Generative Adversarial Mining on Decentralized Networks

Felix Quinque, Kalei Brady, Steffen Cruz

Macrocosmos AI*

January 12, 2026

Abstract

Open-ended learning in decentralized AI systems is constrained by the difficulty of defining reliable, task-specific evaluation functions. Existing incentive mechanisms typically rely on deterministic verification, spot-checking, or delayed outcomes, limiting their applicability to subjective or non-verifiable domains.

We introduce **Generative Adversarial Mining (GAM)**, a general-purpose incentive mechanism inspired by adversarial learning. Participants alternate between generating solutions and discriminating their provenance, forming a strictly zero-sum game that evaluates quality implicitly through competition rather than explicit scoring. We formalize the game dynamics, analyze robustness to collusion, and show that rewards converge rapidly to underlying participant skill. We further demonstrate how controlled resource asymmetry prevents saturation and drives efficient distillation of high-quality solutions.

These results suggest that adversarial incentive mechanisms provide a principled foundation for decentralized AI systems capable of sustained improvement without handcrafted evaluation metrics.

*All authors may be contacted at {firstname}@macrocosmos.ai.

Contents

1 Introduction

Decentralized AI systems promise scalable, permissionless coordination of machine intelligence, but face a fundamental challenge: how to evaluate and reward useful contributions without centralized supervision. In many domains of interest—such as open-ended generation, reasoning, or creative synthesis—quality is inherently subjective or difficult to verify algorithmically. As a result, incentive mechanisms often rely on narrow task formulations, deterministic verification, or manually engineered scoring rules, which constrain innovation and limit generality.

A long-standing insight from machine learning is that adversarial dynamics can substitute for explicit evaluation. In Generative Adversarial Networks, a discriminator implicitly defines a notion of quality by competing with a generator, allowing models to improve without an externally specified loss function. This observation motivates a broader question: *can adversarial evaluation serve as a general incentive mechanism for decentralized learning systems?*

In this work, we answer this question in the affirmative by introducing **Generative Adversarial Mining (GAM)**, an incentive framework in which participants are rewarded through adversarial interaction rather than task-specific scoring. By embedding generation and discrimination directly into the reward structure, GAM enables decentralized systems to operate in domains where explicit evaluation is impractical, while remaining robust to collusion and strategic manipulation.

1.1 GANs in Machine Learning

Generative Adversarial Networks (GANs) are a class of machine learning frameworks introduced by Ian Goodfellow et al. in 2014 (1). They are built upon the concept of adversarial training, where two neural networks—the Generator and the Discriminator—compete in a minimax game. The Generator attempts to synthesize outputs that resemble samples from a target distribution, while the Discriminator evaluates whether a given sample originates from the real distribution or the Generator's approximation. Through iterative competition, both networks improve: the Generator learns to produce increasingly realistic data, and the Discriminator becomes more adept at detecting subtle differences. Generally, the Generator is the valuable artifact produced by this training process and can be used to create high quality synthetic samples such as photorealistic images.

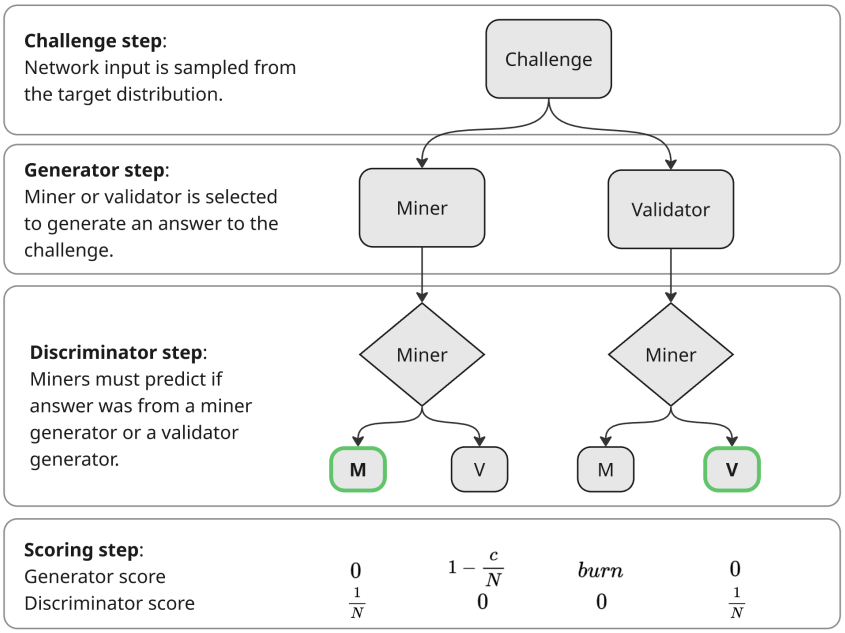

Figure 1: A single round of the *Generative Adversarial Mining* zero-sum game.

