DeepNode: The Evolution of Decentralized AI

J. Ruff, M. Agawa et al.

February 2024

1 Abstract

Markets have traditionally been used to efficiently allocate resources and commodities. We propose a novel approach to producing machine intelligence by establishing a market where intelligence is priced by node-to-node interactions among intelligence systems across the internet. In this proposed market, nodes evaluate and rank each other by training neural networks to assess the value of their neighbors. These evaluations are recorded on a digital ledger, where nodes with higher rankings are rewarded monetarily with additional weight in the network.

A significant challenge in this node-ranking system is its vulnerability to collusion, which can compromise the accuracy of the rankings. To address this, we introduce an incentive mechanism that maximally rewards honestly selected weights, thereby making the system resistant to collusion affecting up to 50 percent of the network weight. This mechanism ensures a fair and reliable intelligence market that continuously generates new models and compensates contributors who provide information-theoretic value.

Similar approaches have demonstrated the utility of markets in various domains, from the valuation of AI models [1] to the licensing of advanced technologies [2]. The proposed market framework aims to extend these principles to machine intelligence, fostering a decentralized and efficient production environment.

The current approach to generating machine intelligence largely depends on benchmarking systems, where machine learning models are optimized for narrowly defined, supervised tasks. Although effective for enhancing performance in specific areas, this method falls short in contexts where market dynamics could be advantageous. Intelligence is increasingly being seen as a commodity that is (1) costly to extract from data [1], (2) financially lucrative [2], (3) transferable across different domains [3], and (4) broadly applicable [4]. Relying solely on supervised objectives to measure intelligence production fails to reward the intrinsic value of the commodity and leads to a focus on specialized, narrow applications [5]. Additionally, these objectives, often quantified using

single metrics like accuracy, lack the granularity needed to recognize the worth of niche or legacy systems, resulting in their obsolescence. The diversity of intelligence systems is stifled by the necessity to develop large, monolithic models that dominate in a competitive landscape. This centralization restricts independent engineers from monetizing their innovations, leading to a concentration of control among a few large entities.

Emerging commodities necessitate innovative market structures. This paper proposes a system where machine intelligence is evaluated by other intelligence systems. Models are assessed based on their informational output, independent of the specific tasks or datasets they were trained on. This shift in evaluation criteria allows the market to (1) reward intelligence that serves a broader range of objectives, (2) monetize legacy systems for their distinct contributions, and (3) enable smaller, diverse systems to thrive in a more granular reward environment. The proposed solution involves a network of computers that continuously and asynchronously exchange representations node-to-node (P2P) over the internet. This market leverages a digital ledger to record rankings and provide decentralized incentives, ensuring that trust is measured and rewards are distributed based on value provided to the majority. This framework allows researchers to monetize their contributions directly, and consumers to purchase machine intelligence services directly.

2 Design

We commence with an abstract depiction of intelligence, as illustrated by Hinton et al. [6], utilizing a parameterized function $y = g(z)$ trained on a dataset $D = [Z, Y]$ to minimize a loss function $\mathcal{L} = E_{\mathcal{D}}[Q(y, g(z))]$. Our network comprises n functions $N = \{n_1, \ldots, n_n\}$, termed 'nodes', each possessing varying amounts of network weight $\mathcal{S} = [w_i]$ documented on a digital ledger. These functions, combined with their respective losses and stake portions, create a stake-weighted machine learning objective given by $\sum_{i=1}^{n} \mathcal{L}_i \cdot w_i$.

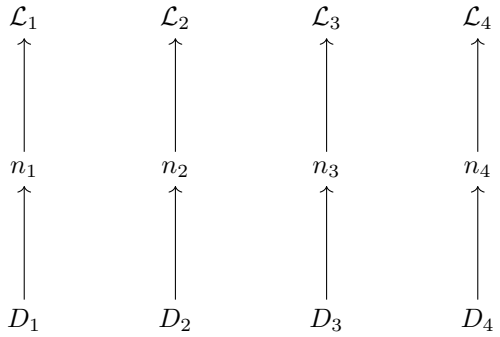

Figure 1: node functions with losses, $\mathcal{L}_i$, and unique datasets D_i.

