al. "Constitutional AI: Harmlessness from AI Feedback". In: *arXiv preprint arXiv:2212.08073* (2022).

[538] Anthropic. *Claude's Constitution*. https://www.anthropic.com/constitution. 2026.

[539] Saffron Huang et al. "Collective constitutional ai: Aligning a language model with public input". In: *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*. 2024, pp. 1395–1417.

[540] Michiel Bakker et al. "Fine-tuning language models to find agreement among humans with diverse preferences". In: *Advances in neural information processing systems* 35 (2022), pp. 38176–38189.

[541] OpenAI. *Democratic inputs to AI grant program: lessons learned and implementation plans*. Accessed: 2024-12-19. 2024. URL: https://openai.com/index/democratic-inputs-to-ai-grant-program-update/.

[542] "Amoral Drift in AI Corporate Governance". In: *Harvard Law Review* 138.6 (Apr. 2025), pp. 1633–1656.

[543] Robin Fritsch, Marino Müller, and Roger Wattenhofer. "Analyzing Voting Power in Decentralized Governance: Who Controls DAOs?" In: *Blockchain: Research and Applications* 5.3 (2024), p. 100208. DOI: 10.1016/j.bcra.2024.100208.

[544] Vitalik Buterin, Zoë Hitzig, and Eric Glen Weyl. "A Flexible Design for Funding Public Goods". In: *Management Science* 65.11 (2019), pp. 5171–5187. DOI: 10.1287/mnsc.2019.3337.

[545] Michael Zargham, Krzysztof Paruch, and Jamsheed Shorish. "Economic Games as Estimators". In: *Mathematical Research for Blockchain Economy*. Springer, 2020, pp. 145–169. DOI: 10.1007/978-3-030-53356-4_8.

[546] Shutter Network. *Coming Soon to DAOs: Permanent Shielded Voting via Homomorphic Encryption*. https://blog.shutter.network/coming-soon-to-daos-permanent-shielded-voting-via-homomorphic-encryption/. 2025.

[547] Samuel Breckenridge et al. "B-Privacy: Defining and Enforcing Privacy in Weighted Voting". In: *arXiv preprint arXiv:2509.17871* (2025).

[548] Snapshot Labs. *Snapshot: Voting Platform for DAOs*. https://snapshot.org. Accessed: 2024-12-03. 2024.

[549] Polkadot Docs. *On-Chain Governance*. https://docs.polkadot.com/reference/governance/. 2024.

[550] Vitalik Buterin. *Notes on Blockchain Governance*. https://vitalik.eth.limo/general/2017/12/17/voting.html. 2017.

[551] Vlad Zamfir. *Against on-chain governance*. https://medium.com/@Vlad_Zamfir/against-on-chain-governance-a4ceacd040ca. 2017.

[552] Bittensor. *Governance Overview*. Accessed: 2024-12-19. 2024. URL: https://docs.learnbittensor.org/governance.

[553] Giza. *Giza: Verifiable AI Agents*. Decentralized machine learning inference with ZK proofs. Accessed: 2024-12-19. 2024. URL: https://www.gizatech.xyz/.

[554] Tanusree Sharma et al. "Inclusive. Ai: engaging underserved populations in democratic decision-making on ai". In: *SocialComputing. Web. Illinois. edu https://socialcomputing. web. illinois. edu/images/Report-InclusiveAI. pdf* (2024).

[555] Maciel M Queiroz, Renato Telles, and Silvia H Bonilla. "Blockchain and supply chain management integration: a systematic review of the literature". In: *Supply chain management: An international journal* 25.2 (2020), pp. 241–254.

[556] Nir Kshetri. "Blockchain's roles in meeting key supply chain management objectives". In: *International Journal of information management* 39 (2018), pp. 80–89.

[557] Hengrui Jia et al. "Proof-of-learning: Definitions and practice". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2021, pp. 1039–1056.

[558] Miles Brundage et al. "Toward trustworthy AI development: mechanisms for supporting verifiable claims". In: *arXiv preprint arXiv:2004.07213* (2020).

[559] Arasu Arun et al. "Verde: Verification via refereed delegation for machine learning programs". In: *arXiv preprint arXiv:2502.19405* (2025).