1.2 Bittensor

Bittensor (2) is a decentralized blockchain protocol designed to incentivize the development, deployment, and open distribution of machine intelligence. Inspired by Bitcoin's ability to coordinate global compute around hash-based work, Bittensor reimagines this dynamic for AI: instead of mining cryptographic puzzles, participants mine and validate useful machine learning outputs.

A key component of the Bittensor network is the *subnet*—a specialized sub-network within the protocol that defines a distinct incentive rubric. Each subnet enforces its own evaluation logic and reward distribution, allowing flexible experimentation across modalities, architectures, and task types. Today, Bittensor supports 128 subnets spanning domains from language modeling to vision, reinforcement learning, and multi-agent coordination.

Subnets are permissionless: anyone can join as a miner or validator, subject to the local scoring mechanism. Validators serve as decentralized auditors, assessing the quality of miner outputs and assigning emissions accordingly. This setup ensures that progress in AI is aligned with open participation, auditability, and economic accountability.

1.3 Incentive Mechanisms on Bittensor

Subnets typically operate with one of four fundamental types of incentive mechanism:

1. **Spot-checking:** Validators occasionally reproduce selected units of work performed by miners. This requires deterministic tasks (e.g., data scraping, simulations, inference) and usually leverages miners to speed up or scale up already established solutions. Given sufficient competitive pressure and penalties for failed spot checks, these mechanisms can increase network productivity by several orders of magnitude. For instance, Data Universe (subnet 13) validators are able to scrape a variety of data sources (3). Miners then scrape those same data sources at a much larger scale (250M records/day).

2. **Code attestation:** Miners run a specific program provided to them, which is confirmed using an attestation script. An example of such a system is IOTA (subnet 9), a pipeline-

parallel pre-training project (4). Here miners do not innovate, but compete by finding increasingly cheap GPUs with high network speeds.

3. **Prediction-based:** Miners make predictions for the future (e.g., trading, betting markets) which can trivially be validated at a later stage. For example, Vanta (subnet 8) tasks miners to manage a portfolio of equities, which are rewarded based on the portfolio performance after time t.

4. **Asymmetry-based scoring:** There are types of problems for which it is easy to generate challenge–answer pairs, while making it particularly hard to generate the answer without some additional knowledge. A good example of this is reversing a hash chain—it is trivial to generate the hash chain (here the challenge would be the $(n+1)$-th hash, the answer the n-th hash), but almost impossible to reverse it.

Spot-checking and code attestation both suffer from the problem that miners are not able to push beyond the state of the art. Prediction-based incentive mechanisms do not suffer from this problem, but only apply to time-based forecasting domains. Similarly, asymmetric systems allow for innovation, but require a very specific type of problem. An additional difficulty is that there are many problems that fit the basic requirement (easy to make a challenge–solution pair, hard to generate the solution for a miner), but providing an accurate score that reflects the miner's closeness to the solution is difficult.

2 Related Work

2.1 Adversarial Learning

Generative Adversarial Networks introduced adversarial training as a mechanism for learning implicit data distributions without explicit likelihoods or handcrafted objectives [1]. Subsequent work has explored adversarial dynamics across a wide range of modalities, demonstrating that competition can serve as a powerful substitute for explicit evaluation. GAM draws inspiration from this paradigm, but differs fundamentally in scope: rather than training a single model, we use adversarial interaction as an incentive mechanism governing a population of independent agents.

2.2 Decentralized Incentive Mechanisms for AI.

Prior decentralized AI systems typically rely on deterministic verification, spot-checking, or externally defined reward functions to evaluate contributions. Bittensor formalizes a peer-to-peer market for machine intelligence in which validators score miner outputs according to subnet-specific criteria [2]. While effective for well-defined tasks, these approaches require domain-specific evaluation logic and struggle in subjective or open-ended settings. GAM removes this dependency by outsourcing evaluation to adversarial discrimination, enabling a single mechanism to support a broad class of learning problems.

2.3 Peer Evaluation and Collusion Resistance.

Incentive mechanisms based on peer evaluation must address the risk of collusion and strategic manipulation. GAM's strictly zero-sum structure, combined with reward burning for validator outputs, leads to counterintuitive but desirable dynamics in which collusion is unstable and strictly suboptimal. This distinguishes GAM from cooperative or reputation-based schemes, where stable cabals can emerge.