Our objective is to allocate stake $\mathcal{I}$ as a reward to nodes that contribute to minimizing the loss function (Figure-1). Crucially, this allocation method ensures that it is challenging for a minority of stakeholders to collude and increase their share in the network without effectively reducing the overall loss (Figure-3).

$$\mathcal{S}_{t+1} = \mathcal{S}_t + \tau\mathcal{I}$$

In this study, we propose that this objective can be accomplished via node-ranking. Here, nodes utilize the outputs of other nodes $\mathcal{G}(z) = [g_1(z), \ldots, g_n(z)]$ as their own inputs $g(\mathcal{G}(z))$ and determine a set of weights $\mathcal{W} = [w_{i,j}]$, where each node i manages the i-th row through transactions recorded on a blockchain.

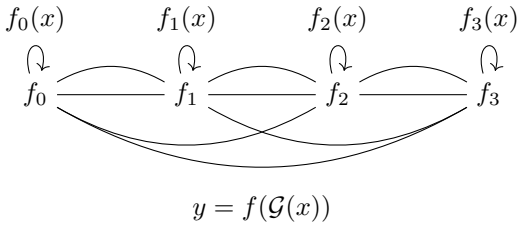

Figure 2: (a) Inter-function connectivity.

Weights are assigned using Fisher information pruning scores [7,8] in the ranking calculation. The ranking $\mathcal{R} = \mathcal{W}^T \cdot \mathcal{S}$ provides an optimal scoring method where each node's incentive aligns with its pruning score, defined as the entropy cost towards $\sum_{i=1}^{n} \mathcal{L}_i \cdot s_i$ incurred by excluding it from the network.

$$r_i \approx \frac{1}{n} \sum_{j=1}^{n} \sum_{x \in D_j} \Delta\mathcal{F}^T(x)_i \cdot \mathcal{H}(\mathcal{Q}_j(x)) \cdot \Delta\mathcal{F}(x)_i$$

Despite its merits, this method is vulnerable to collusion, where nodes self-promote by voting for themselves rather than applying the ranking formula.

They manipulate the weights to inflate their own rankings, undermining the network's integrity. This issue arises because the digital ledger lacks the capability to verify individual model parameters, auditing only the inter-node weights $\mathcal{W}$.

3 Rewards

To prevent collusion, we have developed an improved ranking method incorporating an 'incentive' function $\mathcal{I}(\mathcal{W}, \mathcal{S})$. This function ensures that rewards are given only to nodes that have not yet reached a consensus in the network. Provided that no single group possesses more than half of the total stake, nodes can only increase their stake by attracting votes from the majority.

To clarify, our incentive mechanism utilizes a stake vector $\mathcal{S}$ and a weight matrix $\mathcal{W}$, where the rows reflect inter-node rankings. Additionally, we infer a trust matrix $\mathcal{T}$ from the weight matrix, where $t_{i,j} = 1$ indicates a non-zero connection between node i and node j.

We identify nodes that have achieved 'consensus' as those with non-zero connections from more than 50 percent of the total stake in the network. This is represented by the normalized values of $(\mathcal{T}^T \cdot \mathcal{S}) > 0.5$. To maintain differentiability in our mechanism, we employ the continuous sigmoid function. This function generates a threshold-like behavior, rewarding well-connected nodes while penalizing those that are not trusted. The sigmoid's steepness and threshold can be adjusted using a temperature parameter ρ and a shift term κ.

$$\mathcal{C} = \sigma(\rho(\mathcal{T}^T \cdot \mathcal{S} - \kappa))$$

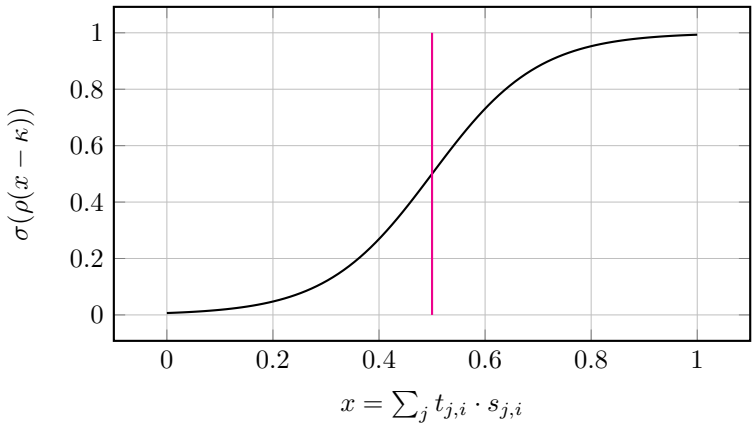

Figure 3: Consensus function $c_i = \sigma(\rho \sum_j t_{j,i} s_j - \kappa)$ with temperature $\rho = 10$ and shift $\kappa = 0.5$. The activation takes the trust scores and produces an exponential scaling up to our inflection point where a node is connected to the majority.