[560] Zhizhi Peng et al. "A survey of zero-knowledge proof based verifiable machine learning". In: *arXiv preprint arXiv:2502.18535* (2025).

[561] Taylor Sorensen et al. "A roadmap to pluralistic alignment". In: *arXiv preprint arXiv:2402.05070* (2024).

[562] John R Douceur. "The sybil attack". In: *International workshop on peer-to-peer systems*. Springer. 2002, pp. 251–260.

[563] Bryan Ford. "Identity and personhood in digital democracy: Evaluating inclusion, equality, security, and privacy in pseudonym parties and other proofs of personhood". In: *arXiv preprint arXiv:2011.02412* (2020).

[564] Alex Blania and Sam Altman. *Introducing Worldcoin*. `https://world.org/cofounder-letter`.

[565] Kali Hays. "Tinder and Zoom Offer 'Proof of Humanity' Eye-Scans to Combat AI". In: *BBC News* (Apr. 2026). Accessed Apr. 2026. URL: `https://www.bbc.com/news/articles/cp9vppem4evo`.

[566] Carla L. Reyes. "(Un)Corporate Crypto-Governance". In: *Fordham Law Review* 88 (2020), pp. 1875–1920. URL: `https://ir.lawnet.fordham.edu/flr/vol88/iss5/13/`.

[567] Jonathan Rose. "The meaning of corruption: Testing the coherence and adequacy of corruption definitions". In: *Public Integrity* 20.3 (2018), pp. 220–233.

[568] Joseph Stiglitz et al. "Transparency in government". In: *The right to tell: The role of mass media in economic development* 25070 (2002), pp. 27–44.

[569] Federal Trade Commission. *Federal Trade Commission Launches Inquiry into Tech Censorship*. Accessed: 2025-04-05. 2025. URL: `https://www.ftc.gov/news-events/news/press-releases/2025/02/federal-trade-commission-launches-inquiry-tech-censorship`.

[570] FindLaw Editorial Team. *Social Media Censorship and the Law*. Accessed: 2025-04-05. 2023. URL: `https://www.findlaw.com/civilrights/enforcing-your-civil-rights/social-media-censorship-and-the-law.html`.

[571] Geoffrey A. Manne. *Net Neutrality and the Paradox of Private Censorship*. `https://truthonthemarket.com/2024/04/25/net-neutrality-and-the-paradox-of-private-censorship/`. Accessed: 2025-04-05. 2024.

[572] Anonymous. *Algorithmic Arbitrariness in Content Moderation*. `https://arxiv.org/abs/2402.16979`. Accessed: 2025-04-05. 2024.

[573] David Wong and Luciano Floridi. "Meta's oversight board: A review and critical assessment". In: *Minds and Machines* 33.2 (2023), pp. 261–284.

[574] Peng Hwa Ang and Sherly Haristya. "The Governance, Legitimacy and Efficacy of Facebook's Oversight Board: A Model for Global Tech Platforms?" In: *Emerging Media* 2.2 (2024), pp. 169–180.

[575] Eddie L Ungless, Nina Markl, and Björn Ross. "Experiences of censorship on TikTok across marginalised identities". In: *arXiv preprint arXiv:2407.14164* (2024).

[576] Kokil Jaidka, Subhayan Mukerjee, and Yphtach Lelkes. "Silenced on social media: the gatekeeping functions of shadowbans in the American Twitterverse". In: *Journal of Communication* 73.2 (2023), pp. 163–178.

[577] Zeeshan Rahman. "Enforcing Copyright on Online Streaming Platforms: Challenges Faced by Rights Holders in the Digital Era". In: *International Journal for Multidisciplinary Research (IJFMR)* 5 (2023), pp. 1–14.

[578] Alex Hoagland, Olivia Yu, and Michal Hornỳ. "Social determinants of health and insurance claim denials for preventive care". In: *JAMA Network Open* 7.9 (2024), e2433316–e2433316.

[579] Massimo Bartoletti and Livio Pompianu. "An empirical analysis of smart contracts: platforms, applications, and design patterns". In: *Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers 21*. Springer. 2017, pp. 494–509.

[580] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[581] Josh Achiam et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[582] Hugo Touvron et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).

