Deuro: Technical AI White Paper

August 3, 2018

Abstract

When we develop our AI software solutions, we realize that approximately 10% to 30% of our budget goes into the artificial intelligence cloud computing infrastructure, and at least couple of million dollars for the labeled training data. In addition, we also experienced the trust and security concerns raised by our enterprise clients. Zoom requires us to do an on-premise distribution on their server to protect their user recordings. Bridgewater requires us to pass the software security compliance requirement. This is where Deuro comes in. Deuro is the world first and only artificial intelligence ecosystem on any blockchain. With decentralization, open-governance, cybersecurity and token economy on the mind, Deuro can help AI companies to dramatically reduce their computation cost, shorten the AI system development cycle and increase the production data privacy as well as security.

AI Solutions provided by Deuro

1. gStorage: Privacy-Preseving Decentralized Artificial Intelligence client-driven data storage system

    1.a. Feature extraction (Data preprocessing)

    1.b. Decentralized pseudo-anonymous multi-party key encryption and decryption

    1.c. AI Data(Conversation) as an asset

    1.d. Enterprise facing solution

    1.e. Consumer facing solution

    1.f. Modeling set repository

    1.g. Predicting data storage

    1.h. Models hub

2 gCrawl: Decentralized high performance algo-generated training data solution

    2.a. Crawling task as a transaction

    2.b. Crawled data as an asset

    2.c. Fault-tolerant distributed high performance crawling

    2.d. RAFT consensus algorithm

    2.e. Decentralized data storage

    2.f. DFSM decoding path

    2.g. Transfer learning for updating the model parameters

3. gPredict: Decentralized gradient learning framework

    3.a. Training task as a transaction

    3.b. Trained model as an asset

    3.c. Floating Point mapping

    3.d. Hyper-Parameters routing

    3.e. Results reducing

4. gCompute: Internet-scale AI dApps solution

    4.a. AI model as an asset

    4.b. Scalable

    4.c. Controllable

4.d. Difficult

4.e. Maintainable

4.f. Responsive

4.g. Concurrent

Keywords

AI dApps: Decentralized Artificial intelligence Application whose backend is usually the blockchain

Weak-AI: Weak Artificial intelligence / Applied Artificial intelligence

AGI: Artificial General intelligence

Brute-force search: In computer science, brute-force search or exhaustive search, also known as generate and test, is a very general problem-solving technique that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement.

AlphaGo: AlphaGo is a computer program that plays the board game Go.[1] It was developed by Alphabet Inc.'s Google DeepMind in London.

GD: Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point. If instead one takes steps proportional to the positive of the gradient, one approaches a local maximum of that function; the procedure is then known as gradient ascent.

Supervised-Learning: Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs

Unsupervised-learning: Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.

NBC: In machine learning, naive Bayes classifiers are a family of simple "probabilistic classifiers "based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

CA: Clustering analysis is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters)

DMR: dimensionality reduction or dimension reduction is the process of reducing the number of random variables under consideration[1] by obtaining a set of principal variables. It can be divided into feature selection and feature extraction

PCA: Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

K-NN: k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.

SVM: support vector machines (SVMs, also support vector networks[1]) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis

RL: Reinforcement learning (RL) is an area of machine learning inspired by behaviorist psychology[citation needed], concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward.

AI-complete: In the field of artificial intelligence, the most difficult problems are informally known as AI-complete or AI-hard, implying that the difficulty of these computational problems is equivalent to that of solving the central artificial intelligence problem—making computers as intelligent as people, or strong AI

NP-complete: In computational complexity theory, an NP-complete decision problem is one belonging to both the NP and the NP-hard complexity classes. In this context, NP stands for "nondeterministic polynomial time". The set of NP-complete problems is often denoted by NP-C or NPC

DL: Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised

CNN: convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

RNN: Recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence.

LSTM: Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate

Singularity: The technological singularity (also, simply, the singularity) is the hypothesis that the invention of artificial super intelligence (ASI) will abruptly trigger runaway technological growth, resulting in unfathomable changes to human civilization. According to this hypothesis, an upgradable intelligent agent (such as a computer running software-based artificial general intelligence) would enter a "runaway reaction" of self-improvement cycles, with each new and more intelligent generation appearing more and more rapidly, causing an intelligence explosion and resulting in a powerful super intelligence

GAN: Generative adversarial networks (GANs) are a class of artificial intelligence algorithms used in unsupervised machine learning, implemented by a system of two neural networks contesting with each other in a zero-sum game framework

SAAS: Software as a service is a software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted.