We apply the consensus term to adjust the initial rankings. As nodes gain more weight in the network, their inflation increases exponentially up to 0.5. Later, we demonstrate how this mechanism ensures that the larger of two competing sub-networks acquires an exponentially greater share of the network by inflation.

$$\mathcal{I} = \mathcal{R} \cdot \mathcal{C}$$

4 Bonds

The previously discussed consensus mechanism guards against straightforward collusion by making it challenging for small groups to manipulate inflation. However, it lacks an incentive for the accurate selection of weights. To address this, we modify the inflation mechanism to include a speculation-based reward system called 'bonds' $\mathcal{B}$. In this context, $b_{i,j} \in \mathcal{B}$ represents the proportion of bonds held by node i in node j.

Bonds accrue at each iteration in a manner akin to token inflation, where $\Delta\mathcal{B} = \mathcal{W} \cdot \mathcal{S}$. Through this process, nodes gather bonds in the nodes they rank, effectively 'bonding' themselves to those with whom they are connected.

$$\mathcal{B}_{t+1} = \mathcal{B}_t + \mathcal{W} \cdot \mathcal{S}$$

Utilizing the bond matrix $\mathcal{B}$, the system redistributes the standard incentive scores $\Delta\mathcal{S} = \mathcal{B}^T \cdot \mathcal{I}$. Similar to market speculation in traditional equities, nodes that hold bonds in other nodes that are later recognized by others increase their own inflation. Therefore, it is logical for nodes to accumulate bonds in other

nodes that are likely to perform well, as indicated by other stakeholders in the network—essentially speculating on their future performance. This mechanism is slightly modified to ensure nodes receive a fixed portion of their own inflation. For example, at 50%, $\Delta\mathcal{S} = 0.5\mathcal{B}^T\mathcal{I} + 0.5\mathcal{I}$. The update step $\Delta\mathcal{S}$ governs the distribution of network incentives among the n nodes.

$$\mathcal{S}_{t+1} = \mathcal{S}_t + \tau\Delta\mathcal{S}$$

5 Finding Consent

While the incentive mechanism rewards highly trusted nodes, it may not entirely prevent collusion if honest nodes fail to achieve consensus. Unused or loosely held stakes, as well as improperly set weights, can reduce the inflation share of honest nodes when compared to a colluding sub-network. Even with more stake, the honest network might not generate sufficient inflation to counter its adversary. The dishonest sub-network only needs to generate enough inflation to rival its main competitor, not necessarily to rule the network completely.

This issue arises when most of the token inflation is allocated to nodes that are not trusted by the majority. To address this, the network employs a loss term $\mathcal{L} = -\mathcal{R}\cdot(\mathcal{C}-0.5)$. This term becomes negative if the majority of inflation goes to nodes with more than 50% consensus. The network uses this loss calculation as a corrective mechanism. Through increasing the number of weights an average node assigns across the network, consensus can be better ensured.

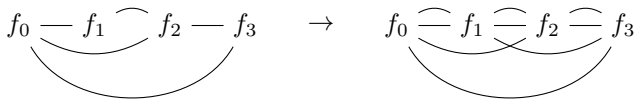

Figure 4: The left network demonstrates low consensus with $\mathcal{L} > 0$. This configuration is vulnerable to collusion by a minority group holding less than 50% of the total stake. To counter this, the system increases the number of connections established by nodes until $\mathcal{L} < 0$. When this occurs, inflation is directed towards nodes with greater consensus.

6 Node Operation

To operate a node in the network, follow these steps:

1. The node defines its dataset $\mathcal{D}_i$, loss function $\mathcal{L}_i$, and parameterized function f_i.

2. At each training iteration, the node broadcasts batches of samples from $\mathcal{D}_i$ to its nodes as $x = [\text{batch\_size}, \text{sequence\_length}, \text{input\_size}]$.

3. The responses $\mathcal{F}(x) = [\dots f_j(x) \dots]$ from each node, having a common shape $f_j(x) = [\text{batch\_size}, \text{sequence\_length}, \text{output\_size}]$, are combined using the gating function and used as input to the local model f_i.

4. Comparing these responses to the target labels generates a loss-gradient $\frac{\partial \mathcal{L}}{\partial \mathcal{F}}$, which is back-propagated through f_i and disseminated to the network.

5. During steps 2 and 3, nodes learn the weights for their row $w_{i,j} \in \mathcal{W}$ by evaluating the signal values produced by their nodes.