[583] Harry Kalodner et al. "Arbitrum: Scalable smart contracts". In: *USENIX Security Symposium*. 2018.

[584] Optimism Team. *Optimistic Rollups: Scalable Execution with Economic Security*. `https://optimism.io/`. 2019.

[585] Hadi Jooybar et al. "GPUDet: a deterministic GPU architecture". In: *Proceedings of the eighteenth international conference on Architectural support for programming languages and operating systems*. 2013, pp. 1–12.

[586] Alex Schlögl, Nora Hofer, and Rainer Böhme. "Causes and effects of unanticipated numerical deviations in neural network inference frameworks". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 56095–56107.

[587] EZKL Contributors. *EZKL: Easy Zero-Knowledge for ONNX Models*. `https://github.com/zkonduit/ezkl`. 2023.

[588] Wyatt Benno et al. "Jolt Atlas: Verifiable Inference via Lookup Arguments in Zero Knowledge". In: *arXiv preprint arXiv:2602.17452* (2026).

[589] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[590] Lagrange Labs. *DeepProve: A zkML Framework for Verifiable Neural Network Inference*. `https://github.com/Lagrange-Labs/deep-prove`. Open-source library. 2025.

[591] Tao Lu et al. "An efficient and extensible zero-knowledge proof framework for neural networks". In: *Cryptology ePrint Archive* (2024).

[592] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[593] Alec Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[594] Meng Hao et al. "Scalable zero-knowledge proofs for non-linear functions in machine learning". In: *33rd USENIX Security Symposium (USENIX Security 24)*. 2024, pp. 3819–3836.

[595] Yanpei Guo et al. "Architecture-private Zero-knowledge Proof of Neural Networks". In: *Cryptology ePrint Archive* (2025).

[596] Pranay Anchuri et al. "Towards Verifiable AI with Lightweight Cryptographic Proofs of Inference". In: 2026. URL: `https://eprint.iacr.org/2026/541`.

[597] Max Klabunde et al. "Similarity of Neural Network Models: A Survey of Functional and Representational Measures". In: *ACM Comput. Surv.* 57.9 (May 2025). ISSN: 0360-0300. DOI: 10.1145/3728458. URL: https://doi.org/10.1145/3728458.

[598] Nikolaus Kriegeskorte, Marieke Mur, and Peter Bandettini. "Representational Similarity Analysis—Connecting the Branches of Systems Neuroscience". In: *Frontiers in Systems Neuroscience* 2 (2008), p. 4.

[599] Simon Kornblith et al. "Similarity of Neural Network Representations Revisited". In: *International Conference on Machine Learning (ICML)*. 2019, pp. 3519–3529.

[600] Xiangyu Qi et al. "Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To!" In: *arXiv preprint arXiv:2310.03693* (2023).

[601] Bahrad A Sokhansanj. "Uncensored AI in the Wild: Tracking Publicly Available and Locally Deployable LLMs". In: *Future Internet* 17.10 (2025), p. 477.

[602] Yifan Sun et al. "SVIP: Towards Verifiable Inference of Open-Source Large Language Models". In: *arXiv preprint* (2025).

[603] OpenClaw Team. *OpenClaw: Open Source AI Assistant*. https://openclaw.org. Accessed: 2025-02-02. 2025.

[604] OpenRouter. *OpenRouter: A Unified API for LLMs*. https://openrouter.ai. Accessed: 2026-06-01.

[605] Edoardo Debenedetti et al. "Agentdojo: A dynamic environment to evaluate attacks and defenses for llm agents". In: *arXiv e-prints* (2024), arXiv–2406.

[606] Atharv Singh Patlan et al. "Context manipulation attacks: Web agents are susceptible to corrupted memory". In: *arXiv preprint arXiv:2506.17318* (2025).

[607] Milad Nasr et al. "The attacker moves second: Stronger adaptive attacks bypass defenses against LLM jailbreaks and prompt injections". In: *arXiv preprint arXiv:2510.09023* (2025).

[608] Sinisa Matetic et al. "{ROTE}: Rollback protection for trusted execution". In: *26th USENIX Security Symposium (USENIX Security 17)*. 2017, pp. 1289–1306.