Colo: A colocation centre (also spelled co-location, or colo) or "carrier hotel", is a type of data centre where equipment, space, and bandwidth are available for rental to retail customers.

On-premise distribution: On-premises software (sometimes "on-premise" or abbreviated as "on-

prem") is installed and runs on computers on the premises (in the building) of the person or organization using the software, rather than at a remote facility such as a server farm or cloud

DFSM: Deterministic finite state machine is a finite-state machine that accepts and rejects strings of symbols and only produces a unique computation (or run) of the automaton for each input string

MSE: Mean squared error

BFT: Byzantine fault tolerance

Force-align: Forced alignment refers to the process by which orthographic transcriptions are aligned to audio recordings to automatically generate phone level segmentation

MFCC: Mel-frequency cepstral coefficients - computing feature representation of audio

Solutions provided by Deuro

Here we propose our innovative AI solutions based on the blockchain to solve the problems mentioned above.

gStorage: For the data security and privacy problem, we propose the gStorage solution. With AI special use case and blockchain smart contract support in mind, we introduced the distributed end-to-end encrypted solution to store/retrieve the training data, the production prediction data, and the model files.

gCrawl: For the lack of the labeled data problem, we propose the innovative gCrawl architecture. The gCrawl system is composed of fault-tolerant distributed high-performance data crawling system, stateful NoSQL database(deduplication as well as metadata storage), decentralized storage, simplified model decoding path and transfer learning for model parameters to finish the feedback loop.

gPredict: For the algorithm accuracy problem, we proposed an innovative way to train/evaluate/test the model performance on the blockchain. gPredict includes floating point mapping, hyperparameters routing, and decoding results reducing.

gCompute: For the high hardware as well software costs when developing, testing and deploying AI systems, we introduced the gCompute module. The gCompute module is scalable, controllable, difficulty, maintainable, responsive and concurrent.

1. gStorage: Privacy-Preseving Decentralized Artificial Intelligence client-driven data storage system

In order to securely store the user recording in a trustless way, we have to ensure privacy and security for our customer. Here we propose an innovate way to store user recording data on the blockchain. Below is the workflow chart for how gStorage works.

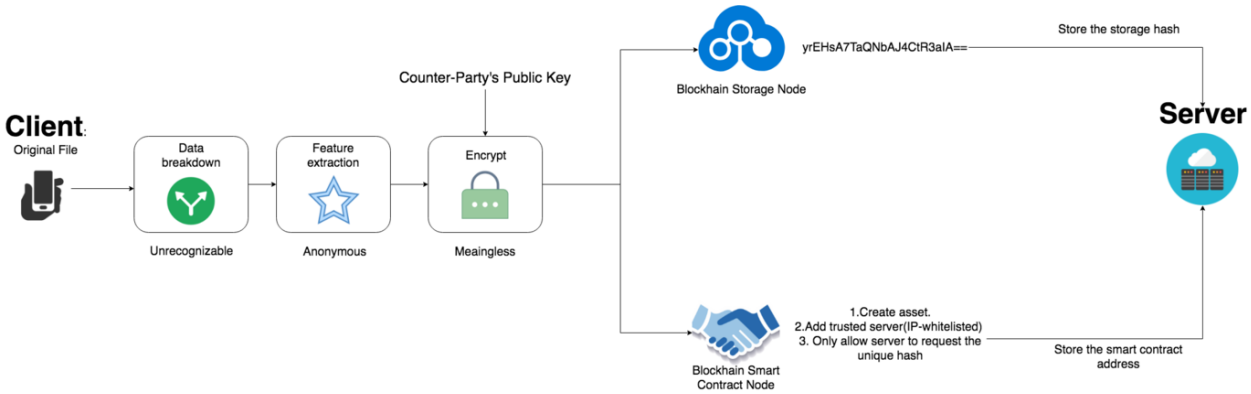


Figure 1: gStorage workflow

1.a. Feature extraction (Data preprocessing)

Here we showed the common features used in both speech recognition and image recognition. Audio Features: MFCC, i-vector. Image features: edges, corners/interest points, blobs/region of interest points, ridges.

