FaceBlock : A Peer-to-Peer Intelligence Market

00/ ABSTRACT

As with other commodities, markets could help us efficiently produce machine intelligence. We propose a market where intelligence is priced by other intelligence systems peer-to-peer across the internet. Peers rank each other by training neural networks which learn the value of their neighbors. Scores accumulate on a digital ledger where high ranking peers are monetarily rewarded with additional weight in the network. However, this form of peer-ranking is not resistant to collusion, which could disrupt the accuracy of the mechanism. The solution is an incentive mechanism that maximally rewards honestly selected weights, making the system resistant to collusion of up to 50 percent of the network weight. The result is a collectively run intelligence market that continually produces newly trained models and pays contributors who create information theoretic value.

0.1/ INTRODUCTION

The production of machine intelligence has come to rely almost entirely on a system of benchmarking, where machine learning models are trained to perform well on narrowly defined supervised problems. While this system works well for pushing the performance on these specific problems, the mechanism is weak in situations where the introduction of markets would enable it to excel. For example, intelligence is increasingly becoming untethered from specific objectives and becoming a commodity that is (`1`) expensively mined from data (Schwartz et al. [2019]), (`2`) monetarily valuable (OpenAI [2020]), (`3`) transferable (Devlin et al. [2019]), and (`4`) generally useful (Radford et al. [2019]). Measuring its production with supervised objectives does not directly reward the commodity itself and causes the field to converge toward narrow specialists (Chollet [2019]). Moreover, these objectives (often measured in uni-dimensional metrics like accuracy) do not have the resolution to reward niche or legacy systems, thus what is not currently state of the art is lost. Ultimately, the proliferation of diverse intelligence systems is limited by the need to train large monolithic models to succeed in a winner-take-all competition. Standalone engineers cannot directly monetize their work and what results is centralization where a small set of large corporations control access to the best artificial intelligence (OpenAI [2020]).

A new commodity needs a new type of market (`1`). This paper suggests a framework in which machine intelligence is measured by other intelligence systems. Models are ranked for informational production regardless of the subjective task or dataset used to train them. By changing the basis against which machine intelligence is measured, (`1`) the market can reward intelligence that is applicable to a much larger set of objectives, (`2`) legacy systems can be monetized for their unique value, and (`3`) smaller diverse systems can find niches within a much higher resolution reward landscape. The solution is a network of computers that share representations continuously and asynchronously, peer-to-peer (P2P) across the internet. The

constructed market uses a digital ledger to record ranks and to provide incentives to peers in a decentralized manner. The chain measures trust, making it difficult for peers to attain rewards without providing value to the majority. Researchers can directly monetize machine intelligence work and consumers can directly purchase it.

01 MODEL

We begin with an abstract definition of intelligence Hinton et al. [2015] in the form of a parameterized function $y = f(x)$ trained over a dataset $D = [X, Y]$ to minimize a loss $\mathcal{L} = E_D[Q(y, f(x))]$. Our network is composed of n functions $F = f_0, ..., f_j, ...f_n$ 'peers' where each is holding zero or more network weight $S = [s_i]$ 'stake' represented on a digital ledger. These functions, together with losses and their proportion of stake, represent a stake-weighted machine learning objective $\sum_i^n \mathcal{L}_i * s_i$

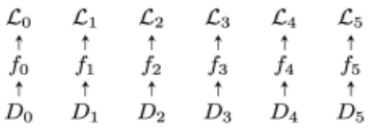
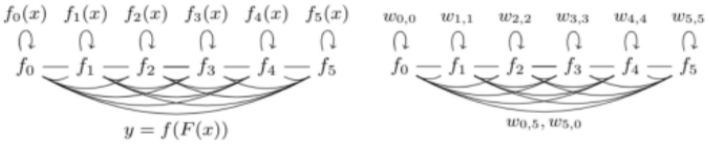


FIGURE 1 / Peer functions with losses $\mathcal{L}_i$ and unique datasets D_i

Our goal is the distribution of stake I, as an incentive, to peers who have helped minimize the loss-objective (Figure-1), and importantly, in such a way that, it is difficult for a small proportion of stake to collude as a means to maximize their distribution in the network without minimizing the loss (Figure-3).

$$S_{t+1} = S_t + \tau I \tag{1}$$

In this paper, we suggest this can be achieved through peer-ranking, where peers use the outputs of others $F(x) = [f_0(x)...f_n(x)]$ as inputs to themselves $f(F(x))$ and learn a set of weights $W = [w_{i,j}]$ where peer i is responsible for setting the i th row through transactions on a digital ledger.



