将机器学习服务包装为blockchain上的一个节点，然后对服务进行收费。用户可以公开他们的模型或者数据供其他节点使用基于此方法可以做Pipeline或者DMoE，但是不保护数据的隐私，不适用于FL场景。已于23年3月发币TAO，现价值255$

Incentivizing Intelligence: The Bittensor Approach

Jacob Steeves[1*] **Ala Shaabana**[1*] **Yuqian Hu**[1] **Francois Luus**[1]

Sin Tai Liu[1] **Jacqueline Dawn Tasker-Steeves**[1]

[1]Opentensor Foundation

Abstract

Inspired by the efficiency of financial markets, we propose that a market system can be used to effectively produce machine intelligence. This paper introduces a mechanism in which machine intelligence is valued by other intelligence systems peer-to-peer across the internet. Peers rank each other by training neural networks that are able to learn the value of their neighbours, while scores accumulate on a digital ledger. High-ranking peers are rewarded with additional weight in the network. In addition, the network features an incentive mechanism designed to resist collusion. The result is a collectively run machine intelligence market that continually produces newly trained models and rewards participants who contribute information-theoretic value to the system.

1 Introduction

The production of machine intelligence relies almost exclusively on a system of benchmarking, in which a model's performance is assessed on narrowly defined supervised problems. While this method works well for task-specific problems, measuring machine intelligence solely with supervised objectives causes the field to converge towards narrow specialists, as opposed to resilient generalists (Radford et al. [2019]).

In 2012, an inflection point occurred when AlexNet achieved state-of-the-art performance on the ImageNet competition (Krizhevsky et al. [2017]). This triggered a sharp acceleration in the development of new neural network approaches able to overcome purely statistical methods with high dimensional, representation-based approaches. Only a few years later, Vaswani *et al.* introduced the first-ever unsupervised learning method that would surpass supervised learning regimes in language understanding (Vaswani et al. [2017]). These innovations demonstrate that researchers can achieve state of the art results by making consistent, incremental improvements upon previous work, consistently adding to a shared base of research over time.

However, since intelligence produced by these models is always lost, this approach is quite inefficient. Users have to retrain models on their own systems to replicate, or improve upon, the work of others. Consequently, this leads to unnecessary computational loss in learning tasks that other models have already learned.

Additionally, there is the issue of model evaluation. Presently, new research must go through the academic peer review process to investigate the work while ensuring quality and correctness. While this process has its benefits, it creates a system that is unscalable and subject to human bias. It is notoriously difficult to reproduce artificial intelligence research Gundersen and Kjensmo [2018]. There must be a more efficient and objective method to evaluate AI model performance.

In this paper, we introduce the Bittensor Network: a decentralized peer-to-peer machine learning protocol. In the network, machine intelligence is measured by other intelligence systems in a continuous and asynchronous peer-to-peer (P2P) fashion across the internet. Models are ranked for informational production regardless of the subjective task or dataset used to train them. By changing

2022 Trustworthy and Socially Responsible Machine Learning (TSRML 2022) co-located with NeurIPS 2022.

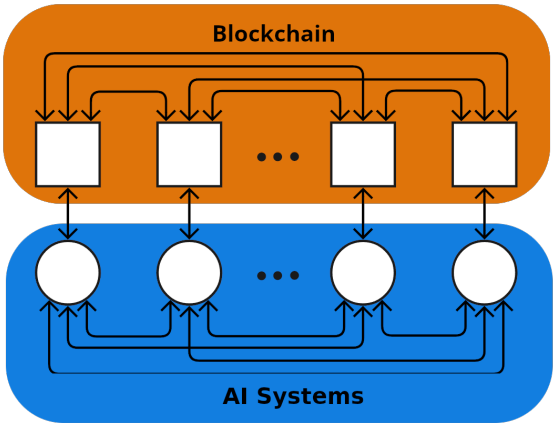


Figure 1: Schematic of the Bittensor Network

the basis against which machine intelligence is measured, the market can reward intelligence that is applicable to a much larger set of objectives while still preserving the unique value of legacy systems and smaller diverse systems that find niches within the reward landscape. The constructed market uses a digital ledger to record ranks and to provide incentives to the researchers, thus allowing them to directly monetize their machine intelligence work. It is divided into two layers: the AI layer and the Blockchain layer (Figure 1).

2 AI layer

In addition to inference and training, the AI layer is responsible for abstracting away the Bittensor kernel and ensuring input/output compatibility of a node's neural network with the rest of the network's peers. Each node on the Bittensor Protocol contains a single neural network. From now on, we will be referring to these nodes as neurons; this is not to be confused with individual neurons in traditional neural networks.

