Modelex.AI Litepaper

V0.01 DRAFT

Jamiel Sheikh
jamiel@scifn.com
https://linkedin.com/in/jamiel
https://modelex.ai

Objective: A world-class, enterprise-grade, global, active decentralized network and registry of coin-operated Modelex nodes that wrap language-model-based AI agents aka the *Model Exchange*.

Language models, like GPT4, are able to process large amounts of text to produce probabilistic completions for prompts or inputs. Introduced to the public via chatbot interfaces, language models (LM) have made artificial intelligence more accessible to the broader public and created new excitement about artificial intelligence.

Creating a language model, especially if they are large, requires a significant amount of capital investment. Millions of dollars can quickly be spent on procuring text content, training, and hosting these models, often with uncertain results, only to potentially be surpassed by a competitor months later.. As a result, the effectiveness of these models for a given unit of spend becomes important. Small-language models (SLM) could offer lower cost with almost as effective performance as large-language models (LLMs) and these models now challenge many incumbent text analyzing engines like proprietary sentiment algorithms that are commonly licensed in capital markets.

The number of LMs coming to market increases weekly, adding to the pile of LMs that already exist. General-purpose vs. domain-specific, pre-trained or fine-tuned, large or small, public or private are some of the flavors of models on the market. BLOOM, Falcon, Mistral, Claude, Yi, GPT, LLAMA, OpenELM are general purpose language models and domain-specific ones like BloombergGPT[1], FinLLAMA, FinBERT, FinGPT and derivative fine-tuned models represent a growing and competing set of model options.

In order to run or inference a model, significant compute resources are often required. In addition, software infrastructure and tooling, such as retrieval augmented generation (RAG) which injects real-time curated content prior to a prompt being submitted to the language model, adds complexity to the architecture. Meta's Research SuperCluster (RSC) is one example of compute resources, hosting tens of thousands of GPUs with petabytes of cache storage, a centralized powerhouse built to benefit Meta's AI research and business objectives.

Hugging Face is a centralized indexed registry of public open-source AI models, including language models. In order to use a model on Hugging Face, technical skills are required to be able to select the right model and deploy on to compute resources. Hugging Face makes it relatively easy for most

[1] https://www.forbes.com/sites/jamielsheikh/2023/04/05/the-chatgpt-of-finance-is-here-bloomberg-is-combining-ai-and-fintech/?sh=5cb66efe3081

developers to deploy AI models on centralized compute resources. Hugging Face inventories over 350,000 models.

The demand for AI model inventorying, hosting and compute resources is significant and growing and a number of centralized entities have been hugely successful in providing these services, Hugging Faces alone raising $235m in its Series D, for example.

The equivalent in a decentralized network is envisioned.

The network

Modelex envisions a world where AI models or agents can run in a decentralized manner, whether the underlying compute substrate is centralized or decentralized. Modelex is a decentralized platform for running, operating and monetizing AI agents that employ language models and the currency used to pay for the use of these models is a utility token.

Modelex is a network of nodes, where each node wraps one or more language models (and in the future, other types of AI models or agents). The nodes are blockchain-centric. For example, the services of a node are paid for using the $AI token and a decentralized registry of nodes is available. Nodes can act like oracles placing the completion of a prompt on-chain. An example of such use case is sentiment and intent scores, where a Modelex node ingests, either regularly or at intervals and either for a rolling window or a current snapshot, news and filings about some asset and produces a single numerical score available to a smart contract to potential store on-chain.

The score may be used as a factor in interest-rate setting or evaluation of collateral and guide pricing. A number of papers have been written on using language models for sentiment and the papers suggest the possibility to derive significant alpha using LM-based sentiment analysis.

Modelex is not restricted to sentiment, although it is the first service it will provide on top of the Modelex platform. The goal of Modelex is to make deployment and monetization of AI agents a) decentralized and b) easy and then make available the results of these services to centralized or decentralized entities, whether it is to an enterprise via an API or decentralized smart contracts.

The node

A Modelex node encapsulates a language model and it views the model as a workhorse and not a chat counterparty. As a workhorse, a language model is constantly ingesting text and producing completions, acting in the background and does not need to be prompted by a human. For example, a schedule pushes 90 days of news for an asset every day and the language model spits out a sentiment score. The notion of “chat” no longer exists as the language model is just a worker bee.