[609] Jianyu Niu et al. "Narrator: Secure and practical state continuity for trusted execution in the cloud". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2022, pp. 2385–2399.

[610] Annika Wilde et al. "The Forking Way: When TEEs Meet Consensus". In: *arXiv preprint arXiv:2412.00706* (2024).

[611] Raoul Strackx and Frank Piessens. "Ariadne: A minimal approach to state continuity". In: *25th USENIX Security Symposium (USENIX Security 16)*. 2016, pp. 875–892.

[612] Samira Briongos et al. "No forking way: Detecting cloning attacks on Intel SGX applications". In: *Proceedings of the 39th Annual Computer Security Applications Conference*. 2023, pp. 744–758.

[613] Zhongshu Gu et al. "NVIDIA GPU Confidential Computing Demystified". In: *arXiv preprint arXiv:2507.02770* (2025).

[614] NVIDIA. *Confidential Computing on NVIDIA H100 GPUs for Secure and Trustworthy AI*. Tech. rep. Accessed: June 2, 2026. Aug. 2023. URL: https://developer.nvidia.com/blog/confidential-computing-on-h100-gpus-for-secure-and-trustworthy-ai/.

[615] Tobin South et al. "Identity Management for Agentic AI: The new frontier of authorization, authentication, and security for an AI agent world". In: *arXiv preprint arXiv:2510.25819* (2025).

[616] Yedidel Louck, Ariel Stulman, and Amit Dvir. "Proposal for Improving Google A2A Protocol: Safeguarding Sensitive Data in Multi-Agent Systems". In: *arXiv preprint arXiv:2505.12490* (2025).

[617] Subramanya Nagabhushanaradhya. "OpenID Connect for Agents (OIDC-A) 1.0: A Standard Extension for LLM-Based Agent Identity and Authorization". In: *arXiv preprint arXiv:2509.25974* (2025).

[618] Brendan McMahan et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[619] Andrew Hard et al. "Federated learning for mobile keyboard prediction". In: 2018.

[620] Qinbin Li et al. "A survey on federated learning systems: Vision, hype and reality for data privacy and protection". In: *IEEE Transactions on Knowledge and Data Engineering* 35.4 (2021), pp. 3347–3366.

[621] Jie Wen et al. "A survey on federated learning: challenges and applications". In: *International journal of machine learning and cybernetics* 14.2 (2023), pp. 513–535.

[622] Omair Rashed Abdulwareth Almanifi et al. "Communication and computation efficiency in federated learning: A survey". In: *Internet of Things* 22 (2023), p. 100742.

[623] Zihao Zhao et al. "Towards efficient communications in federated learning: A contemporary survey". In: *Journal of the Franklin Institute* 360.12 (2023), pp. 8669–8703.

[624] Minghong Fang et al. "Local model poisoning attacks to {Byzantine-Robust} federated learning". In: *29th USENIX security symposium (USENIX Security 20)*. 2020, pp. 1605–1622.

[625] Arjun Nitin Bhagoji et al. "Analyzing federated learning through an adversarial lens". In: *International conference on machine learning*. PMLR. 2019, pp. 634–643.

[626] Eugene Bagdasaryan et al. "How to backdoor federated learning". In: *International conference on artificial intelligence and statistics*. PMLR. 2020, pp. 2938–2948.

[627] Jonghyun Lee et al. "Characterization of GPU TEE Overheads in Distributed Data Parallel ML Training". In: *arXiv preprint arXiv:2501.11771* (2025).

[628] All of Us Research Program Investigators. "The "All of Us" research program". In: *New England Journal of Medicine* 381.7 (2019), pp. 668–676.

[629] Melissa A Haendel et al. "The National COVID Cohort Collaborative (N3C): rationale, design, infrastructure, and deployment". In: *Journal of the American Medical Informatics Association* 28.3 (2021), pp. 427–443.

[630] Pablo Villalobos et al. "Position: Will we run out of data? Limits of LLM scaling based on human-generated data". In: *Forty-first International Conference on Machine Learning*. 2024.

