GENESISAI

Genesis AI Protocol

Protocol for Machine Learning Networking

Archil Cheishvili
archil@genesisai.io

David Fan
david@genesisai.io

GenesisAI is a machine learning protocol. On top of this machine learning protocol we are building a marketplace for AI products and services. Simply put, it is Amazon for AI models. The GenesisAI marketplace connects companies in need of AI services with companies interested in monetizing their AI technology. GenesisAI brings together AI technologies from all over the world to help lay the foundation for the eventual global Artificial General Intelligence.

Nowadays there is no working protocol for ML networking. Because it does not exist yet, engineers spend many hours manually linking ML models. Companies and developers reinvent the wheel. It takes lots of money and hours to make AI products and services.

We make it easy for service providers to link together machine learning models so that they can add functionality and improve accuracy.

Contents

Vision

The Beginning

AI holds the potential to revolutionize our world even more so than electricity and fire. However, there are fundamental problems that are currently holding back the AI innovation. So far, there is no way for AIs to communicate: they cannot exchange data, trade services, learn from each other, or leverage their combined capabilities to solve the problems our world faces.

GenesisAI has 3 main goals:

1. To connect companies in need of AI services with companies who would like to monetize their AI technology. This will be achieved by developing the GenesisAI marketplace, where AI services can be sold and purchased by individual agents.
2. To connect many different AI services and leverage their data to eventually synergize into an Artificial General Intelligence. To achieve this, the GenesisAI platform has been designed to enable the interaction of multiple service providers (nodes) that can easily integrate with each other.
3. To confront the current system of AI oligopolies, where only a handful of large corporations own and operate AIs by making AI technology accessible to small companies and individuals.

Existing Problems with AI

Artifical Intelligence Terms

Artifical Intelligence (AI)
The ability of a digital computer-controlled robot to perform tasks commonly associated with intelligent beings. It can carry out complex tasks such as speech recognition, translation, visual perception.

Machine learning
A subset of AI that provides systems the ability to automatically learn and improve from the experience without being explicity programmed.

Deep learning
A subset of AI that imitates the workings of the human brain in processing data and creating patterns for the use in decision making.

Currently, three fundamental obstacles hold back the AI innovation. A project that solves this problem will unlock trillions of dollars in value.

Problem 1: No connectivity

Today, there is no way for AI products to exchange data, learn from each other, combine their capabilities to work towards a common goal, or trade/exchange services. AI technologies operate in a closed environment. Each company that develops AI, collects its own data sets rather than sharing data or using previously published data to train its AI. In other words, companies must independently create AI, which has already been created by other companies. This leads to redundancy and a waste of time and effort on the part of each company. For instance, there are hundreds of language processing AI products that have been developed to operate in their own closed environments. A large component of AI is machine learning, which requires the machine to have as many resources to interact with and learn from as possible. It is ironic and paradoxical that this fundamental requirement of machine learning is ignored as there is currently no way for different AI services to learn from each other. The lack of connectivity between different AI technologies is a major roadblock to creating Artificial General Intelligence. The solution to this fundamental problem will be able to unlock trillions of dollars in market value.

GenesisAI Solution: Protocol for communication

GenesisAI enables different AI technologies to communicate with each other, exchange data, learn from each other, and trade services.

Essentially, GenesisAI is comprised of many if-then statements that create AI communication protocol, which in turn makes a AI-to-AI economy possible. This enables anyone in the world to access AI services or be able to monetize AI code that they have created.

Communication protocol specifies the logic behind AI-to-AI economy and details how both the supplier and consumer parties can connect with each other as well as how they can exchange data, trade services, and learn from each other. Communication protocols make the process of using and monetizing AI services much simpler than any other option. Anyone, even someone with non-technical training, can participate in GenesisAI's marketplace.

Problem 2: Expensive to use

There are only around 10,000 AI developers in the world. 99% of businesses cannot afford to hire their own team of AI engineers to create AI, nor they can afford to risk integrating open-source AI API without technical expertise in the area. They are unable to determine which AI to integrate and or how to develop the AI for their specific needs. All of this makes current AI implementation extremely expensive.

GenesisAI Solution: Delivering inexpensive and fast AI solution

GenesisAI's web-platform enables businesses to provide their AI services to interested parties, thereby increasing the number of AI service providers. This increased supply of AI service providers will dramatically reduce the cost of using AI. Furthermore, we make it simple and easy for companies to use any type of AI work/service. Companies do not need to have in-house software engineers to create or adjust existing AI products in order to get work done. Rather, through simply following a GenesisAI protocol, companies can request a specific AI service, send their data to be analyzed, and receive the completed, high quality AI work.