MFCC

Mel-Frequency Cepstral Coefficients. MFCC are mostly used features in state-of-art speech recognition system.

```
class MFCC(object):
    def __init__(self, nfilt=40, ncep=13,
                 lowerf=133.3333, upperf=6855.4976, alpha=0.97,
                 samprate=16000, frate=100, wlen=0.0256,
                 nfft=512):
        # Store parameters
        self.lowerf = lowerf
        self.upperf = upperf
        self.nfft = nfft
        self.ncep = ncep
        self.nfilt = nfilt
        self.frate = frate
        self.fshift = float(samprate) / frate

        # Build Hamming window
        self.wlen = int(wlen * samprate)
        self.win = numpy.hamming(self.wlen)

        # Prior sample for pre-emphasis
        self.prior = 0
        self.alpha = alpha

        # Build mel filter matrix
        self.filters = numpy.zeros((nfft/2+1,nfilt), 'd')
```

```
dfreq = float(samprate) / nfft
if upperf > samprate/2:
    raise(Exception,
          "Upper frequency %f exceeds Nyquist %f" % (upperf, samprate/2))
melmax = mel(upperf)
melmin = mel(lowerf)
dmelbw = (melmax - melmin) / (nfilt + 1)
# Filter edges, in Hz
filt_edge = melinv(melmin + dmelbw * numpy.arange(nfilt + 2, dtype='d'))

for whichfilt in range(0, nfilt):
    # Filter triangles, in DFT points
    leftfr = round(filt_edge[whichfilt] / dfreq)
    centerfr = round(filt_edge[whichfilt + 1] / dfreq)
    rightfr = round(filt_edge[whichfilt + 2] / dfreq)
    # For some reason this is calculated in Hz, though I think
    # it doesn't really matter
    fwidth = (rightfr - leftfr) * dfreq
    height = 2. / fwidth

    if centerfr != leftfr:
        leftslope = height / (centerfr - leftfr)
    else:
        leftslope = 0
    freq = leftfr + 1
    while freq < centerfr:
        self.filters[freq,whichfilt] = (freq - leftfr) * leftslope
        freq = freq + 1
    if freq == centerfr: # This is always true
        self.filters[freq,whichfilt] = height
        freq = freq + 1
```

```
            if centerfr != rightfr:
                rightslope = height / (centerfr - rightfr)
            while freq < rightfr:
                self.filters[freq,whichfilt] = (freq - rightfr) * rightslope
                freq = freq + 1

        # Build DCT matrix
        self.s2dct = s2dctmat(nfilt, ncep, 1./nfilt)
        self.dct = dctmat(nfilt, ncep, numpy.pi/nfilt)

    def sig2s2mfc(self, sig):
        nfr = int(len(sig) / self.fshift + 1)
        mfcc = numpy.zeros((nfr, self.ncep), 'd')
        fr = 0
        while fr < nfr:
            start = round(fr * self.fshift)
            end = min(len(sig), start + self.wlen)
            frame = sig[start:end]
            if len(frame) < self.wlen:
                frame = numpy.resize(frame,self.wlen)
                frame[self.wlen:] = 0
            mfcc[fr] = self.frame2s2mfc(frame)
            fr = fr + 1
        return mfcc

    def sig2logspec(self, sig):
        nfr = int(len(sig) / self.fshift + 1)
        mfcc = numpy.zeros((nfr, self.nfilt), 'd')
        fr = 0
        while fr < nfr:
            start = round(fr * self.fshift)
```

```
            end = min(len(sig), start + self.wlen)
            frame = sig[start:end]