6. At specific time steps t, participants submit updates to the weights $\Delta\mathcal{W}_i$ to adjust the ranking $\mathcal{R}$, inflation $\mathcal{I}$, consensus term $\mathcal{C}$, and bond distributions $\Delta\mathcal{B}$.

7. The network assesses loss and optionally distributes newly created stake into the network $\Delta\mathcal{S}$ according to bond ownership.

7 Creating Regularity

A unified encoding for inputs and outputs is essential for the interaction of different model types and input forms. Tensor modalities can be utilized to segment the network into separate graphs. Initially, the network can start with a single modality, such as TEXT, and later expand to include IMAGE, SPEECH, and TENSOR. Eventually, combinations of these modalities, like TEXT-IMAGE, can be incorporated to transition into a multi-modality environment.

To encourage the integration of different modalities, incentives can be aligned with the same trust scaling mechanism. Successful models should eventually be capable of receiving inputs from any modality and converting them into a useful representation. For uniformity, a standard output shape, such as $[\text{batch\_size}, \text{sequence\_dim}, \text{output\_dim}]$, similar to the typical tensor shapes used in language and image models, can be adopted across the network and adjusted as the network's complexity grows.

Focusing on abstract input classes helps ensure that participants develop a broad multi-task understanding [9]. Participants might employ: 1. Entirely different computing frameworks [10], 2. Diverse datasets [11], 3. Various models, and 4. Unique strategies to optimize their incentives within the market. It is practical for nodes to engage with unsupervised datasets, where data is inexpensive and privacy concerns are minimal.

8 Calculation Conditions

As the network scales, managing outward bandwidth effectively becomes crucial to prevent major bottlenecks. To address this, reducing network transfer and implementing an efficient node selection method is essential. Conditional computation can facilitate this by allowing nodes to use gradient descent to learn how to select and prune their neighbors within the network. Techniques such as a product key layer or a sparsely gated layer [12] can be employed to achieve this.

$$f_i = f_i(G(x))$$

$$G(x) = \sum_j g_j(x) * f_j(x)$$

The conditional layer selects a sparse subset of nodes to query for each example and recombines them multiplicatively. This approach reduces outward bandwidth by querying only a limited number of nodes per example. This method can significantly enhance outward bandwidth efficiency [12, 13], enabling nodes to communicate with a larger number of neighbors within the network. Essentially, the layer functions as a trainable DNS lookup for nodes, based on inputs. Additionally, since it is trainable with respect to the loss, it serves as an effective proxy for the weights $w_{i,j} \in \mathcal{W}$.

9 Deriving Information

The interdependence of functions mandates that models must remain online and are unsuitable for production. This reliance can be eliminated through distillation [6], a technique for compressing and extracting knowledge. In this process, a smaller model—the student—replicates the behavior of the broader network. The distillation layer works alongside the conditional computation layer, where the student model learns to imitate the network using the cross-entropy (denoted as KL) between the logits from the gating network and the student's predicted output [14].

$$\text{distillation loss} = \text{KL}_D(\text{dist}(x), G(x))$$

Since the distilled model serves as a proxy for the network, models can be completely taken offline and evaluated. This separation allows recursion within the network to be cut between components, enabling arbitrary network graphs. If models go offline, their peers can use the distilled versions as replacements. Private data can be validated using the distilled models rather than querying the network. Eventually, components can fully detach from the network by using the distilled models for offline validation and inference.

10 Learning Weights

The objective of this work is to create a ranking $r = [r_i]$ over nodes, where the score $r_i \in \mathcal{R}$ indicates a participant's information-theoretic importance to the benchmark. Inspired by the work of LeCun et al. [7] and Yu et al. [8], it is logical to define this importance by equating it with the cost of removing each node from the network. We can analytically derive this score, where $\Delta\mathcal{F}(x)_i$ represents the perturbation of the j-th node's inputs when the i-th node is excluded from the network:

$$r_i \approx \frac{1}{n} \sum_{j}^{n} \sum_{x \in \mathcal{D}_j} \Delta\mathcal{F}^T(x)_i \cdot \mathcal{H}(\mathcal{Q}_j(x)) \cdot \Delta\mathcal{F}(x)_i$$

$$\Delta\mathcal{F}(x)_i = [0, \ldots, 0, -f_i(x), 0, \ldots, 0]$$

Note that when the error function $\mathcal{Q}_j$ is the twice-differentiable cross-entropy, $\mathcal{H}(\mathcal{Q}_j)$ becomes its Fisher information matrix. Thus, $r_i \in \mathcal{R}$ can effectively measure each node's informational significance to the entire network. However, calculating information-theoretic weights requires the complete Hessian of the error function. It is more feasible to employ a heuristic to propagate a contribution score from the error function to the inputs [8]. For example, weights from the gating layer serve as a practical, differentiable proxy.