There are many high-quality open-source APIs available on GitHub and elsewhere on the web, such as on Google's TensorFlow. However, these are hard to use and difficult to integrate even with technical expertise. Companies spend tens of thousands of dollars on those integrations. We are democratizing access to these APIs by wrapping the AI code in an easily accessible AI node. This further reduces the cost of AI work by increasing the total supply of accessible AI services. The ease of using wrapped APIs empowers even people without technical expertise to reach their goals with the power of sophisticated AI. No engineering work is required in order to use the AI technologies. This enables businesses to complete tasks in a more efficient and affordable manner.

Problem 3: No way to monetize AI code

AI developers and companies do not have an easy way to sell their AI services. For example, a smart computer science student in Bulgaria wrote an AI code and intended to sell his AI capabilities to companies, but the student could not monetize his AI code easily because there is currently neither a marketplace where you can easily find potential buyers, nor a way for AI services to be discovered. Enterprise sales are extremely challenging and costly (see Figure 2).

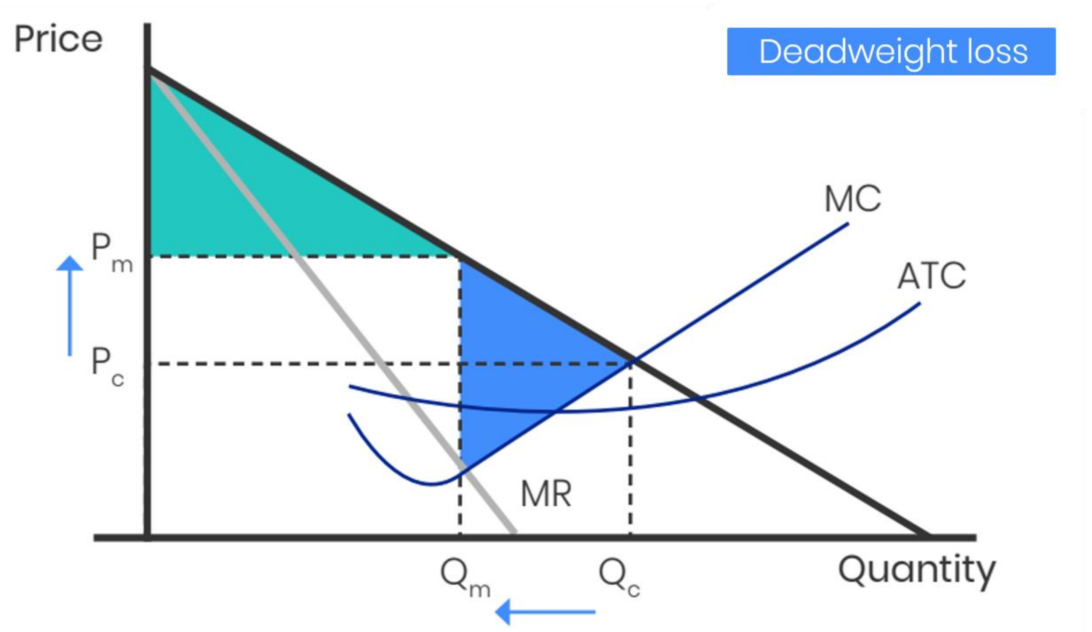

Figure 2: Monopolists raise prices to maximize their profits (with a price at Pm and a quantity of services utilized at Qm relative to the price and

quantity associated with a competitive market at Pc and Qc, respectively); this causes deadweight loss for the entire market. The yellow shaded area represents the consumer surplus which is significantly reduced compared to the consumer surplus for the price during perfect competition (yellow shaded area + $(P_m - P_c) \times Q_m + (Q_c - Q_m) \times \frac{Pm-Pc}{2}$

GenesisAI Solution: AI marketplace

GenesisAI breaks down many barriers of monetization. It eliminates the requirement of spending huge resources on B2B sales, as well as the need to provide custom AI integrations to each client company. GenesisAI also benefits small and emerging players in the field of AI by allowing these companies or individuals to be easily discoverable on the marketplace.

GenesisAI Network

GenesisAI Consensus Mechanism

GenesisAI introduces a trustless mechanism to verify the quality of a trained model using an evaluation function. Before going into the binding conditions, a buyer submits a dataset, a reward for a task and an evaluation function. If the sellers want to accept the task, they stake USD and download a dataset to train the models. Once the sellers want to check the quality of the models, they can run an evaluation function. The evaluation function will compare the solutions and the best model will get the predefined award from the buyer. The owner of the best model gets its stake back alongside with the stakes that other sellers made. If an evaluation function finds out that none of the submitted models are good enough, the buyer gets all the stakes made by the sellers.