            if len(frame) < self.wlen:
                frame = numpy.resize(frame,self.wlen)
                frame[self.wlen:] = 0
            mfcc[fr] = self.frame2logspec(frame)
            fr = fr + 1
        return mfcc

    def pre_emphasis(self, frame):
        # FIXME: Do this with matrix multiplication
        outfr = numpy.empty(len(frame), 'd')
        outfr[0] = frame[0] - self.alpha * self.prior
        for i in range(1,len(frame)):
            outfr[i] = frame[i] - self.alpha * frame[i-1]
        self.prior = frame[-1]
        return outfr


    def frame2logspec(self, frame):
        frame = self.pre_emphasis(frame) * self.win
        fft = numpy.fft.rfft(frame, self.nfft)
        # Square of absolute value
        power = fft.real * fft.real + fft.imag * fft.imag
        return numpy.log(numpy.dot(power, self.filters).clip(1e-5,numpy.inf))

    def frame2s2mfc(self, frame):
        logspec = self.frame2logspec(frame)
        return numpy.dot(logspec, self.s2dct.T) / self.nfilt

def s2dctmat(nfilt,ncep,freqstep):
    "" "Return the 'legacy' not-quite-DCT matrix used by Sphinx" ""
```

```
    melcos = numpy.empty((ncep, nfilt), 'double')
    for i in range(0,ncep):
        freq = numpy.pi * float(i) / nfilt
        melcos[i] = numpy.cos(freq * numpy.arange(0.5, float(nfilt)+0.5, 1.0, 'double'))
    melcos[:,0] = melcos[:,0] * 0.5
    return melcos

def logspec2s2mfc(logspec, ncep=13):
    """Convert log-power-spectrum bins to MFCC using the 'legacy'
    Sphinx transform"""
    nframes, nfilt = logspec.shape
    melcos = s2dctmat(nfilt, ncep, 1./nfilt)
    return numpy.dot(logspec, melcos.T) / nfilt

def dctmat(N,K,freqstep,orthogonalize=True):
    """Return the orthogonal DCT-II/DCT-III matrix of size NxK.
    For computing or inverting MFCCs, N is the number of
    log-power-spectrum bins while K is the number of cepstra."""
    cosmat = numpy.zeros((N, K), 'double')
    for n in range(0,N):
        for k in range(0, K):
            cosmat[n,k] = numpy.cos(freqstep * (n + 0.5) * k)
    if orthogonalize:
        cosmat[:,0] = cosmat[:,0] * 1./numpy.sqrt(2)
    return cosmat

def dct(input, K=13):
    "" "Convert log-power-spectrum to MFCC using the orthogonal DCT-II""
    nframes, N = input.shape
    freqstep = numpy.pi / N
    cosmat = dctmat(N,K,freqstep)
```

```
    return numpy.dot(input, cosmat) * numpy.sqrt(2.0 / N)

def dct2(input, K=13):
    “” “Convert log-power-spectrum to MFCC using the normalized DCT-II” “”
    nframes, N = input.shape
    freqstep = numpy.pi / N
    cosmat = dctmat(N,K,freqstep,False)
    return numpy.dot(input, cosmat) * (2.0 / N)

def idct(input, K=40):
    “” “Convert MFCC to log-power-spectrum using the orthogonal DCT-III” “”
    nframes, N = input.shape
    freqstep = numpy.pi / K
    cosmat = dctmat(K,N,freqstep).T
    return numpy.dot(input, cosmat) * numpy.sqrt(2.0 / K)

def dct3(input, K=40):
    “” “Convert MFCC to log-power-spectrum using the unnormalized DCT-III” “”
    nframes, N = input.shape
    freqstep = numpy.pi / K
    cosmat = dctmat(K,N,freqstep,False)
    cosmat[:,0] = cosmat[:,0] * 0.5
    return numpy.dot(input, cosmat.T)
```