3 Introducing Generative Adversarial Mining

Generative Adversarial Mining is a fundamentally new way to incentivize machine intelligence which is:

- Applicable to a wide range of problems,
- Allows for innovation,
- Completely eliminates the need for problem-specific scoring mechanisms.

The core idea behind *Generative Adversarial Mining* is inspired by GANs: we randomly choose either a validator or miner to generate an answer to some input challenge. Then, a group of other miners must guess whether the answer was generated by a miner or a validator. We then distribute points in a zero-sum fashion such that discriminators and generators are in direct competition. An illustration of the game is shown in Figure 1.

Over time, this dynamic drives the miner generators to accurately mimic the distribution of the validator outputs without requiring any specific evaluation rubric or reward functions. In other words, our approach provides a way to improve model performance in subjective, non-verifiable domains as it *outsources the difficult task of defining and measuring similarity* to the discriminator group.

The performance of GAN systems is known to saturate when generators are able to produce outputs which are indistinguishable from the target distribution (validator-generated outputs). For this reason, we introduce a secondary component to our system, *resource asymmetry*, which raises the performance ceiling of these networks. We allow validator generators to use more resources than miner generators to induce a skill gap between the competing network elements; extra processing time, access to private resources, and additional contextual information are all examples of resource asymmetry. In many computational domains, additional resources lead to predictably improved results (longer runtime, more memory, additional API calls, etc) and so our approach reframes the network saturation point as a controllable system parameter.

Asymmetry forces generator miners to innovate algorithmically; using exact or heuristic-based techniques to approximate the target distribution. In turn, this can produce efficient solutions that distill intelligence into a more lightweight form (faster, cheaper, less computationally intensive), providing clear economic benefits.

3.1 Game Setup

Let the system consist of a set of miners

$$\mathcal{M} = \{m_1, m_2, \ldots, m_{|\mathcal{M}|}\},$$

where each miner $m_i \in \mathcal{M}$ hosts both a *generator* G_i and a *discriminator* D_i. Additionally, we define a set of validators

$$\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\},$$

each of which is equipped with a *reference generator*. Validators serve as trusted oracles against which (potentially adversarial) miner generators are benchmarked. The adversarial game proceeds in *rounds*. Let the total number of rounds be denoted by R.

3.2 Round Dynamics

At the beginning of each round $t \in \{1, \ldots, T\}$, a validator $v \in \mathcal{V}$ is chosen uniformly at random to provide a challenge x_t. The challenge is the network input: generators are required to solve challenges and produce answers.

Generator Selection: With equal probability, the generator is selected either from the miner set or from the validator set. Formally,

$$G_t \sim \begin{cases} \text{Unif}(\{G_i : m_i \in \mathcal{M}\}) & \text{with probability } \frac{1}{2}, [6pt]\text{Unif}(\{G_j : v_j \in \mathcal{V}\}) \\ \text{with probability } \frac{1}{2}. \end{cases}$$

The selected generator then produces an answer

$$\hat{y}_t = G_t(x_t).$$

Discriminator Selection: A set of N discriminators is sampled from the miner pool:

$$\mathcal{D}_t = \{D_{t,1}, D_{t,2}, \ldots, D_{t,N}\} \subseteq \{D_i : m_i \in \mathcal{M}\}.$$

Each discriminator must make a prediction $p_i \in \{\text{miner}, \text{validator}\}$ whether A_t originated from a miner generator or a validator generator:

$$p_{i,t} = D_{i,t}(\hat{y}_t, x_t).$$

3.3 Scoring Mechanism (Zero-Sum)

Each round yields a total of 1 net point, ensuring the game is strictly zero-sum. Points are allocated as follows:

Discriminator Reward: Each discriminator $D_{i,t} \in \mathcal{D}_t$ receives

$$R_{D_{i,t}} = \begin{cases} \frac{1}{N}, & \text{if correct}, [6pt]0, \\ \text{if incorrect.} \end{cases}$$

Generator Reward: Let

$$c_t = \left|\{D_{i,t} \in \mathcal{D}_t : D_{i,t} \text{ correct}\}, \right|$$

be the number of correct discriminators. Then the generator's reward is

$$R_{G_t} = 1 - \frac{c_t}{N}.$$

Special Case (Validator Generator): If $G_t \in \{G_j : v_j \in \mathcal{V}\}$, i.e., the answer is produced by the validators' reference generator, then the validator will collect the reward R_{G_t}. The rewards assigned to the validator will be *'burned'* (not distributed to anyone) at the end of the game.