[631] Pablo Villalobos et al. *Will We Run Out of Data? Limits of LLM Scaling Based on Human-Generated Data*. Epoch AI Blog. Updated analysis, based on arXiv preprint arXiv:2211.04325. June 2024. URL: https://epoch.ai/blog/will-we-run-out-of-data-limits-of-llm-scaling-based-on-human-generated-data.

[632] Kyle Wiggers. *Elon Musk agrees that we've exhausted AI training data*. Jan. 2025. URL: https://techcrunch.com/2025/01/08/elon-musk-agrees-that-weve-exhausted-ai-training-data/.

[633] Grant Gross. *Synthetic data takes aim at AI training challenges*. Gartner projects that by 2028, 80% of data used by AIs will be synthetic, up from 20% in 2024. Feb. 2025. URL: https://www.cio.com/article/3827383/synthetic-data-takes-aim-at-ai-training-challenges.html.

[634] Ilia Shumailov et al. "AI models collapse when trained on recursively generated data". In: *Nature* 631.8022 (2024), pp. 755–759.

[635] Sina Alemohammad et al. "Self-consuming generative models go mad". In: International Conference on Learning Representations (ICLR). 2024.

[636] Rahul Rao. "AI-Generated Data Can Poison Future AI Models". In: *Scientific American* (23 July 2023). URL: https://www.scientificamerican.com/article/ai-generated-data-can-poison-future-ai-models/.

[637] Michael K Bergman. "White paper: the deep web: surfacing hidden value". In: *Journal of electronic publishing* 7.1 (2001).

[638] Somesh Rai, Kunwar Singh, and Akhilesh Kumar Varma. "A Bibliometric Analysis of Deep Web Research during 1997-2019." In: *DESIDOC Journal of Library & Information Technology* 40.2 (2020).

[639] Kled AI. *Kled AI Surges to a $100 Million Valuation as It Builds the World's First Consumer Data Marketplace*. ACCESS Newswire. Dec. 2025. URL: https://www.accessnewswire.com/newsroom/en/business-and-professional-services/kled-ai-surges-to-a-100-million-valuation-as-it-builds-the-world-1117096.

[640] Martin Abadi et al. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318.

[641] Alberto Blanco-Justicia et al. "A critical review on the use (and misuse) of differential privacy in machine learning". In: *ACM Computing Surveys* 55.8 (2022), pp. 1–16.

[642] Lea Demelius, Roman Kern, and Andreas Trügler. "Recent advances of differential privacy in centralized deep learning: A systematic survey". In: *ACM Computing Surveys* 57.6 (2025), pp. 1–28.

[643] Cynthia Dwork and Aaron Roth. "The algorithmic foundations of differential privacy". In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[644] Ke Pan et al. "Differential privacy in deep learning: A literature survey". In: *Neurocomputing* 589 (2024), p. 127663.

[645] Kang Wei et al. "Federated learning with differential privacy: Algorithms and performance analysis". In: *IEEE transactions on information forensics and security* 15 (2020), pp. 3454–3469.

[646] Yanling Wang et al. "Differential privacy in deep learning: Privacy and beyond". In: *Future Generation Computer Systems* 148 (2023), pp. 408–424.

[647] Foteini Baldimtsi et al. "zklogin: Privacy-preserving blockchain authentication with existing credentials". In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2024, pp. 3182–3196.

[648] Florian Tramer et al. "On adaptive attacks to adversarial example defenses". In: *Advances in neural information processing systems* 33 (2020), pp. 1633–1645.

[649] Nicholas Carlini et al. "On evaluating adversarial robustness". In: *arXiv preprint arXiv:1902.06705* (2019).

[650] Nicolas Papernot et al. "Distillation as a defense to adversarial perturbations against deep neural networks". In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 582–597.

[651] Chuan Guo et al. "Countering adversarial images using input transformations". In: *arXiv preprint arXiv:1711.00117* (2017).

[652] Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples". In: *International conference on machine learning*. PMLR. 2018, pp. 274–283.

[653] Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. "Jailbreaking leading safety-aligned llms with simple adaptive attacks". In: *arXiv preprint arXiv:2404.02151* (2024).

[654] Francesco Croce and Matthias Hein. "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks". In: *International conference on machine learning*. PMLR. 2020, pp. 2206–2216.