i-Vector

An i-vector system uses a set of low-dimensional total variability factors (w) to represent each conversation side. Each factor controls an eigen-dimension of the total variability matrix (T), and are known as the i-vectors.

s(Conversationside supervector) = m + Tw(Total-variability matrix and i-vector)

1. To train T, run exact training procedure used to train V, but treat all conversation sides of all training speakers as belonging to different speakers
2. Given T, obtain i-vectors (w) for each conversation side
3. For channel compensation of i-vectors, perform LDA then WCCN (techniques empirically determined to perform well) on i-vectors. Denote channel-compensated i-vectors as ω.

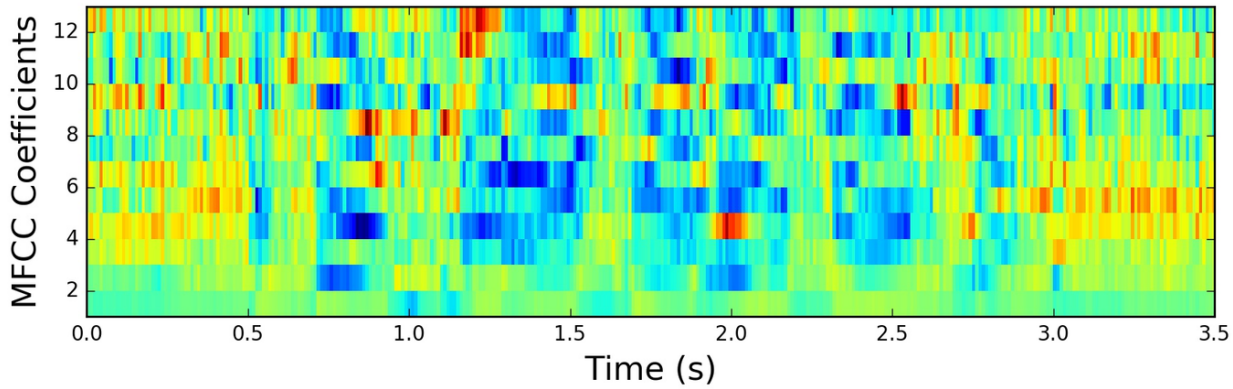


Figure 2: MFCC visulization

4. Perform cosine distance scoring (CDS) on channel-compensated i-vectorsω for a pair of conversation sides

$$score(\omega_1, \omega_2) = \frac{\omega_1^* * \omega_2}{\| \omega_1 \| * \| \omega_2 \|} = \cos(\theta_{\omega_1, \omega_2})$$

Figure 3: cosine distance scoring (CDS)

Edges

Edges are focuses where there is a limit (or an edge) between two picture locales. As a rule, an edge can be of relatively discretionary shape, and may incorporate intersections. By and by, edges are generally characterized as sets of focuses in the picture which have a solid angle extent. Moreover, some regular calculations will then chain high inclination directs together toward frame a more total depiction of an edge. These calculations more often than not put a few limitations on the properties of an edge, for example, shape, smoothness, and inclination esteem. Locally, edges have a one-dimensional structure.

Corners / interest points

The terms corners and intrigue focuses are utilized to some degree conversely and allude to point-like highlights in a picture, which have a neighborhood two dimensional structure. The name "Corner" emerged since early calculations initially performed edge discovery, and afterward broke down the edges to discover quick alters in course (corners). These calculations were then grown with the goal that express edge identification was never again required, for example by searching for abnormal amounts of shape in the picture inclination. It was then seen that the purported corners were likewise being recognized on parts of the picture which were not corners in the conventional

sense (for example a little brilliant spot on a dim foundation might be identified). These focuses are every now and again known as intrigue focuses, however the expression "corner" is utilized by custom.

Blobs / regions of interest points

Blobs give a correlative portrayal of picture structures as far as locales, instead of corners that are more point-like. By and by, blob descriptors may regularly contain a favored point (a neighborhood most extreme of an administrator reaction or a focal point of gravity) which implies that numerous blob identifiers may likewise be viewed as intrigue point administrators. Blob finders can distinguish territories in a picture which are too smooth to be in any way recognized by a corner indicator.

Think about contracting a picture and afterward performing corner identification. The finder will react to focuses which are sharp in the contracted picture, yet might be smooth in the first picture. It is now that the distinction between a corner locator and a blob identifier turns out to be to some degree dubious. To a vast degree, this qualification can be cured by including a proper thought of scale. In any case, because of their reaction properties to various sorts of picture structures at various scales, the LoG and DoH blob finders are additionally said in the article on corner discovery.

Ridges

For stretched items, the thought of edges is a characteristic apparatus. An edge descriptor processed from a dim level picture can be viewed as a speculation of an average pivot. From a useful perspective, an edge can be thought of as a one-dimensional bend that speaks to a hub of symmetry, and what's more has a quality of neighborhood edge width related with each edge point. Tragically, be that as it may, it is algorithmically harder to separate edge highlights from general classes of dark level pictures than edge-, corner-or blob highlights. By and by, edge descriptors are as often as possible utilized for street extraction in flying pictures and for removing veins in restorative pictures—see edge identification.

1.b. Decentralized pseudo-anonymous multi-party key encryption and decryption

Here we will discuss the core of the gStorage. The decentralized pseudo-anonymous multi-party key encryption and decryption algorithms.