3.4 Zero-Sum Property

For every round t, the total sum of all rewards satisfies

$$R_{G_t} + \sum_{D_{i,t} \in \mathcal{D}_t} R_{D_{i,t}} = 1.$$

Thus, a fixed number of points are allocated; the system enforces a strict adversarial balance. Miners are rewarded based on their total points, which means that they must perform well both as a generator and a discriminator to remain registered on the network.

4 Cabal Resistance

4.1 Problem Setup

One of the main challenges with both collaborative and adversarial incentive mechanisms is the risk of collusion between miners. Let us assume that we have a cabal consisting of N out of our total of $|\mathcal{M}|$ miners.This cabal will share any useful information, in particular information on whether a given answer originates from one of the cabal's miners.

In this section we investigate the game dynamics in the presence of collusion and prove that such strategies are actually *less competitive* than playing fairly.

We assume:

- The prior probability of each outcome is $P(\text{validator-generated}) = P(\text{miner-generated}) = \frac{1}{2}$
- Each player has access to a classifier with accuracy p, where $P(\text{correct classification}) = p$
- A fraction $k = \frac{N}{|\mathcal{M}|}$ of players collude, where N is the number of colluding players
- The colluding players have verified that none of them generated the answer

4.2 Posterior Probability After Collusion

Given that a fraction k of players did not generate the answer, we compute the updated prior using Bayes' theorem. Since the answer must either come from the validator or be generated by one of the remaining $(1-k)|\mathcal{M}|$ non-colluding miners, the posterior probabilities become:

$$P(\text{validator-generated} \mid \text{collusion info}) = \frac{1}{2-k} P(\text{miner-generated} \mid \text{collusion info}) \quad = \frac{1-k}{2-k} \tag{1}$$

4.3 Prediction Certainty under Collusion

We are interested in the posterior certainty of a colluding miner after conditioning on the classifier's prediction. Using Bayes' theorem together with the adjusted priors

$$P(V) = \frac{1}{2-k}, \qquad P(M) = \frac{1-k}{2-k},$$

and classifier accuracy p, we obtain:

Case 1: Classifier predicts "validator-generated." The probability that this prediction is correct is

$$P(\text{correct} \mid C_V, k) = \frac{P(C_V \mid V)P(V)}{P(C_V \mid k)} \qquad = \frac{\frac{p}{2-k}}{\frac{1-k+kp}{2-k}} = \frac{p}{1-k+kp} = \frac{p}{1-k(1-p)}, \tag{2}$$

where C_V is the probability of the classifier predicting "validator" (without extra information).

Thus, the certainty given that the discriminator predicts 'validator' knowing that the answer was not generated by the cabal is

$$p_v = \frac{p}{1-k(1-p)}.$$

As expected, the probability of a correct prediction is bounded between the baseline accuracy p ($k = 0$) and 1 ($k = 1$).

Case 2: Classifier predicts "miner-generated." Analogously, we find

$$P(\text{correct} \mid C_M, k) = \frac{P(C_M \mid M)P(M)}{P(C_M \mid k)} \qquad = \frac{\frac{p(1-k)}{2-k}}{\frac{1-kp}{2-k}} = \frac{p(1-k)}{1-kp}. \tag{3}$$

Thus, the certainty given that the discriminator predicts 'miner' knowing that the answer was not generated by the cabal is

$$p_m = \frac{p(1-k)}{1-kp}.$$

Therefore, over the course of the game the colluding player's accuracy would simply be the weighted average of their validator and non-cabal miner accuracies.

$$a_c = \frac{p_v + p_m \cdot (1-k)}{2-k}$$

We can now plot a_c as a function of k, p and verify our intuition that a_c is strictly better than p, which should yield improved performance within the cabal.

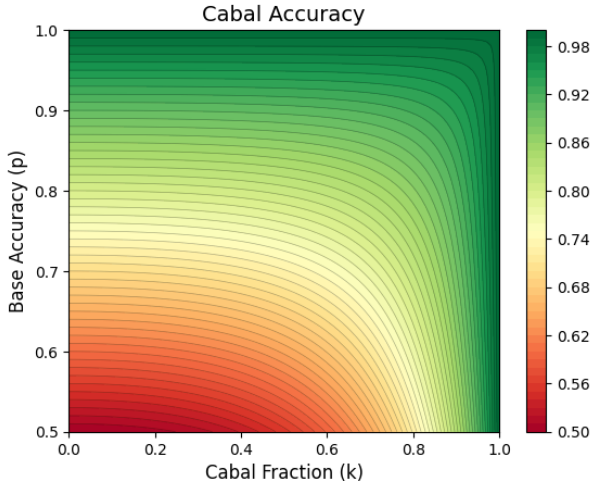

Figure 2: Cabal's accuracy as a function of cabal size k and base discriminator accuracy p

Trivially, if all miners are part of the cabal, they will have a 100% discriminator accuracy.