[655] Aleksander Madry et al. “Towards deep learning models resistant to adversarial attacks”. In: *arXiv preprint arXiv:1706.06083* (2017).

[656] Eric Wong and Zico Kolter. “Provable defenses against adversarial examples via the convex outer adversarial polytope”. In: *International conference on machine learning*. PMLR. 2018, pp. 5286–5295.

[657] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. “Certified adversarial robustness via randomized smoothing”. In: *international conference on machine learning*. PMLR. 2019, pp. 1310–1320.

[658] Linyi Li, Tao Xie, and Bo Li. “Sok: Certified robustness for deep neural networks”. In: *2023 IEEE symposium on security and privacy (SP)*. IEEE. 2023, pp. 1289–1310.

[659] Guy Katz et al. “Reluplex: An efficient SMT solver for verifying deep neural networks”. In: *International conference on computer aided verification*. Springer. 2017, pp. 97–117.

[660] Nicholas Carlini et al. “Stealing part of a production language model”. In: *arXiv preprint arXiv:2403.06634* (2024).

[661] Matthew Jagielski et al. “High accuracy and high fidelity extraction of neural networks”. In: *29th USENIX security symposium (USENIX Security 20)*. 2020, pp. 1345–1362.

[662] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. “Knockoff nets: Stealing functionality of black-box models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 4954–4963.

[663] Nicholas Carlini et al. “Extracting training data from large language models”. In: *30th USENIX security symposium (USENIX Security 21)*. 2021, pp. 2633–2650.

[664] Ahmed Salem et al. “ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models”. In: *Network and Distributed System Security Symposium (NDSS)*. 2019.

[665] Reza Shokri et al. “Membership inference attacks against machine learning models”. In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[666] Samuel Yeom et al. “Privacy risk in machine learning: Analyzing the connection to overfitting”. In: *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE. 2018, pp. 268–282.

[667] Shangbin Feng et al. “From pretraining data to language models to downstream tasks: Tracking the trails of political biases leading to unfair NLP models”. In: *arXiv preprint arXiv:2305.08283* (2023).

[668] Martin Abadi et al. “Deep learning with differential privacy”. In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318.

[669] Nicolas Papernot et al. “Scalable private learning with pate”. In: *arXiv preprint arXiv:1802.08908* (2018).

[670] Yossi Adi et al. “Turning your weakness into a strength: Watermarking deep neural networks by backdooring”. In: *27th USENIX security symposium (USENIX Security 18)*. 2018, pp. 1615–1631.

[671] Erwan Le Merrer, Patrick Perez, and Gilles Trédan. “Adversarial frontier stitching for remote neural network watermarking”. In: *Neural Computing and Applications* 32.13 (2020), pp. 9233–9244.

[672] Jialong Zhang et al. “Protecting intellectual property of deep neural networks with watermarking”. In: *Proceedings of the 2018 on Asia conference on computer and communications security*. 2018, pp. 159–172.

[673] John Kirchenbauer et al. “A watermark for large language models”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 17061–17084.

[674] Hanlin Zhang et al. “Watermarks in the sand: Impossibility of strong watermarking for generative models”. In: *arXiv preprint arXiv:2311.04378* (2023).

[675] Mika Juuti et al. “PRADA: protecting against DNN model stealing attacks”. In: *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2019, pp. 512–527.

[676] Ryan Feng et al. “Stateful defenses for machine learning models are not yet secure against black-box attacks”. In: *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 2023, pp. 786–800.

[677] Kimberley Kao. *Alibaba Unveils Upgraded AI Model, Claims It Surpasses Rival DeepSeek-V3*. The Wall Street Journal. 2025. URL: https://www.wsj.com/tech/ai/alibaba-unveils-upgraded-ai-model-claims-it-surpasses-rival-deepseek-v3-506b7f28.

[678] Ari Juels and Farinaz Koushanfar. “Props for Machine-Learning Security”. In: *arXiv preprint arXiv:2410.20522* (2024).

[679] Klim Kireev, Bogdan Kulynych, and Carmela Troncoso. “Adversarial robustness for tabular data through cost and utility awareness”. In: *NDSS Symposium*. 2023.

[680] Milad Nasr et al. “Scalable extraction of training data from (production) language models”. In: *arXiv preprint arXiv:2311.17035* (2023).