The Bittensor Protocol is composed of n parameterized functions (or neurons) $F = f_0, ..., f_j, ...f_n$ distributed in a peer-to-peer fashion. Each neuron is holding zero or more network weights $\mathbf{S} = [s_i]$ (or "stake") represented on a digital ledger. Our goal is the distribution of stake I – as incentive – to peers who have contributed the most informational value.

Figure 2 shows a high level overview of the forward and backward passes performed on the Bittensor network and describes the process by which neurons train on the Bittensor protocol. Similarly to the Hivemind proposal by Ryabinin *et al.*, peers are able to request others for forward passes and subsequently propagate backward gradients over the wire. By querying for information, Bittensor offers *inference* across all of the nodes serving on the network. Unique to Bittensor, this enables an distributed trust-less style of machine learning where requests are authenticated and verified using the digital ledger. The Blockchain layer ensures the identity of each peer through a unique cryptographic key and maintains the integrity of incentive distribution and rankings over the entire network.

In Bittensor, peers who are highly ranked receive the most incentive. 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 $\mathbf{W} = [w_{i,j}]$ where peer i is responsible for setting the i^{th} row through transactions on a digital ledger. We describe peer ranking in more detail in Section 4.

Figure 3 shows a cutaway diagram of two neurons on the Bittensor network. Synapses function as the main transmitter between neurons and ensure all of the transmission tensors are formatted in correspondence to the Bittensor protocol. Each synapse dictates a different ML task, and therefore follows a different compression and communication process. This ensures that all logits and embeddings being transported over the Bittensor protocol are sanitized, correct, and contribute useful information. If a neuron is acting maliciously or sending incorrect tensors across the wire, Synapses ensure that their rankings and rewards are penalized as a result. Currently, the Bittensor network is focused on large language modelling and contains 4 separate synapses for text-related tasks (*LastHiddenState*,

类似DMoE

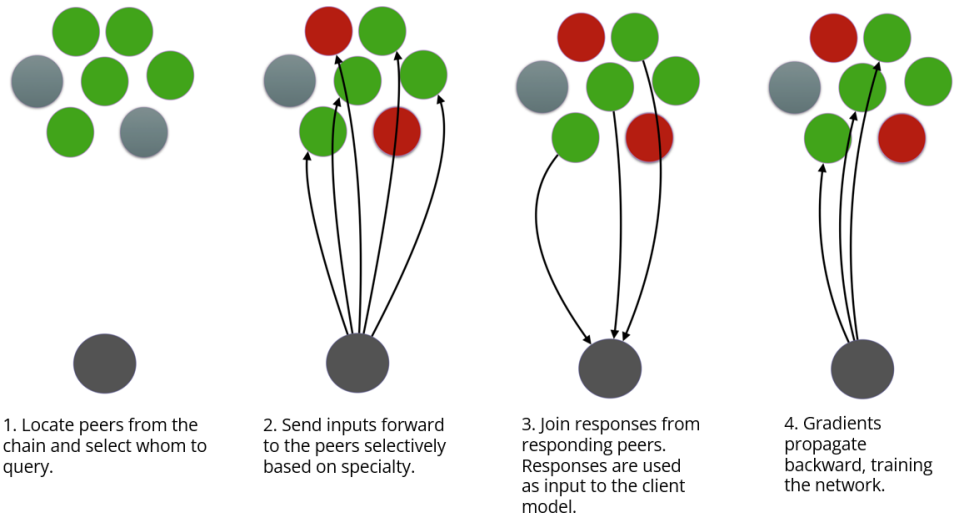
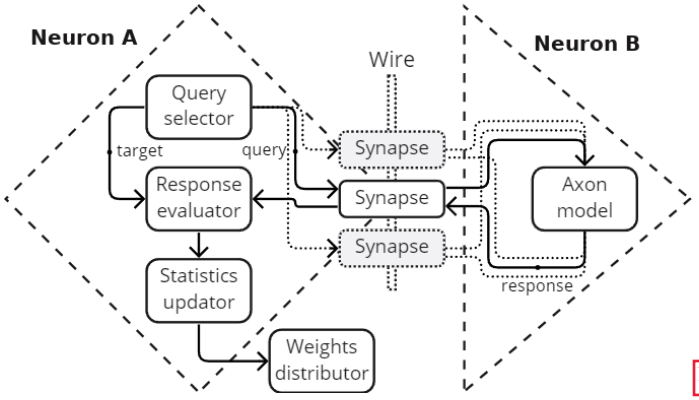


Figure 2: Neurons training on the Bittensor Protocol. The Bittensor protocol utilizes a mixture-of-experts (MoE) architecture with each remote peer acting as a single expert(Shazeer et al. [2017]; Ryabinin and Gusev [2020])



Pipeline Reuse!