(a) Inter-function connectivity. (b) Weight matrix.

Setting weights using an fishers information pruning score LeCun et al. [1989]; Yu et al. [2017] in the ranking calculation, $R = W^T \cdot S$ achieves an idealized scoring where each peer's incentive is equivalent to its pruning score: the cost in entropy towards $\sum_i^n \mathcal{L}_i * s_i$ induced by removing it from the network.

$$r_i \approx \frac{1}{n} \sum_{j}^{n} \sum_{x \in D_j} \Delta F^T(x)_i \cdot H(Q_j(x)) \cdot \Delta F(x)_i \tag{2}$$

However, this approach is not resistant to collusion, where peers vote for themselves, notably instead of using (2), and set weights to enhance their own inflation at the expense of the network(Figure-3). This attack is trivial since the digital ledger cannot audit the parameters of each model, only the inter-model weights W.

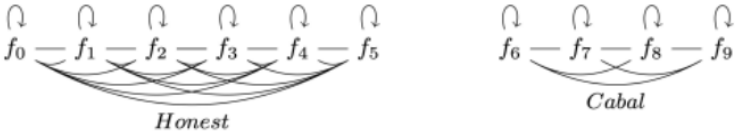


FIGURE 3 / Disjoint cabal: peers in the right sub-network only vote for themselves.

$$C = \sigma(\rho(T^T S - \kappa)) \tag{4}$$

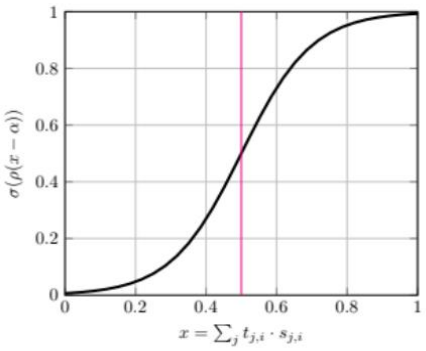


FIGURE 4 / Consensus function $c_i = \sigma(\rho \sum_j^n t_{j,i} s_j - \kappa)$ with temperature ρ = 10 and shift κ = 0.5. The activation takes the trust scores and produces an exponential scaling up to our inflection point where a peer is connected to the majority.

We use the consensus term to scale the original rankings. As peers attain more weight in the network they increase their inflation exponentially up to 0.5. In section 10 we show how this ensures that the larger of two competing sub-graphs comes to own an exponentially larger proportion of the network through inflation.

$$C = \sigma(\rho(T^T S - \kappa)) \tag{4}$$

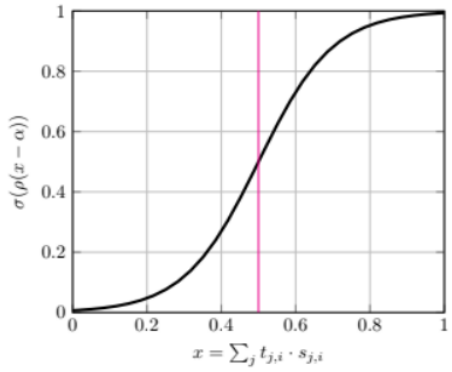


FIGURE 4 / Consensus function $c_i = \sigma(\rho \sum_j^n t_{j,i} s_j - \kappa)$ with temperature ρ = 10 and shift κ = 0.5. The activation takes the trust scores and produces an exponential scaling up to our inflection point where a peer is connected to the majority.

We use the consensus term to scale the original rankings. As peers attain more weight in the network they increase their inflation exponentially up to 0.5. In section 10 we show how this ensures that the larger of two competing sub-graphs comes to own an exponentially larger proportion of the network through inflation.

$$I = R \cdot C \tag{5}$$

This consensus described above protects against naive collusion by making it difficult for small groups to achieve inflation. However, it does not provide a incentive for correctly selecting weights. We introduce these incentives by adapting the inflation mechanism with a speculation based reward in the form of 'bonds' B. Here, $b_{i,j} \in B$ is the proportion of bonds owned by peer i in peer j.

$$B = \begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix} \tag{6}$$

Bonds accumulate at each step similarly to token inflation where $\Delta B = W \cdot S$ In this way, peers accumulate bonds in the peers they rank, thus 'bonding' themselves to those that they are connected to.