1. It's decentralized in a sense that we will use blockchain as the backend and broadcast the signed transaction to the whole network without relaying on a trusted centralized party, which may cause single point of failure.
2. Then, It's pseudo-anonymous in a sense that we only protect the extracted feature and expose the encrypted hash to the whole network. Just like how Bitcoin works(user signed the transaction locally using their private key, and then broadcast the transaction hash to the whole network to verify).
3. It involves multi-party in a sense that we will need to know the counter-party public key ahead of time so that the counter-party(server, IOT device, website, IP address, smart contract) can decrypt the encrypted hash later using their own private key.

Protect the extracted feature

Once we have the extracted feature representation for AI model, we will need to protect it. We

protect it by encrypt the extracted feature into cipher text using counter-party's public key. Here we will use the industry standard public key encryption algorithm.

1. each party has a PAIR (K, K-1) of keys: K is the public key and K-1 is the secret key, such that DK-1[EK[M]] = M
2. Knowing the public-key and the cipher, it is computationally infeasible to compute the private key Public-key crypto system is thus known to be asymmetric crypto systems
3. The public-key K may be made publicly available, e.g., in a publicly available directory
4. Many can encrypt, only one can decrypt

Key generation: Select 2 large prime numbers of about the same size, p and q. Compute n = pq, and $\Phi(n) = (q-1)(p-1)$ Select a random integer e, $1 < e < \Phi(n)$, s.t. $gcd(e, \Phi(n)) = 1$. Compute d, $1< d< \Phi(n)$ s.t. $ed =1 \bmod \Phi(n)$. Public key: (e, n) Secret key: d

Encryption: Given a message M, $0 < M < n$ use public key (e, n) computeC=Me modn

Decryption: M ϵ Zn- {0} CϵZn-{0}.Given a ciphertext C, use private key (d).ComputeCd modn=(Me modn)d modn n = M

Broadcast the encrypted hash

Once we have the extracted feature encrypted, we will put into into the p2p decentralized database. Then register the unique hash of the encrypted features into the blockchain, and broadcast this encrypted hash into the whole network. The miner will know the encrypted hash of the features, can find all features belong to a single user, but they can not decrypt it. Just like all bitcoin miners can find transaction belong to a single user, but they can not transfer the money into miner's own account, because the miners do not have the private key of the interested account.

Transfer the ownership of asset to counter-party

Once we register the feature asset on the blockchain, server can request the unique hashes of features belonged to a particular user. Then server can retrieve the encrypted cipher text from the p2p decentralized database, and decrypt the cipher text for the features using its own private key.

———-

Solidity ETH access_control_template()

———-

```
pragma solidity ^0.4.23;


contract gStorageAccessControl {

    address admin;

    address server_address;


    modifier onlyByServer { if (msg.sender != server_address) throw; _ }
```

```
    modifier onlyByAdmin { if (msg.sender != admin) throw; _ }
     address[] public features;
    event FindTheUniqueHash(address[] unique_hash);


    function gStorageAccessControl(address server_address_ ) {
        admin = msg.sender;
        server_address = server_address_;
    }
    function registerFeatures(address unique_hash) onlyByAdmin{
        features.push(unique_hash);
    }
    function transfer() onlyByServer {
        //Retrive data from the storage
        FindTheUniqueHash(features);
    }
}
```

Privacy-preserving pseudo-anonymous data sharing

In this way we protect the privacy of the customer by applying the feature extraction. We protect the security of the data using industry standard public key encryption. We also enforce the access control by passing the countery-party's public key into the smart contract.

1.c. AI Data(Conversation, Images, Sentences) as an asset

Audio data comes in a sometimes bewildering variety of forms. The number of fundamental ways in which sound can be represented is actually fairly small. The variety of audio file types is due to the fact that there are quite a few approaches to compressing audio data and a number of different ways of packaging the data. We can abstract an conversation recording as an asset on the blockchain. An asset can also be a state machine where the state transition is represented in the metadata. Each time the machine changes its state, a transaction is triggered to update the metadata to the new state (possibility to listen to it with the WebSocket or HTTP rest way). In our case, we can implement the timestamped_private_key in this asset state machine transformation way.