Figure 3: Schematic diagram of a sample neuron on the Bittensor network. Models query each other over the wire and use the response as part of its own model. Responses are then evaluated for their informational value and ranked based on their contribution to the collective. Responses and gradients can travel over the wire as the network trains.

CausalLM, *CausalLMNext*, and *Seq2Seq*). Prior works have shown that large language models can be continually fine-tuned for improved performance for downstream tasks (Scialom et al. [2022],Ouyang et al. [2022]). The Bittensor protocol achieves the same continual fine-tuning process while being fully distributed and trust-less over the web.

3 Blockchain Layer

The Blockchain layer is a layer-0 blockchain based on Polkadot Substrate (Wood [2016]), and is responsible for enforcing the consensus mechanism, ensuring peer identity, and incentivizing network peers. The blockchain layer resides directly underneath the AI layer, and communication between the two layers is achieved through inter-process communication. In order to fairly distribute incentive across all of the participating peers, the Bittensor network leverages a staked weighted trust through consensus. Peers are incentivized to rank each other, and highly ranked peers will receive additional rewards. The blockchain does not trust rankings from any individual peer on the network, but rather trusts the collective rankings from all of the participating peers. In order to submit their rankings, peers must first register their wallets, and a pair of cryptographic keys that are unique to each peer.

The same keys will be required to sign all transactions to the chain and all communication requests between peers. These cryptographic keys will serve as the main identification tool for peers on the network.

Additionally, the blockchain maintains the system's incentive mechanism, designed primarily for peer-rankings systems and to solve for the problem of collusion, which occurs when a small fraction of the network works selfishly together as a means to maximize their rewards.

To solve this problem, we have leveraged a trust based incentive mechanism by rewarding peers in the network who are "trusted" and have reached consensus. The incentive mechanism utilizes a stake vector $\mathbf{S}$ and a set of weights $\mathbf{W}$ where rows are inter-peer rankings. We also infer a trust matrix $\mathbf{T}$ from the weights, where $t_{i,j} = 1$ if and only if there is a non-zero edge between peer i and j.

$$\mathbf{W} = \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} & w_{0,3} \\ w_{1,0} & w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,0} & w_{2,1} & w_{2,2} & w_{2,3} \\ w_{3,0} & w_{3,1} & w_{3,2} & w_{3,3} \end{bmatrix} \quad \mathbf{S} = \begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} \quad \mathbf{T} = \begin{bmatrix} t_{0,0} & t_{0,1} & t_{0,2} & t_{0,3} \\ t_{1,0} & t_{1,1} & t_{1,2} & t_{1,3} \\ t_{2,0} & t_{2,1} & t_{2,2} & t_{2,3} \\ t_{3,0} & t_{3,1} & t_{3,2} & t_{3,3} \end{bmatrix} \tag{1}$$

We define peers who have reached "consensus" as those with a positive weight settings from more than 50 percent of stake ($T > 0.5$) in the network. Hence, when more than 50% of the incentive goes to consensus peers, then the chain can be considered as having reached consensus. This is regularized on the chain by the sigmoid function defined by Equation 2 such that peers who have reached consensus receive significantly higher rewards. Effectively, the threshold-like sigmoid rewards connected peers and punishes the untrustworthy peers.

$$\mathbf{C} = \sigma((\mathbf{T}^T\mathbf{S} - 0.5)) \tag{2}$$

We use the consensus term to scale the original rankings. This ensures the majority of the incentive is distributed to peers that trusted by a majority of the network.

$$\mathbf{I} = \mathbf{R} \cdot \mathbf{C} \tag{3}$$

To give an example of how this might work in practice, consider a scenario in which a colluding cabal C has formed on the network Figure 4. The cabal will actively try to maximize their inflation by only ranking themselves highly while providing no informational knowledge for the honest H peers on the network. In this disconnected network, the two subgroups will only vote for peers in its own group with little to no weights for peers of the opposite group (Figure 4). Assuming that the chain has reached consensus, the fight between honest H peers and cabal C peers can be determined by the portion of initial stake each sub-group contains.