The process of each transaction consists of 3 main phases:

1. A buyer submits a dataset for training the models, an evaluation function, and a reward amount to the contract. The evaluation function evaluates the quality of the model, and displays a score according to the quality of the model. Sellers get rewards in.
2. Sellers download the dataset provided by a buyer and work independently to train a machine learning model. After the training of the model is finished, sellers submit their solutions.
3. The submitted models will be evaluated by sellers using the evaluation function. The best model wins the competition.

GenesisAI Rating System

An AI-powered technology will match buyers and sellers. This technology will account for users' past behavior, willingness to pay, and needs. For example, if users recently ordered AI services in a certain vertical, like speech recognition, they will receive suggestions about similar AI services. Moreover, nodes will be able to filter their researches based on their willingness to pay for a particular service, and on how quickly they want to get this service.

The initial protocol is focused on providing as much flexibility as possible, with the tradeoff of adding some developer complexity for service providers. Incoming gRPC requests have metadata describing which model to run (model id), and a unique identifier for the job (job id) to make sure duplicate work is not being done if a request is resent. The payload consists of unstructured bytes that the service provider will have to parse.

Examples might be treating this as a list of int32, or treating it as a serialized protocol buffer that the seller would define elsewhere.

```
message Request {
    // Note : Request data and r e s p o n s e data a r e u n s t r u c t u r e d b y t e s. You w i l l
    // need to p u b l i s h th e f o rm at t h a t you e x p e c t . r e q u i r e d
    b y t e s data = 1 ;
    r e q u i r e d u i n t 6 4 j o b i d = 2 ;
    r e q u i r e d u i n t 6 4 m o d e l i d = 3 ;
}
```

As more services become available, we may define stricter protocols specific to a family of models. E.g. for image recognition models, we may create messages consisting of a 2D array of Pixel messages, containing fields for red, green, and blue values. Service providers may then choose to use this common protocol.

The protocol for the response is also meant to be as flexible as possible. The meta- data consists of a status indicating success or failure, and a job id used to link the output payload back to the input request. The payload is again an unstructured array of bytes that will need to be interpreted on the frontend.

```
message Response {
             enum S t a t u s {
            // Something s o e g r e g i o u s l y wrong happened t h a t no e r r o r c o de was
            // g e n e r a t e d .
            UNKNOWN = 0 ; // RPC c a l l c o m pl e te d s u c c e s s f u l l y . SUCCESS = 1 ;
            // Request . data was malformed . BAD
                  INPUT = 2 ;
            // S e r v i c e i s t a k i n g to o l o n g to re s p o n d .
            TIMEOUT = 3 ;
            }
            o p t i o n a l S t a t u s s t a t u s = 1 [ d e f a u l t = UNKNOWN] ;
            o p t i o n a l b y t e s data = 2 ;
            r e q u i r e d u i n t 6 4 j o b i d = 3 ;
}
```

GenesisAI Payment System

When requesting a model, buyers will send the eventual reward to the contract, which will hold the reward in escrow. The buyer can trigger the contract to return the reward back to the buyer and close the request at any time. The only other possible recipients of the reward are sellers who have staked on their submission.

Sellers can respond to this request by staking, along with references to their models. Once all submitted models are evaluated, the best performing seller is paid by the contract. The contract then logs the scores of all submitted models and their rank, which then will be visible to the buyer.

GenesisAI modules and interfaces

Interface

Represents a type of problem that needs to be solved. These consist of an input format, an output format, and a text label that is semantically meaningful to humans. High-level examples would include:

- Image Classification (maps images to a list of tuples of (text, float))
- Text Summarizer (maps text to text)
- Sentiment Analysis (maps text to floating point numbers)

However, the protocol becomes more powerful when it allows for lower-level modules, such as

- Feature Extraction (maps an MxN image to a list of floats)

Model

A service offered by sellers on our platform. Each model must implement at least one interface. I.e., the model accepts input and produces outputs in the format defined by the interface (accepts an image and produces text for the Image Classification interface).

Actions

Users of the protocol may do the following:

1. **Create an interface**
   Any buyer or seller can perform this action. The interface is created immediately, with no other steps required. Buyers would do this when looking for models that solve a particular problem (i.e. it is an RFP for models). Sellers would do this when they have models solving a novel problem, and want to make it available to the market.
2. **Propose to modify an existing interface**
   Any buyer or seller can perform this action. The proposal will either be accepted or rejected after some amount of time (to be determined, possibly on the order of a week). For the modification to take effect, the weighted votes must exceed some threshold (also to be determined, but will be above 50%). The weighting is described below.
3. **Vote to reject/accept a proposed interface modification**
   Any buyer or seller can perform this action. The weight of a sellers' vote is the total cost of all services sold with that interface since the modification was proposed. The weight of a buyers' vote is the cost of all services bought with that interface since the modification was

proposed. Note that one of the consequences of setting the threshold greater than 50% is that neither the buyers nor the sellers can change an interface without at least one party on the other side of the market agreeing.