1.d. Enterprise facing solution

For the enterprise customer, we will deploy a out-of-the box solution for them. And the customer will need to purchase our token in order to receive our service. Once we finish our services, we will

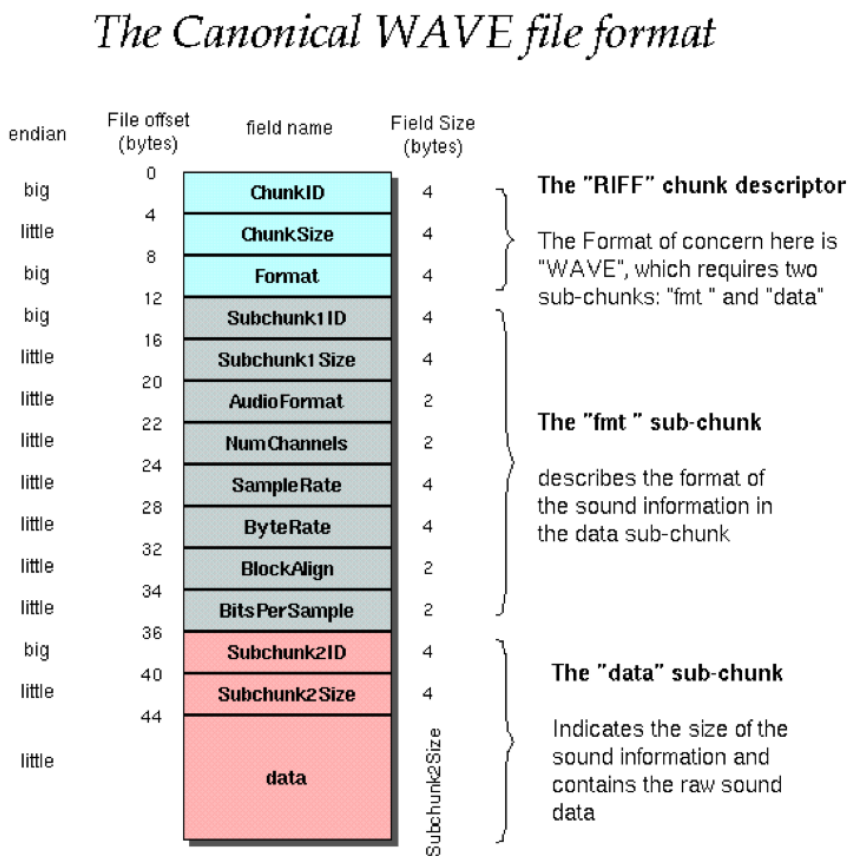


Figure 4: The raw audio file Wav representation

burn their purchased token.

1.e. Consumer facing solution

For the consumer facing application, developers can quickly have gStorage working on their app by simply installing our SDK. We will implement a freemium model, once they hit certain usage, we will ask them to purchase certain amount of our token as a reserve. And we will burn their token per their API call.

1.f. Modeling set repository

With the above architecture in mind, we can store all the training data easily into the blockchian peer-to-peer database. For audio training data, we can store raw uncompressed or lossless compressed wav file and corresponding proud truth file.

1.g. Predicting data storage

For all the production data generated by the AI dApps, it can also go through our existing gStorage pipeline. And allow user to securely retrieve as well as send the files.

1.h. Models hub

For the models generated by popular deep learning frameworks like Tensor-flow, Keras, Pytorch, Kaldi, Caffee, CNTK etc, we can also use the above method to securely put and get them from the blockchain database.

2. gCrawl: Distributed high-performance algo-generated training data solution

We exhibit a brand new machine learning architecture called "gCrawl" for utilizing unlabeled information in supervised classification tasks. We don't assume that the unlabeled data takes after a similar class names or generative description as the marked data. Hence, we might want to utilize an extensive number of unlabeled pictures (or sound examples, or content records) arbitrarily downloaded from the Internet to enhance execution on a given picture (or sound, or content) order task. We describe an approach to gCrawl that uses simplified decoding path and transfer learning to iterate models from unlabeled data.