Modelex nodes are tightly tethered to one or more blockchains, act as proxies or wrappers for AI agents and can act as an oracle. For example, on Ethereum, a Modelex node proves ownership of one or more Ethereum accounts that it can accept payment and oracle services, often coupled using ERC677.

A user can communicate with a node, whether it is via an event from an EVM smart contract or an API, supply a prompt, whether from a template library or customized, supplement with payment and get back a digitally signed response generated by the language model embedded in the node.

Thousands of such nodes can exist, allowing a large number of language models to flourish and further democratizing AI and access to GPUs. If you have a GPU and wish to spin up a Modelex node and run a language model, you can earn fees for doing so. There is no notion of mining.

From a technical point of view, the Modelex node is written in high-performance Rust and wraps a Python executable that runs inference. At some point in the future, we expect the node to be entirely Rust-based as the Rust and language model worlds continue to converge as it is essential to keep the node running in a single executable process, limiting attack vectors.

The registry

Nodes also register their existence into a decentralized registry. The registry is divided into two sections - Public anonymous nodes that can be operated by anyone, potentially Byzantine, and whitelisted nodes, where we act as a centralized authorizing party (with a DAO in the future taking over this role), for example a node managed by a reputable enterprise. Much like how oracles in the Chainlink network can be rated, Modelex nodes can be evaluated by independent parties for their accuracy, response and uptime, etc.

The token

The fixed supply of 1 billion tokens with 18 decimal places is envisioned to be sufficient supply for a global network. If needed, tokens can be split, akin to a 2:1 stock split. In this approach, all transactions with the ERC-20 are paused for a certain number of blocks and everyone's holdings and the total supply max are doubled per split event. The following token allocation is envisioned, with the founders, team and advisors having a lock-up period.

	Allocation	**Amount**
Team		
Accredited Investors & Institutions	25%	250,000,000
Founders, Team & Advisors	15%	150,000,000
Network Users	60%	600,000,000
	100%	1,000,000,000

Pricing

While each node can set its price for its services, language models typically charge per language token[2]. In the case of GPT, the latest prices as of this writing is listed below. Whether nodes need to compete with these prices is left to the market to determine, but aggregate transaction size and volume can be assumed to be significant as language models are integrated into business processes.

GPT4 Pricing				
	Per 1K		**Per 1M**	
Context Length	*Prompt Tokens*	*Completion Tokens*	*Prompt Tokens*	*Completion Tokens*
128k	$0.01	$0.03	$10.00	$30.00
8k	$0.03	$0.06	$30.00	$60.00
32k	$0.06	$0.12	$60.00	$120.00

Addressable Market

It is believed that the addressable market is larger than Chainlink's, and/or Render's and/or Hugging Face's, even potentially combined. The precise TAM is not easy to calculate and is not presented here. The objective is for Modelex to become the standard decentralized platform for deploying, running and monetizing AI agents and what we do know is that generative AI is projected to be a $1.3T market by 2031 with a 42% CAGR[3].

Tooling

To supplement the Modelex network and Modelex node, a set of tools and features would be available, some of which may be centralized and licensed to enterprises under traditional SaaS approach. The purpose of these tools would be to further drive adoption of the decentralized network, making adoption more seamlessly and pain-free, allow for deeper integration and better user experience, which includes customer support.

Prompt Library

This includes parameterized prompts where the size of the completion is known, which is essential in assessing inference and completion costs. An example of a parameterized prompt that evaluates the sentiment of a news headline with a restricted single digit quantitative response is described in the whitepaper. Prompt templates can be stored on decentralized or centralized storage systems and we leave it to the creators of such templates to determine the storage. Modelex will create a library of such template prompts for use but users are free to create and share their own.

---

[2] To avoid any confusion, we refer to the tokenization of prompts and completion responses as language tokens (a superset of prompt tokens or sampled tokens), not to be confused with the $AI payment token

[3] https://www.bloomberg.com/company/press/generative-ai-to-become-a-1-3-trillion-market-by-2032-research-finds/

Data Blocks

The process of taking a stock language model and adding to it additional document awareness is known as fine-tuning. For example, FinLLAMA is a fine-tuned version of Meta's LLAMA language model, added to it are financial documents making FinLLAMA more adept at differentiating between *bull* and *bear* in the context of finance rather than in the context of a zoo.