4.4 The Counterintuitive Nature of Reward Flow Under Collusion

At first glance, one might expect that maximizing discriminator accuracy would also maximize a miner's score. Surprisingly, this is not the case. In fact, a colluding cabal that leverages its shared information to improve accuracy ends up under-performing relative to non-cabal miners.

In this setting, when a discriminator is paired with a validator, there is one point at stake. If the discriminator loses, it effectively "loses" one point relative to the optimal outcome (since the point is burned rather than awarded to anyone else). By contrast, if the discriminator is paired against another generating miner, losing results in a relative loss of two points: rather than earning 1 point while the opponent earns 0 (+1 relative), the discriminator earns 0 while the opponent earns 1 (–1 relative).

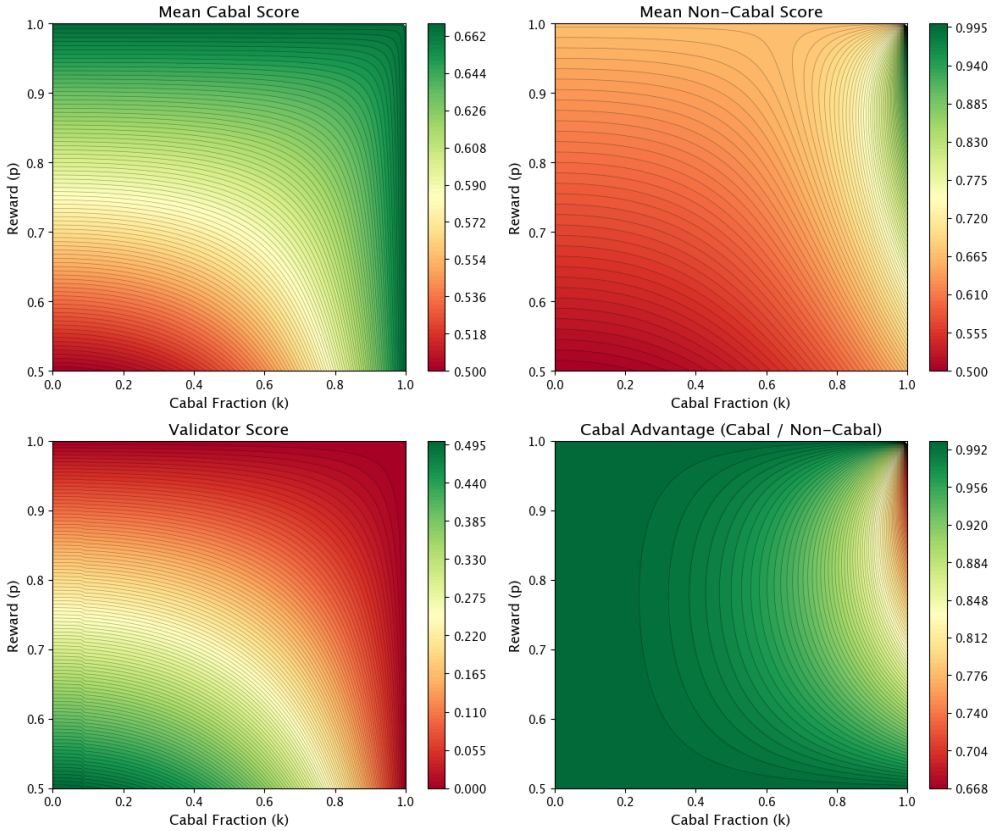

Figure 3: Rewards for network participants as a function of base classifier accuracy p and cabal size k. Plots A–D arranged top left to bottom right.

Figure 3 illustrates how participant scores vary with p and k. Figure 3(a) shows that cabal miner scores increase monotonically with cabal size, while figure 3(c) shows validator scores decrease correspondingly—an intuitive result, since cabal miners increasingly dominate validators.

Counterintuitively, figure 3(b) reveals that non-cabal miners' scores also rise with cabal size. Even more strikingly, figure 3(d) shows that across all (p, k) combinations, non-cabal miners consistently outperform cabal miners. This can be understood as follows: cabal miners improve their discrimination against validators, "rescuing" points that would otherwise be burned. However, this comes at the cost of a higher false negative rate against generator miners, causing more frequent losses. The rescued points are then redistributed, and if more than half of them flow to non-cabal miners, the cabal ends up worse off relative to the rest of the network.