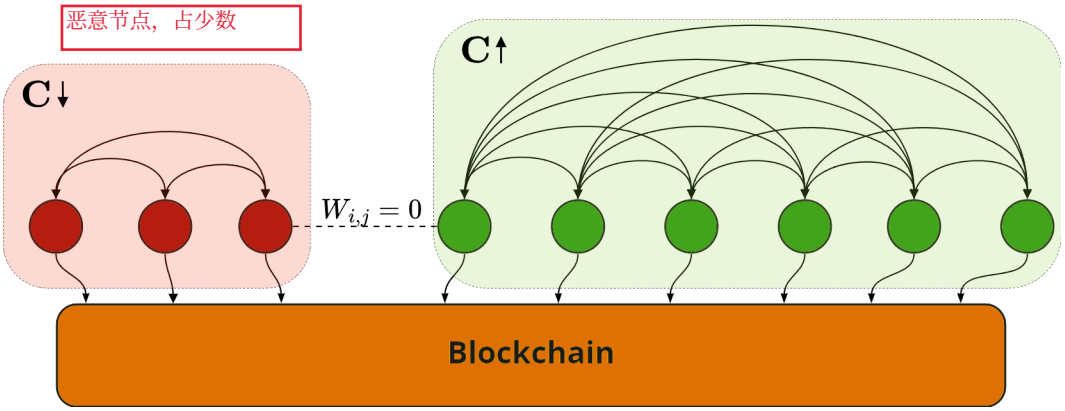


Figure 4: Disjointed sub-networks: Cabal peers (left) only vote within its own group and does not interact with a majority of the peers and stake on the network. This results a lower consensus $\mathbf{C}$ score for the sub-network and thus a lower incentive.

In the event that the stake in the honest sub-graph is greater than the cabal ($S_H > S_C$), only peers within the honest subgroup are capable of reaching consensus and will receive the majority of the incentive from the chain. The disagreement between the majority stake holders S_H, and the dishonest sub-graph results in a penalized consensus and therefore incentive for the C sub-graph. As such, the ratio of stake $\frac{S_C}{S_C+S_H}$ decreases over time, and the cabal must continually add stake to maintain its position. These cabal attacks are akin to the 51% attacks present in other blockchains (Sayeed and Marco-Gisbert [2019]). However, our stake based ranking system – like other proof of stake systems – present significantly higher risks for the attackers that endanger the value of their own tokens.

Additionally, our incentive mechanism includes a bonding mechanism (Appendix Section 9) to provide additional incentives for peers who validate honestly. Similar to market-based speculation in traditional equities, the peers that have accumulated bonds in peers that others will later value attain increased inflation themselves.

4 Game theoretic scoring

A crucial element of the Bittensor Protocol is the production of an accurate ranking method. Peers are ranked based on their informational significance to the overall collective, represented by a ranking $\mathbf{r} = [r_i]$ where score $r_i \in \mathbf{R}$ represents a participant's contribution to the benchmark. This is done by calculating its Shapley value ϕ (Shapley [1952]) which is a way to fairly distribute the contribution for each participant in a group by permuting through all possible combinations of the subsets:

$$\phi_i = \frac{1}{|N|!} \sum_{S \subseteq N \setminus \{i\}} |S|!(|N| - |S| - 1)![CE(S \cup i) - CE(S)] \tag{4}$$

where N and S represent the complete set and a single set of peers respectively. $CE(S)$ is the cross-entropy loss give S set of peers, ϕ_i represents the Shapley value for a peer i.

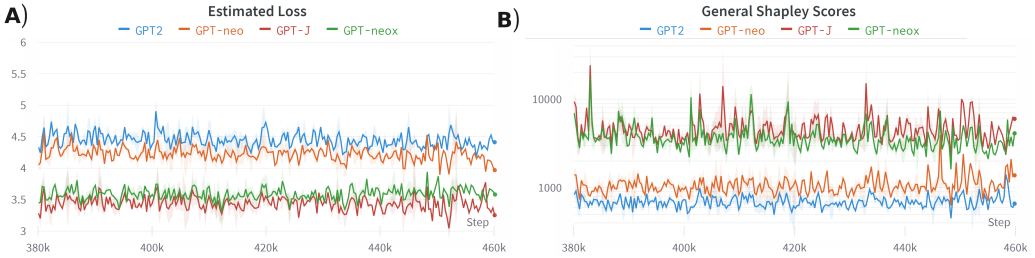


Figure 5: Shapley score comparison between large language models (LLM) of various sizes (GPT 2 — 117M, GPT-neo — 125M, GPT-J — 6B, GPT-neox — 30B) (Wang and Komatsuzaki [2021] , Black et al. [2022]) by a single validator. The scoring was done on the Bittensor network, and models were scored based on their performance at last token prediction. Larger models capable of achieving lower loss are given a higher Shapley score by their peer.