$$B_{t+1} = B_t + W \cdot S \tag{7}$$

Using the B bond matrix, the chain redistributes the normal incentive scores $\Delta S = B^T \cdot I$ Like market based speculation on traditional equities, the peers that have accumulated bonds in peers that others will later value attain increased inflation themselves. Thus it makes sense for peers to accumulate bonds in peers which it expects to do well according to other peers with stake in the system - thus speculating on their future value. Finally, we adapt this mechanism slightly to ensure peers attain a fixed proportion of their personal inflation. For instance, 50 percent, $\Delta S = 0.5B^T I + 0.5I$. ΔS becomes the mechanism step update which determines network incentives across the n peers.

$$S_{t+1} = S_t + \tau \Delta S \tag{8}$$

The incentive function in Section 2 rewards highly trusted peers, however, it may not solve the collusion problem if the honest nodes do not reach consensus. Notably loose, unused stake or incorrectly set weights will detract from the inflation proportion of honest peers in comparison to a colluding sub-network. The honest network, although holding more stake, may not gain enough inflation to overshadow its adversary. The dishonest sub-graph need only attain enough inflation to compete with its largest competitor, not to entirely dominate the network.

This attack is possible when the majority of token inflation is being distributed towards peers which are non-majority-trusted in the graph. The chain can measure this through a 'loss term' $\mathcal{L} = -R \cdot (C - 0.5)$ (Figure 7). The term is negative if the majority of inflation is being distributed towards peers with more than 0.5 consensus. The chain uses the loss calculation as a peg. By increasing the number of weights the average miner sets across the network the chain can ensure consensus.

FIGURE 5 / The left network has low consensus $\mathcal{L} > 0$ The system is not resistant to a cabal with less than 50 percent of the stake. The chain increases the number of edges set by peers until $\mathcal{L} < 0$. At this point the majority of inflation flows to peers with majority consensus.

05 RUNNING THE NETWORK

The steps to run a peer in the network are:

1. The peer defines its dataset D_i
2. At each training iteration, the peer conditionally broadcasts batches of examples from D_i to its peers x = [batch_size,sequence_length, input_size].
3. The responses $F(x) = [...f_j(x)...]$ – each of the common shape $f_j(x)$ = [batch_size, sequence_length, output_size] – are joined using the gating function and used as input to the local model f_i
4. Comparison against the target labels produces a loss-gradient $\frac{\partial \mathcal{L}}{\partial \mathcal{F}}$ which back-propagates through fi and out to the network
5. During 2 and 3 the peers learn the weights for their row $w_{i,j} \in W$ by measuring the value of the signals produced by their peers.
6. At distinct time-step t participants submit changes to the weights ΔW_i to update the ranking R, inflation I, consensus term C, and bond distributions δB
7. The chain measures 'loss' and optionally distributes newly minted stake into the network ΔS according to the bond ownership.

A common encoding of inputs and outputs is required for the various model types and input types to interact. The use of tensor modalities can be used to partition the network into disjoint graphs. At the beginning, the network can be seeded with a single modality TEXT, then expanded to include IMAGE, SPEECH, and TENSOR. Eventually, combinations of these modalities can be added; for instance TEXT-IMAGE, to bridge the network into the multi-modality landscape. Incentives to connect modalities can be integrated with the same trust scaling suggested in section (2). Eventually, successful models should accept inputs from any modality and process them into a useful representation. For consistency, we can use a standard output shape across the network [batch_size, sequence_dim, output_dim] similar to the common tensor-shapes produced by language and image models – and extend this size as the network increases in complexity.

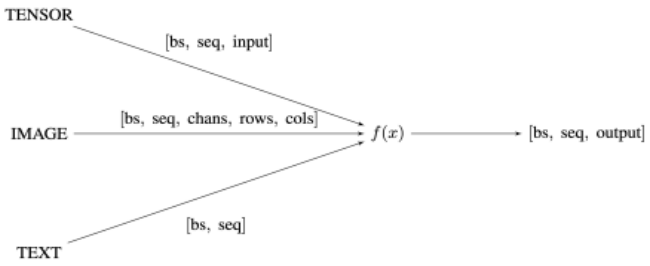


FIGURE 6 / Standardization of input dimensions within the network

By working on abstract input classes we can ensure participants work towards a general multi-task understanding Kaiser et al. [2017]. Participants may use: (2) completely distinct computing substrates Nugent and Molter [2014], (2) datasets Lample and Conneau [2019], (3) models, and (4) strategies for maximizing their incentives in the market. It makes sense for peers to work on unsupervised datasets where data is cheap and privacy not required.