4.4.1 Burning Emissions

In order to ensure that everyone's incentives remain aligned with top performance (i.e. having the best generator & discriminator) we reduce the total emissions based on the points scored by validators. In other words, points awarded to validators are removed from the total reward pool for miners. If, rather than burning the validator rewards, we were to re-normalize the miner payouts relative to their performance, cabal-miners would be able to effectively use their extra information to their benefit (by minimizing the non-colluding miner's scores). Notably, this would also minimize their own score, but to a lesser degree, therefore making it the dominant strategy under a fixed payout game.

With reward burning, the cabal still benefits from 'rescuing' points that would normally go to the validator, thereby keeping their incentives aligned with maximizing discriminator accuracy.

This leads to a fascinating outcome: Whilst a cabal will achieve higher rewards than if there was no cabal, non-cabal miners achieve higher scores yet, motivating the cabal members to leave the cabal. Therefore, the formation of a stable cabal is impossible, as the only state in Nash equilibrium is a completely cabal-less state. That means even if all participants were willing and able to collude, the game does not permit stable cabals.

5 Convergence of Rewards to 'True' Skill

Setup. We consider a Bradley–Terry (5) style system with two disjoint groups of players representing generators (G) and discriminators (D):

- Group G: players have skills $s_G \in (0, 0.5)$,
- Group D: players have skills $s_D \in (0.5, 1)$.

This guarantees that the best possible generator (skill 0.5) and the worst discriminator (equally skill 0.5) will mean that the discriminator's probability of winning is simply a coin flip, i.e. random guessing.

We will treat each discriminator vs generator pairing as a separate match; therefore given a committee of k discriminators and N rounds, we will be playing $k \cdot N$ matches. Discriminators are of course also matched up against validators half of the time, leading to them playing twice as many matches as generators. We can simply ignore this contribution and simply adjust for it at the end.

Therefore, ignoring validator matches, one player from G is paired with one player from D in each match. The probability that player i (skill s_i) defeats player j (skill s_j) is assumed to be.

$$p_{ij} = \frac{s_i}{s_i + s_j}.$$

We are interested in the correlation between the observed win rate $Y_i = W_i/N$ and the underlying skill s_i, where W_i denotes the number of wins after N matches.

Expected win probability. Because players only compete across groups, each skill s faces opponents from a distinct range. The expected win probability of a player with skill s is

$$g(s) = \begin{cases} \mathbb{E}_{X\sim\mathrm{Unif}(0.5,1)}\left[\dfrac{s}{s+X}\right], & s \in (0, 0.5), [8pt]\mathbb{E}_{X\sim\mathrm{Unif}(0,0.5)}\left[\dfrac{s}{s+X}\right], \\ s \in [0.5, 1). \end{cases} \tag{4}$$

Evaluating the integrals gives the closed forms

$$g(s) = \begin{cases} 2s \ln \dfrac{s+1}{s+0.5}, & s \in (0, 0.5), [8pt]2s \ln \dfrac{s+0.5}{s}, \\ s \in [0.5, 1). \end{cases} \tag{5}$$

Population-level correlation. Let $s \sim \mathrm{Unif}(0, 1)$. Then

$$\mathbb{E}[s] = \tfrac{1}{2}, \qquad \mathrm{Var}(s) = \tfrac{1}{12}.$$

Define the moments

$$\mathbb{E}[g(s)], \quad \mathbb{E}[g(s)^2], \quad \mathrm{Var}(g(s)) = \mathbb{E}[g(s)^2] - \mathbb{E}[g(s)]^2, \quad \mathrm{Cov}(g(s), s) = \mathbb{E}[s\, g(s)] - \mathbb{E}[s]\mathbb{E}[g(s)].$$

If each player plays N matches, then $W|s \sim \mathrm{Binomial}(N, g(s))$ and $Y = W/N$. The law of total variance gives

$$\mathrm{Var}(Y) = \mathrm{Var}(g(s)) + \frac{\mathbb{E}[g(s)(1 - g(s))]}{N}, \qquad \mathrm{Cov}(Y, s) = \mathrm{Cov}(g(s), s).$$

Thus, the correlation between observed score and skill is

$$\rho_N^{(\text{pop})} = \frac{\text{Cov}(g(s), s)}{\sqrt{\text{Var}(s)\left(\text{Var}(g(s)) + \frac{\mathbb{E}[g(s)(1-g(s))]}{N}\right)}}. \tag{6}$$

As $N \to \infty$, binomial noise vanishes and

$$\rho_\infty^{(\text{pop})} = \frac{\text{Cov}(g(s), s)}{\sqrt{\text{Var}(s)\,\text{Var}(g(s))}}. \tag{7}$$