11 Complicity

Consider a scenario where a subset of nodes in the network form a cabal: a group of colluding nodes attempting to maximize their inflation without accurately evaluating their neighbors. The competition between the honest sub-graph A with stake S_A and the dishonest cabal B with stake S_B is determined by the proportion of network stake held by each group. The honest sub-graph must achieve higher inflation to maintain its dominance and safeguard the network, ensuring $I_A \gg I_B$.

We assume the honest sub-graph A holds a larger proportion of the stake than the dishonest sub-graph B, $(S_A > S_B)$ and that the chain has reached a consensus with $\mathcal{L} < 0$. Since all nodes in B are distinct from A, our loss term $-R_B \cdot (C_B - 0.5) > 0$ is positive. Given that $\mathcal{L} < 0$, it must be the case that $R_A \cdot (C_A - 0.5) < 0$, indicating there are nodes in the honest sub-graph A connected to the majority.

As the chain progresses, newly minted stake is emitted at an inflation rate τ proportional to $I = R \cdot T$. Notably, the gradient of the incentive function with respect to the stake is positive and super-linear at our inflection point between the honest and dishonest sub-graphs. Specifically, $\frac{\delta I}{\delta S} = \frac{5}{2}$, ensuring that the amount of stake held by each sub-graph reflects a non-linear change in their

inflation at the next iteration.

Initially, since $S_A > 0.5$ and $S_B < 0.5$, the proportion of stake emitted in sub-graph A surpasses that in sub-graph B, and the incentive for sub-graph A grows super-linearly compared to B. Consequently, the ratio of stake $\frac{S_B}{S_A+S_B}$ decreases, forcing the cabal to continually add stake to its sub-graph to sustain itself over time.

12 Conclusion

We have introduced an intelligence market operating on a P2P network outside a trusted environment. The benchmark measures performance based on representational-knowledge production, using other intelligence systems to assess its value. This collaborative and high-resolution approach suggests that the benchmark could provide a more effective reward mechanism for the field overall.

To accomplish this, we began by defining a P2P network of abstract intelligence models. We demonstrated how this framework enables the creation of a ranking for each node based on the cost of removing it from the network. nodes negotiated this score using a set of weights recorded on a digital ledger. However, the system needed mechanisms to prevent participants from forming dishonest sub-graphs.

To address this, we proposed an incentive scheme based on node connectivity, which exponentially rewarded nodes for being trusted by a significant portion of the network. This approach ensures that dishonest sub-graphs diminish in importance over time.

Additionally, we illustrated 1. How nodes could reduce network bandwidth by learning connectivity using a differential layer and 2. How they could extract fully network-disconnected machine learning models for production use. The outcome is an intelligence market that rewards participants for generating knowledge and making it accessible to new learners within the system.

References

[1] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai," *arXiv preprint arXiv:1907.10597*, 2019.

[2] OpenAI, "Openai licenses gpt-3 technology to microsoft," *OpenAI Blog*, vol. 1, no. 1, p. 1, 2020.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2019.

[4] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[5] F. Chollet, "On the measure of intelligence," *arXiv preprint arXiv:1911.01547*, 2019.

[6] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[7] Y. LeCun, J. S. Denker, S. A. Solla, R. E. Howard, and L. D. Jackel, "Optimal brain damage," vol. 2, pp. 598–605, 1989.

[8] R. Yu, A. Li, C. F. Chen, J.-H. Lai, V. I. Morariu, X. Han, M. Gao, C.-Y. Lin, and L. S. Davis, "Nisp: Pruning networks using neuron importance score propagation," pp. 9194–9203, 2017.

[9] Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit, "One model to learn them all," *arXiv preprint arXiv:1706.05137*, 2017.

[10] M. A. Nugent and T. W. Molter, "Cortical processing with thermodynamic-ram," *arXiv preprint arXiv:1404.5282*, 2014.

[11] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv preprint arXiv:1901.07291*, 2019.

[12] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," *arXiv preprint arXiv:1701.06538*, 2017.

[13] M. Ryabinin and A. Gusev, "Towards crowdsourced training of large neural networks using decentralized mixture-of-experts," *arXiv preprint arXiv:2002.04013*, 2020.

[14] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2020.