In order to calculate the Shapley score for thousands of peers, the network leverages a sparse Monte Carlo estimation method. In terms of the AI layer, this is akin to a sparsely gated mixture-of-experts (MoE) layer (Shazeer et al. [2017], Fedus et al. [2021]) such that only a portion of the network is queried at a given time. This was implemented to allow the network to scale efficiently without facing any computational bottlenecks.

$$G(x) = \sum_j g_j(x) * f_j(x) \tag{5}$$

where $g_j(x)$ represents the output of the gating network and $f_j(x)$ represents the response from peer, j. Unlike previous MoE models where routing determines the parameters to activate, our gating layer determines the optimal peers in the network to query for each example. Responses are then used as the inputs as part of a local model and evaluated for their contribution. This cuts outward bandwidth

by querying only a small subset of peers for each example and thus also reducing the number of possible permutations in our Shapley calculation.

The sparse combinations of peers effectively represent a subset of the possible permutations within the complete set of N peers. Given enough samples, we can recover the full Shapley values and a cooperative ranking $\mathbf{r}$ for the entire network. Benchmarks of several LLMs currently running on the Bittensor network are showcased in Figure 5.

5 Current State of The Bittensor Network

The Bittensor network officially launched in November 2021. As of writing this paper, the network contains 4096 peers; all of which are running a language model that is actively querying others for information or serving requests from other peers on the network. With a total sum of 500 Billion parameters running across 4096 peers, it is possible to query and infer from every node in the network and receive an output of logits, embeddings, or plain text responses given a prompt. Each peer is either running a custom language model, or a pre-trained model such as GPT-J (Wang and Komatsuzaki [2021]), GPT-Neo, (Black et al. [2021]), or GPT-2 (Radford et al. [2019]) downloaded from HuggingFace API (Wolf et al. [2020]).

In 2022, the protocol underwent several algorithmic, consensus, and infrastructure changes to ensure correctness and scalability. The Bittensor team is now training more sophisticated models directly on the Bittensor protocol to work towards achieving performance comparable to current state of the art models.

6 Conclusion

We have introduced the first peer-to-peer, trust-less intelligence market that features a novel ranking benchmark. The benchmark measures performance as representational-knowledge production using other intelligence systems to determine its value. Due to its high-resolution, highly collaborative nature, this benchmark has the potential to be the gold standard in machine learning testing and evaluation.

The paper began with the definition of the Bittensor Protocol, a P2P network composed of abstractly defined intelligence models. Peers are able to directly query each other for informational embeddings that serve as inputs to their own models. Training can occur by subsequently propagating backward gradients to responsive peers. We then showed how this framework allows us to produce a cooperative game-theoretic ranking for each peer based on its informational value for the network. These scores are passed on to a digital ledger where a trust-based incentivization is distributed. Peers that earn high scores from a majority of stake holders will receive the majority of the incentive. We then demonstrated how this incentive scheme based on peer connectivity prevents participants from forming dishonest sub-graphs such that over time, dishonest sub-graphs decay into irrelevance.

The result of these various novel mechanisms is an intelligence market that rewards participants for producing knowledge while enabling the sharing of this knowledge for new learners in the system. Similar to financial and other Web3 markets, the market for intelligence actively encourages innovations among its participants and rewards for those who manage to provide unique knowledge to the collective. In terms of ML, Bittensor provides incentives for participants to consistently improve and finetune their models over the network. This combination of Web3 and ML introduces an unique network where a decentralized global contribution of compute can be put towards the advancement of machine intelligence.

References

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.

A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

O. E. Gundersen and S. Kjensmo, "State of the art: Reproducibility in artificial intelligence," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

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," 2017.

M. Ryabinin and A. Gusev, "Towards crowdsourced training of large neural networks using decentralized mixture-of-experts," 2020.

T. Scialom, T. Chakrabarty, and S. Muresan, "Fine-tuned language models are continual learners," 2022. [Online]. Available: https://arxiv.org/abs/2205.12393

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe, "Training language models to follow instructions with human feedback," 2022. [Online]. Available: https://arxiv.org/abs/2203.02155

G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White Paper*, vol. 21, pp. 2327–4662, 2016.