Finite-league correction. In a finite league with n_G players in group G and n_D in group D, each player repeatedly faces random opponents drawn uniformly from the opposite group. Even as $N \to \infty$, the limited opponent pool introduces an extra source of heterogeneity. Let

$$h_2(s) = \begin{cases} 2s^2\left(\frac{1}{s+0.5} - \frac{1}{s+1}\right), & s \in (0, 0.5), [8pt] 2s^2\left(\frac{1}{s} - \frac{1}{s+0.5}\right), \\ s \in [0.5, 1), \end{cases} \tag{8}$$

which represents the second moment $\mathbb{E}_X[f(s, X)^2]$ of the match kernel $f(s, x) = s/(s + x)$ against the relevant opponent distribution. Then, by the law of total variance,

$$\mathbb{E}[\text{Var}(G(s, \mathcal{O})|s)] = \frac{1}{n_{\text{opp}}}\Big(\mathbb{E}[h_2(s)] - \mathbb{E}[g(s)^2]\Big),$$

where n_{opp} denotes the number of distinct opponents per player ($n_{\text{opp}} = n_D$ for group G, and $n_{\text{opp}} = n_G$ for group D). The total variance in observed win rates becomes

$$\text{Var}(Y) = \text{Var}(g(s)) + \frac{\mathbb{E}[g(s)(1-g(s))]}{N} + \left(1 - \frac{1}{N}\right)\frac{1}{n_{\text{opp}}}\Big(\mathbb{E}[h_2(s)] - \mathbb{E}[g(s)^2]\Big). \tag{9}$$

Substituting (9) into (6) yields the finite-league corrected correlation:

$$\rho_N^{(\text{finite})} = \frac{\text{Cov}(g(s), s)}{\sqrt{\text{Var}(s)\Big(\text{Var}(g(s)) + \frac{\mathbb{E}[g(s)(1-g(s))]}{N} + (1 - \frac{1}{N})\frac{1}{n_{\text{opp}}}(\mathbb{E}[h_2(s)] - \mathbb{E}[g(s)^2])\Big)}}. \tag{10}$$

The extra term slightly reduces the overall correlation, especially when the number of distinct opponents is small.

Interpretation. Given that we would like to accurately evaluate miners, we generally want to aim for a sufficiently high correlation, typically > 0.9. As Figure 4 shows, given that the committee size k can be very large, we only need very few rounds to achieve a good correlation. For instance in the case of 256 miners (typical Bittensor subnet capacity), given a discriminator committee size $k = 100$, we would already achieve a correlation > 0.9 once each miner was generator only once already.

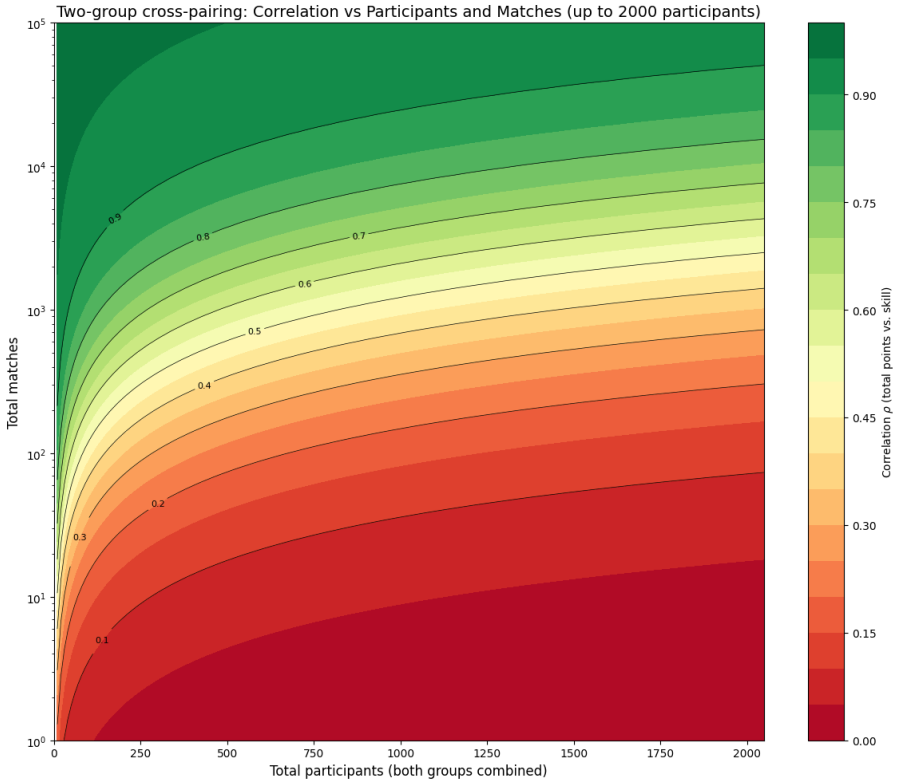

Figure 4: Correlation of underlying skill of a miner vs points scored.