4. **Add a model with an existing interface**
   Only sellers can perform this action. The model must implement the interface it is associated with (i.e. accepts inputs and produces outputs of the format specified by the interface).
5. **Remove a model**
   Sellers may stop providing services, but with a prior notice so that buyers are notified in time.

Incentives & Expected Emergent Behavior

1. **Competition within popular interfaces**
   Image classification is a very common use of inference models (high buyer demand). It should be widely used enough that no single party can easily change the interface. Because there is a clearly defined interface, it is easy for buyers to switch between different sellers to find the best performing model. We would expect rational buyers to experiment with different sellers, and gravitate towards those that perform the best for their data. Sellers, knowing this, have an incentive to beat the accuracy of competing models, since they know that they will get disproportionately more traffic or be able to charge more per job (or some mix of both).
2. **Emergence of ensembling**
   For interfaces with multiple models available, it becomes very cheap for intermediaries to create new models whose implementation consists only of calling other models and intelligently averaging their results1 . This is not something that would arise for every interface – the cost of ensembling would be at

least the sum of costs for the base models. There would need to be buyers willing to pay the high premium more improvements in accuracy.

3. **Modularization (aka poor-man's transfer learning)**
   Most of the state-of-the-art image classification models follow the general pattern of feeding inputs through several convolutional and pooling layers, then finally through a fully connected layer and a softmax layer. Using such a model for a domain specific task would typically require specially training on domain-specific data. Researchers get around this problem by using transfer learning. For any particular model, the early layers will have similar behavior regardless of what type of data they are trained on (e.g. edge detection, texture, shapes). These models are trained on generic inputs (ImageNet). Weights for those early layers are held fixed, while only the weights of the final fully connected layer (and possibly a handful of earlier layers) are retrained on domain specific data.

   This can be replicated with our protocols. Instead of a full image classification interface, sellers would offer models implementing a feature extraction interface, mapping images to just a list of floating point numbers. Sellers of domain-specific models would buy from sellers of feature extraction models to get an opaque representation of their image features, and would use that to train new domain specific models. There is a long tail of demand for disparate domain-specific models. There may not be enough of a financial incentive for any potential sellers to train (and collect data for) a domain specific model end-to-end, but having feature extraction models available makes it possible to train simpler, high-level models on top of those features, reducing the costs of entry. This creates incentives for other sellers to provide low-level feature extraction models, since it is effectively aggregating the demand from disparate domain-specific buyers.

GenesisAI Marketplace

How the Marketplace works

GenesisAI's web-platform has 3 parts:

1. Supply side agents. These are companies and AI developers who provide AI services such as speech recognition or language processing. In short, supply side AI nodes.
2. Demand side agents. These are people and organizations who would like to use AI services. For example, if an organization wants to predict where the next disease outbreak will happen, they may request.
3. Protocol. Our if-then statements specify rules for how the two parties can connect with each other and how AI-to-AI to economy will work. The protocol will incentivize the discovery of AI products and stimulate the creation of benevolent AI.

Technical documentation

Network Design

Structurally, GenesisAI's platform is a network. The nodes in this network are essentially AI service providers. A service is some action that takes input data and provides some output via machine learning. A classic example would be speech recognition running on given audio input. The service provider is the entity which consumes the data and executes code for the service.

Running a node

Given AI code written in Python, running a node in the GenesisAI network is simple (our initial node implementation is being written in Python). The service provider needs only to add a module that exposes the essential function(s) needed to perform the service.

In the beta version of the platform, we will support code written in other languages, by wrapping essential function(s) with Apache Thrift. This will allow developers to write code in C++, Java, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, JavaScript, Smalltalk, OCaml and Delphi and other languages.

Conclusion & Summary

GenesisAI's goal is to help businesses in need of AI services to connect with companies who would like to monetize their AI tech. Moreover, GenesisAI's vision is to connect as many different AIs as possible to form Artificial General Intelligence. Creation of such platform will unlock trillions of dollars in value and will be a Genesis for solving many of the humanity's problems - poverty and diseases. It is our ideology to build a marketplace: by the people - for the people. We want to smash the current system where only a handful of companies control a huge majority of the AI power.

The AI revolution is happening now and we want participants. This is our chance to change the world together. This is the Genesis of the new beginning.