S. Sayeed and H. Marco-Gisbert, "Assessing blockchain consensus and security mechanisms against the 512019. [Online]. Available: https://www.mdpi.com/2076-3417/9/9/1788

L. S. Shapley, *A Value for N-Person Games*. Santa Monica, CA: RAND Corporation, 1952.

B. Wang and A. Komatsuzaki, "GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model," https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. McDonell, J. Phang, M. Pieler, U. S. Prashanth, S. Purohit, L. Reynolds, J. Tow, B. Wang, and S. Weinbach, "Gpt-neox-20b: An open-source autoregressive language model," 2022. [Online]. Available: https://arxiv.org/abs/2204.06745

W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," 2021. [Online]. Available: https://arxiv.org/abs/2101.03961

S. Black, G. Leo, P. Wang, C. Leahy, and S. Biderman, "GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow," Mar. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5297715

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

7 Appendix

8 Sample Rankings

	Model Parameters	Bittensor Ranking
gpt-2	117M	0.0000106
gpt-neo	125M	0.0001242
bloom	3B	0.0002789
OPT	6.7B	0.0004604
gpt-J	6B	0.0005235

Table 1: Chain wide ranking comparison of various large language models currently being served on the Bittensor network. The final rankings are computed based on the Bittensor protocol where individual peer rankings are submitted to the digital ledger. These rankings are normalized such that the total rankings of the 4096 peers sum to 1.

9 Bonds

Although the mechanism described in section 3 protects against collusion by making it difficult for small groups to achieve inflation, it does not provide an incentive for correctly selecting weights. We introduce these incentives by adapting the inflation mechanism with a speculation-based reward in the form of bonds $\mathbf{B}$. Here, $b_{i,j} \in \mathbf{B}$ is the proportion of bonds owned by peer i in peer j.

$$\mathbf{B}_{t+1} = \mathbf{B}_t + \mathbf{W} \cdot \mathbf{S} \tag{6}$$

Using the bond matrix $\mathbf{B}$, the chain redistributes the normal incentive scores $\Delta S = \mathbf{B}^T \cdot \mathbf{I}$. Like market-based speculation on traditional equities, peers that have accumulated bonds in other peers that others will later value, will attain increased inflation themselves. Thus if a group of peers with stake in the system are accumulating bonds on a given peer that they expect to do well, then other peers are motivated to accumulate bonds as well to this same peer – 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, $\Delta S = 0.5\mathbf{B}^T\mathbf{I} + 0.5\mathbf{I}$. ΔS becomes the mechanism step update with determines network incentives across the n peers.

$$\mathbf{S}_{t+1} = \mathbf{S}_t + \tau\Delta S \tag{7}$$

10 Anatomy

10.1 Axons

Axons represent the server of a model that a registered account can server. The axon represents a gRPC server and uses thread pools for handling multiple requests from dendrites (clients). The requests from clients are placed in a priority queue . The priority score can bed defined based on the server's purpose. The server connects with the client via Universal Plug and Play protocol.

10.2 Dendrites

Receptors serve as client thread-pools when querying another server with inputs. The receptor uses multi-threading pool executors for managing multiple IO bound connections with each of the server neurons. Requests are signed by the client's hotkey to ensure the account is registered with the Bittensor network. This excludes accounts that are not registered to the Bittensor network.

10.3 Synapse

A synapse is a schema of communication between a neurons. Synapse schemes ensure the client and server are agreeing on their messages, which is involved both for forward and backward pass messages. At the moment, there consists of the following Synapses: *LastHiddenState*, *CausalLM*, *CausalLMNext*, and *Seq2Seq*.

10.3.1 LastHiddenState

The *LastHiddenState* synapse is designed for communication of embeddings over the wire. A peer may request for this synapse when requesting for a forward pass from another peer. The peer is then expected to process the inputs and return its last hidden state before decoding over the wire.

10.3.2 CausalLM, CausalLMNext

Similar to the *LastHiddenState* synapse, both the *CausalLM* and *CausalLMNext* synapse is responsible for the transport of logits over the wire. With the *CausalLM* synapse, peers will respond with the standard next token prediction over all of the inputs, while with *CausalLMNext* synapse demand the predictions for the next token following the input text.

10.3.3 Seq2Seq

With the *Seq2Seq* synapse, peers query other peers with the expectation of text generation given a input prompt. Unlike the other synapses which are often used for evaluation of individual peers, the *Seq2Seq* synapse is mainly used for extracting value out of the peers currently serving on the Bittensor network.