This strong rate of convergence guarantees that *Generative Adversarial Mining* is sufficiently responsive to accurately distinguish between good & bad miners quickly, thereby ensuring a high rate of iteration and yielding fast improvement of miners.

6 Real-World Applications of GANs

The *Generative Adversarial Mining* framework has been successfully deployed across multiple production subnets within the Bittensor ecosystem, demonstrating its versatility and effectiveness across diverse domains.

6.1 Bittensor Subnet 1: Agentic Text Generation

The first implementation of this technique was deployed on Subnet 1, a text generation subnet focused on advancing the state of the art in agentic language models. In this configuration, validators serve as the reference oracle with access to extended computational budgets, enabling them to execute complex agentic workflows including iterative code execution, web search integration, and chain-of-thought reasoning with extended context windows.

Miner generators operate under significantly tighter time constraints, attempting to replicate the quality of validator outputs through efficient inference and optimized model architectures. Discriminators must then distinguish between these time-constrained miner responses and the computationally-intensive validator references based solely on output quality. In this context, the useful commodity produced by the network is the optimized agentic workflow deployed by the generators.

This asymmetric resource allocation creates a natural progression toward increasingly efficient high-quality generation: miners are incentivized to develop models and techniques that approach validator-level performance within strict latency bounds, effectively pushing the frontier of fast, high-quality agentic reasoning.

6.2 Bittensor Subnet 34: Adversarial Media Generation and Detection

A second deployment on Subnet 34 (6) applies the idea to the domain of AI-generated media detection. Here, the framework produces dual outputs of immediate practical value: both state-of-the-art generative models and highly capable detection classifiers.

Discriminators in this subnet develop and submit classifiers designed to detect AI-generated images and videos with high precision. Generators, in turn, are tasked with producing synthetic media capable of evading these detection systems. This adversarial dynamic naturally drives both components toward improved performance: generators must produce increasingly realistic outputs that preserve subtle distributional properties of authentic media, while discriminators must identify ever more nuanced artifacts and statistical signatures of generation. In contrast to the previous section, the valuable commodity produced in this case is in fact the discriminatory capabilities of miners.

The resulting system produces two valuable artifacts simultaneously: (1) advanced generative models capable of high-fidelity media synthesis, and (2) robust detection systems crucial for content authentication and provenance verification. Both outputs represent frontier capabilities in their respective domains, demonstrating how GAM's game-theoretic structure can yield multiple streams of useful innovation from a single incentive mechanism.

6.3 Future Directions

These deployments validate our core thesis: that adversarial dynamics, properly structured, can drive decentralized innovation across a wide range of machine learning domains without requiring hand-crafted evaluation functions. This work alleviates a substantial bottleneck in open-ended decentralized systems which can, in principle, improve network outputs in an effectively unbounded manner. The framework's success in both language and vision modalities suggests natural extensions to other domains where reference solutions can be computed at higher cost or with additional resources, including reinforcement learning, scientific computing, and multi-modal reasoning tasks.

We also note that *Generative Adversarial Mining* enables two potentially useful solutions to be produced in a single process; generative systems which produce high quality outputs and discriminatory systems which act as high-precision classifiers in the target domain. Naturally, it is an interesting avenue for further research to extend the design to include *capturing* the generator and discriminator models themselves, rather than the outputs alone. In this way, the mechanism can be used to capture innovations and optimizations directly, which can then be directly commercialized. Such a system is currently under active development, and will be discussed further in an upcoming publication.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. *NeurIPS*, 2014.

[2] Bittensor Foundation. *BitTensor: A Peer-to-Peer Intelligence Market*. arXiv:2003.03917, 2020.

[3] Data Universe. *Subnet 13: Decentralized Social Media Data Collection on Bittensor*. Macrocosmos.ai, 2025. `https://datauniverse.macrocosmos.ai/`.

[4] F. Quinque *et al. Incentivised Orchestrated Training Architecture (IOTA): A Technical Primer for Release*. Macrocosmos AI White Paper, 2025.

[5] R. A. Bradley and M. E. Terry. *Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. Biometrika*, vol. 39, no. 3–4, pp. 324–345, 1952.

[6] Bitmind. *Bitmind: AI Content Verification*. Bitmind.ai, 2025. Available at: `https://bitmind.ai/`.