[681] Jiayuan Ye et al. “Enhanced membership inference attacks against machine learning models”. In: *Proceedings of the 2022 ACM SIGSAC conference on computer and communications security*. 2022, pp. 3093–3106.

[682] Rui Wen et al. “Membership inference attacks against in-context learning”. In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2024, pp. 3481–3495.

[683] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. In: *Machine learning* 20.3 (1995), pp. 273–297.

[684] Vitaly Feldman. “Does learning require memorization? a short tale about a long tail”. In: *Proceedings of the 52nd annual ACM SIGACT symposium on theory of computing*. 2020, pp. 954–959.

[685] Pang Wei Koh and Percy Liang. “Understanding black-box predictions via influence functions”. In: *International conference on machine learning*. PMLR. 2017, pp. 1885–1894.

[686] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[687] Kamil Kaczmarski et al. “Confidential Computing on NVIDIA H100 GPU: A Performance Benchmark Study”. In: *arXiv preprint arXiv:2409.03992* (2024). Version 2, October 2024. URL: https://arxiv.org/abs/2409.03992v2.

[688] Adrian Lutsch et al. “An Analysis of AWS Nitro Enclaves for Database Workloads”. In: *Proceedings of the 21st International Workshop on Data Management on New Hardware*. 2025, pp. 1–8.

[689] Steve Vassallo. *AI x Blockchain: The Next Level*. Accessed: March 2026. 2023. URL: https://www.forbes.com/sites/stevevassallo/2023/06/13/ai-x-blockchain-the-next-level/.

[690] VentionTeams. *Generative AI and Blockchain*. Accessed: March 2026. 2025. URL: https://ventionteams.com/blog/generative-ai-blockchain.

[691] Helen Partz. *AI-era Internet: Can Blockchain Prove What's Real Anymore?* Accessed: March 2026. 2025. URL: https://cointelegraph.com/news/ai-era-internet-can-blockchain-prove-what-is-real.

[692] Eric Mitchell et al. "DetectGPT: zero-shot machine-generated text detection using probability curvature". In: *Proceedings of the 40th International Conference on Machine Learning*. ICML'23. Honolulu, Hawaii, USA: JMLR.org, 2023.

[693] Sheng-Yu Wang et al. "CNN-Generated Images Are Surprisingly Easy to Spot... for Now". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 8692–8701. DOI: 10.1109/CVPR42600.2020.00872.

[694] Coalition for Content Provenance and Authenticity. *Content Credentials : C2PA Technical Specification*. Version 2.0. Accessed: February 2026. 2024. URL: https://c2pa.org/specifications/specifications/2.0/specs/C2PA_Specification.html.

[695] Numbers Protocol. *Numbers Protocol Whitepaper: Provenance Infrastructure for Humans and AI*. Accessed: February 2026. 2025. URL: https://docs.numbersprotocol.io/introduction/whitepaper/.

[696] Starling Lab for Data Integrity. *The Starling Framework for Data Integrity: Capture, Store, Verify*. Accessed: February 2026. 2024. URL: https://www.starlinglab.org/.

[697] Lyn Labs. *Introducing Lyn: Our Vision for Video AI and Human-Centric Agential Video*. Accessed: February 2026. 2025. URL: https://lynlabs.gitbook.io/lyn.

[698] LEAP IN. *How blockchain can solve AI's bias problem*. [Online; accessed 2026-02-14]. July 2025. URL: https://www.insights.onegiantleap.com/how-blockchain-can-solve-ais-bias-problem/#.

[699] Will Ogden Moore. *AI Is Coming — Crypto Can Help Make It Right*. [Online; accessed 2026-02-14]. URL: https://research.grayscale.com/reports/ai-is-coming-crypto-can-help-make-it-right.

[700] *AI Trust Problem: How Blockchain Can Solve It — Onchain Magazine*. [Online; accessed 2026-02-14]. URL: https://onchain.org/magazine/ai-trust-problem-how-blockchain-can-solve-it/.

[701] Dana Pessach and Erez Shmueli. "A review on fairness in machine learning". In: *ACM Computing Surveys* 55.3 (2022), pp. 1–44.