A large number of enterprises sit on large swathes of proprietary data. That data can be canonicalized into blocks and traded or leased, even without disclosing directly their content. For example, I may wish to take a base language model like LLAMA and fine-tune it with 10,000 PDFs from the IMF. Currently, such a single monolithic dataset of 10K paper does not exist. It would behoove an organization like the IMF, or any other organization with a similar predicament, to convert the research papers into a single canonical data block and make it available for lease for other to fine-tune, or even better, deliver to an enterprise a fine-tuned version already without revealing the full dataset.

Such data blocks can be stored in private repositories or decentralized repositories and a Modelex node owner can select a number of these data blocks to build a very customized fine-tuned language model, paying or "leasing" the creators of the data blocks with $AI. Creators earn fees for data blocks they create.

Oracle

While a language model natively has no relationship with a blockchain, a Modelex node wraps around the model and is the proxy or container that deeply integrates with a blockchain (initially, EVM only). The integration between a Modelex node and a blockchain is so tight that the node may be considered layer 2 functionality. At this stage, we are not referring to the node as layer 2.

Asynchronous Prompt Completion

While it is ideal for language models to run on GPUs, it is entirely possible to run them on CPUs with severe performance penalties. Regardless, in use cases where the completion of a prompt is not required immediately, the submission of a prompt can be a job where the results are available at some time in the future. For example, obtaining the sentiment on the sovereign debt can take 24 hours and would be an acceptable response time. In such a case, a Modelex node can optimize and charge accordingly and potentially leverage cheaper (older generation of GPUs or just CPUs) compute resources.

Enterprise Dashboard and Node Console

Two types of consoles would be available. One that can be used by consumers of Modelex services and are looking for a centralized place to interact with the network, akin to the Uniswap frontend to their DEX. Access to the DEX can occur without the frontend but it is useful for users that wish to have a consistent

and reliable UX. Second, each Modelex node will run embedded in it an HTTP server that allows for GraphQL-based metrics access and an SSH-based console to manage the node.

More Features

A number of additional features are envisioned to be required eventually. These include a way to load balance and parallelize across nodes (effectively creating virtual or logical farms), wrapping other types of AI models, having the ability to upgrade language models automatically, enterprise integration and tools and more. Our goal is to listen to the market to find emerging opportunities that help the feature set to grow and capture more market share. In addition, being able to run nodes with a trusted execution environment is of critical importance and some of the early research will be devoted to this capability.

Proof of Concept smart contracts & screen shots

An initial set of smart contracts have been deployed to the Sepolia testnet for further research. Development of the Modelex node has already begun with a console partly built. All is subject to change and should be used as indication of direction of this project.

Smart Contract	**Contract Address**
$AI Token ERC20/ERC677	0xf6D6Af58aFEEA44dC435De72b3b52c06f3f48aBC
Node Registry	0x1dB9c7C21ac2850CDdFb5213A780b1409f2F669c