As the network grows, outward bandwidth is likely to become a major bottleneck. The need to reduce network transfer and a method of selecting peers is required. Conditional computation can be used where peers learn through gradient descent how to select and prune neighbors in the network. For example, a product key layer or a sparsely gated layer Shazeer et al. [2017].

$$f_i = f_i(G(x)) \tag{9}$$

$$G(x) = \sum_j g_j(x) * f_j(x) \tag{10}$$

The conditional layer determines a sparse combination of peers to query for each example and multiplicatively re-joins them, cutting outward bandwidth by querying only a small subset of peers for each example. The method can drastically increase outward bandwidth Shazeer et al. [2017] Ryabinin and Gusev [2020], allowing peers to communicate with many more neighbors in the graph. In essence, the layer acts as a trainable DNS lookup for peers based on inputs. Furthermore, being trainable with respect to the loss, it provides a useful proxy for the weights $w_{i,j} \in W$

Dependence between functions ensures that models must stay online and cannot be run in production. Breaking this dependence can be achieved using distillation Hinton et al. [2015]: A compression and knowledge extraction technique in which a smaller model – the student - mimics the behavior of the remaining network. The distillation layer is employed in conjunction with a conditional computation (10) layer where the student model learns to mimic the network using the cross-entropy (shown below as KL) between the logits produced by the gating network and the student's predicted distribution Sanh et al. [2020].

$$distillation\ loss = KL_D(dist(x), G(x) \tag{11}$$

Because the distilled model acts as a proxy for the network, models can be fully taken off-line and evaluated. Recursion through the network is also cut between components allowing for arbitrary network graphs. If models go offline, their peers can use the distilled versions in-place. Private data 6 can be validated over the distilled models instead of querying the network. Eventually, components can fully disconnect from the network using the distilled models to do validation and inference offline.

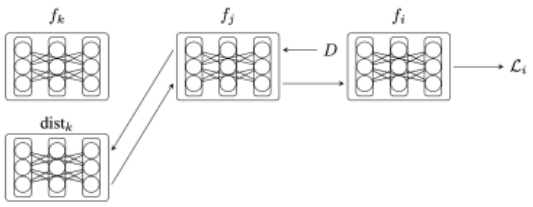


FIGURE 7 / Queries propagate to depth=1 before the distilled model is used.

Our goal in this work is the production of a ranking $r = [r_i]$ over peers where the score $r_i \in R$represents a participant's information-theoretic significance to the benchmark. Following LeCun and others LeCun et al. [1989]; Yu et al. [2017], it is reasonable to define this significance by equating it with the cost of removing each peer from the network. We can derive this score analytically where $\Delta F(x)_i$ is a perturbation of the j^{th} peers's inputs when the, i^{th} peer is removed from the network (Appendix 12.2):

$$r_i \approx \frac{1}{n} \sum_{j}^{n} \sum_{x \in D_j} \Delta F^T(x)_i \cdot H(Q_j(x)) \cdot \Delta F(x)_i$$
$$\Delta F(x)_i = [0, ...0, -f_i(x), 0, ...0] \tag{12}$$

Note, when the error function Q_j is the twice-differentiable cross-entropy, then $H(Q_j)$ is its Fisher- information matrix, and $r_i \in R$ is suitably measured as each peer's informational significance to the network as a whole. However, information theoretic weights require the full Hessian of the error. In practice it is more reasonable to use a heuristic to propagate a contribution score from the error function through to the inputs Yu et al. [2017]. For instance, weights from the gating layer (Section 6) provide a useful differentiable proxy.

We consider the scenario where a subset of the peers in the network have formed a 'cabal': A set of colluding peers attempting to maximize their inflation without accurately scoring their neighbors. The fight between the honest graph A with stake S_A and the disjoint cabal B with stake S_B can be determined by the proportion of network stake held by each. The honest graph must attain more inflation to maintain its dominance and protect the network $I_A \gg I_B$

We assume that the proportion of stake in the honest graph is more than that found in the dishonest graph $S_A \gg S_B$ and that the chain has reached consensus $\mathcal{L} < 0$ Since all peers in B are disjoint from A, our loss term $-R_B \cdot (C_B - 0.5) > 0$ is positive. Because $\mathcal{L} < 0$ it must be the case that $R_A \cdot (C_A - 0.5) < 0$ is negative and there are peers in the honest sub-graph A who are connected to the majority.

As the chain progresses, newly minted stake is being emitted at our inflation rate τ in proportion to I = R · T. Importantly, 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 graph. Notably, $\frac{\delta I}{\delta S} = \frac{5}{2}$, this ensures 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 exceeds that in sub-graph B, and sub-graph A's incentive grows super-linearly compared to B. The result is that the ratio of stake $\frac{S_B}{S_A + S_B}$ decreases – the cabal must continually add stake to its sub-graph to maintain itself through time.