[702] Ninareh Mehrabi et al. "A survey on bias and fairness in machine learning". In: *ACM computing surveys* 54.6 (2021), pp. 1–35.

[703] Pawel Drozdowski et al. "Demographic bias in biometrics: A survey on an emerging challenge". In: *IEEE Transactions on Technology and Society* 1.2 (2020), pp. 89–103.

[704] Shashank Gupta et al. "Bias Runs Deep: Implicit Reasoning Biases in Persona-Assigned LLMs". In: *The Twelfth International Conference on Learning Representations*. 2024.

[705] Kristian Lum et al. "Bias in language models: Beyond trick tests and towards RUTEd evaluation". In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2025, pp. 137–161.

[706] Xudong Han, Timothy Baldwin, and Trevor Cohn. "Balancing out bias: Achieving fairness through balanced training". In: *Proceedings of the 2022 conference on empirical methods in natural language processing*. 2022, pp. 11335–11350.

[707] Debmalya Mandal et al. "Ensuring fairness beyond the training data". In: *Advances in neural information processing systems* 33 (2020), pp. 18445–18456.

[708] Cynthia Dwork et al. "Fairness through awareness". In: *Proceedings of the 3rd innovations in theoretical computer science conference*. 2012, pp. 214–226.

[709] Bashir Sadeghi and Vishnu Naresh Boddeti. "Imparting fairness to pre-trained biased representations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 16–17.

[710] Yi Dong et al. “Safeguarding large language models: A survey”. In: *Artificial intelligence review* 58.12 (2025), p. 382.

[711] Traian Rebedea et al. “Nemo guardrails: A toolkit for controllable and safe llm applications with programmable rails”. In: *Proceedings of the 2023 conference on empirical methods in natural language processing: system demonstrations*. 2023, pp. 431–445.

[712] Jacy Reese Anthis et al. “The impossibility of fair LLMs”. In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*. 2025, pp. 105–120.

[713] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. “Inherent Trade-Offs in the Fair Determination of Risk Scores”. In: *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. 2017, pp. 43–1.

[714] *Introducing Agentic Wallets: Give Your Agents the Power of Autonomy — Coinbase*. Feb. 11, 2026. URL: https://www.coinbase.com/developer-platform/discover/launches/agentic-wallets (visited on 02/27/2026).

[715] *What is x402? — Payment Protocol for AI Agents on Solana — Solana*. URL: https://solana.com/x402/what-is-x402 (visited on 02/27/2026).

[716] *Unleashing AI Agents: How Blockchain Enables True Digital Autonomy*. Feb. 12, 2025. URL: https://blog.sei.io/research/unleashing-ai-agents-how-blockchain-enables-true-digital-autonomy/ (visited on 02/27/2026).

[717] Stuart Jonathan Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 2022. 1288 pp. ISBN: 978-93-5606-357-0. Google Books: tfYG0AEACAAJ.

[718] Anton Wahrstätter et al. “Blockchain Censorship”. In: *Proceedings of the ACM Web Conference 2024*. WWW ’24. New York, NY, USA: Association for Computing Machinery, May 2024, pp. 1632–1643. ISBN: 979-8-4007-0171-9. DOI: 10.1145/3589334.3645431. (Visited on 02/04/2026).

[719] *Fork-Choice Enforced Inclusion Lists (FOCIL): A Simple Committee-Based Inclusion List Proposal - Proof-of-Stake / Block Proposer - Ethereum Research*. https://ethresear.ch/t/fork-choice-enforced-inclusion-lists-focil-a-simple-committee-based-inclusion-list-proposal/19870. (Visited on 09/21/2024).

[720] Sarisht Wadhwa et al. *AUCIL: An Inclusion List Design for Rational Parties*. 2025. (Visited on 03/21/2025).

[721] IBM. *How Blockchain Adds Trust to AI and IoT*. https://www.ibm.com/think/topics/blockchain-for-trustworthy-ai. IBM Think Blog. 2026.

[722] Aymen Boudguiga et al. “Towards better availability and accountability for iot updates by means of a blockchain”. In: *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE. 2017, pp. 50–58.

[723] Ben Laurie et al. *Certificate transparency version 2.0*. 2021.