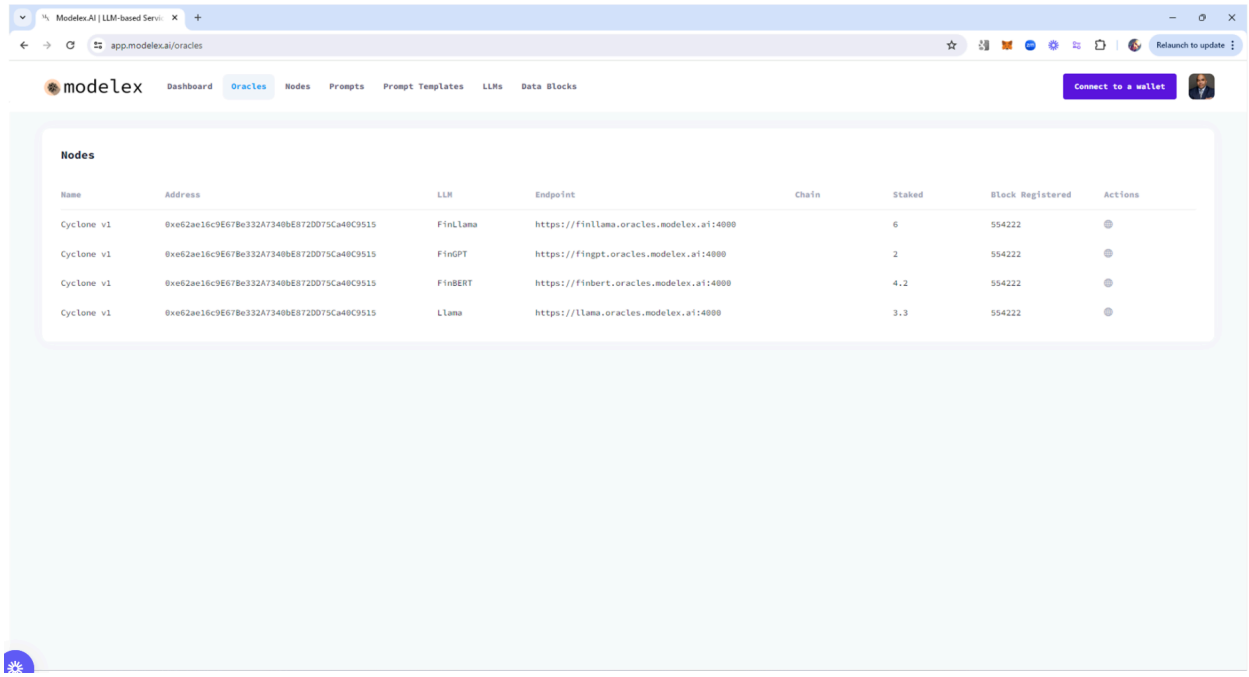

Figure 1: Registry of nodes

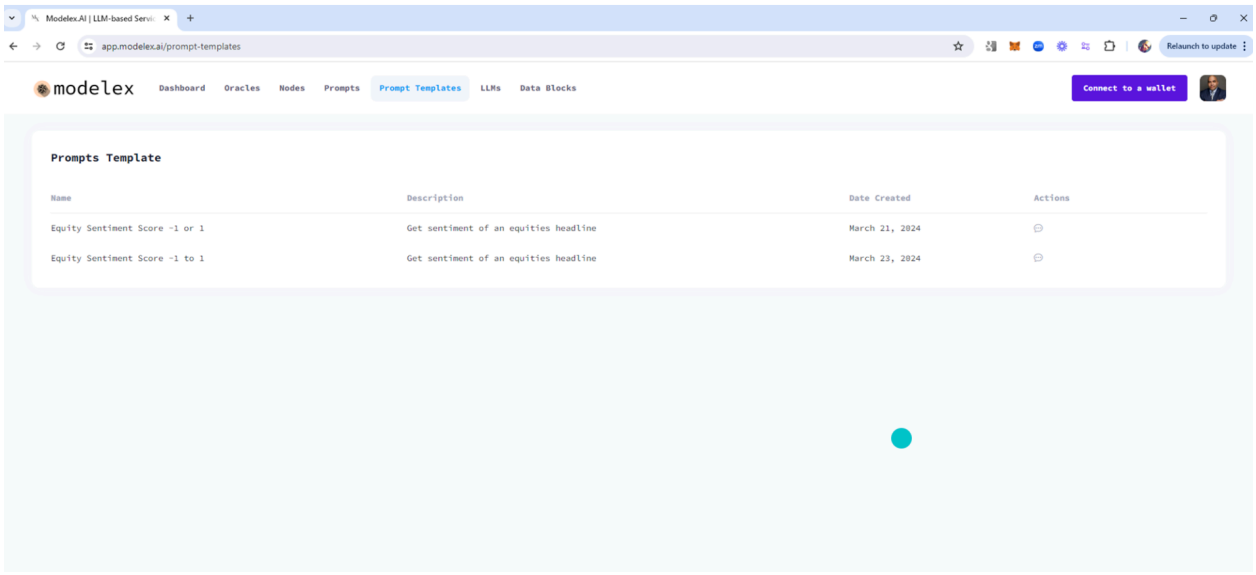

Figure 2: Prompt Library