Below is an architecture of a gCrawl computing cluster and it is horizontal scalable. In practice, we have 1 masterNode and 11 slaveNode per each cluster , which is capable of processing **1 million crawling tasks a day!** By preprocessing all the tasks in the NoSQL database, we were able to deploy 12 such clusters, resulting **12 million crawling tasks a day!**

2.a. Crawling task as a transaction

We will abstract crawling task as a transaction on the blockchain ledger.

```
contract CrawlingTask {

    struct CrawlingMachine {
        bool crawled;
        address location;
    }

    struct Task {
        string url;
    }

    address submiter;
    mapping(address => CrawlingMachine) machines;
    Task[] tasks;

    function crawl() {
        //The crawling business logic
    }
}
```

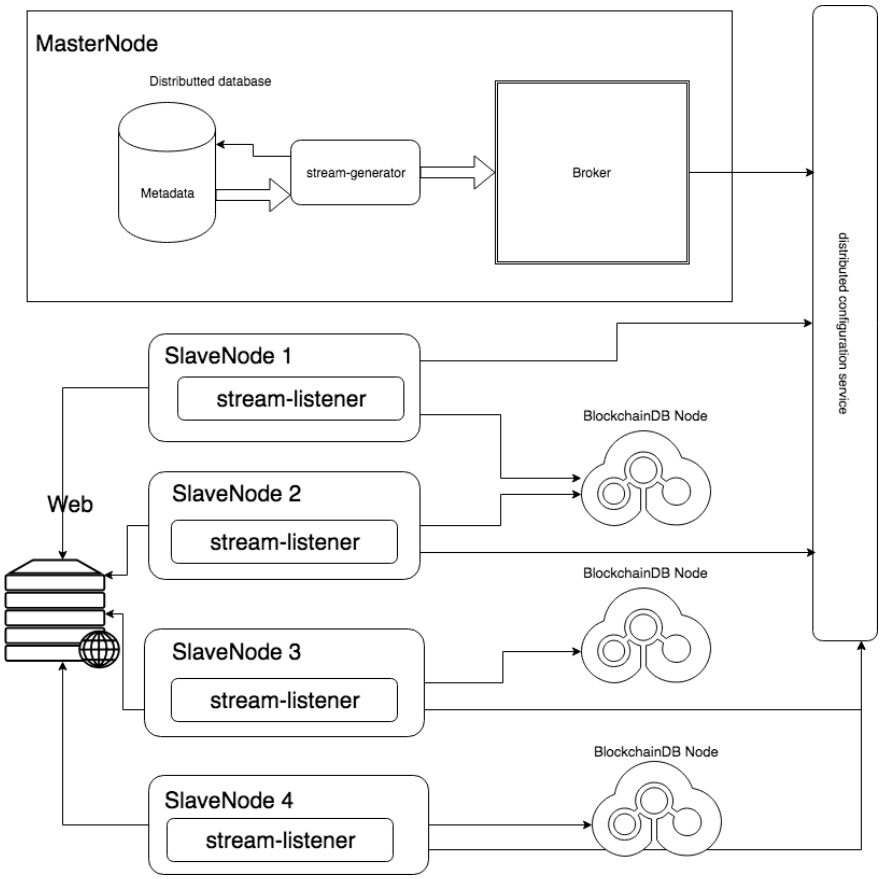


Figure 5: gCrawl crawling system

2.b. Crawled data as an asset

We can abstract crawled data as an asset on the blockchain

```
contract CrawledData {
    string[] metadata;
    string transcript;
    address audio_url;
}
```

2.c. Fault-tolerant distributed high performance crawling

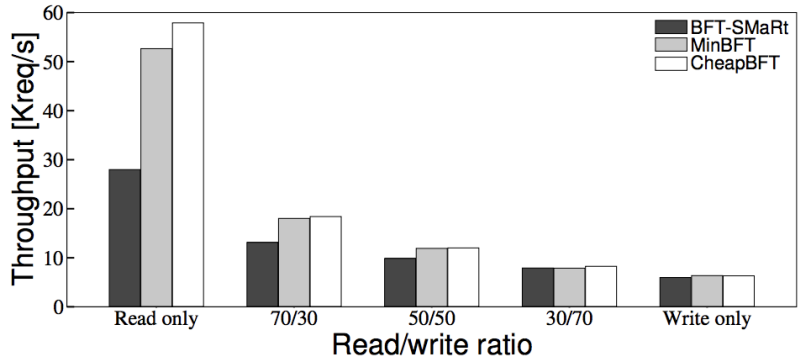
ZooKeeper is a crash-tolerant coordination service utilized as a part of large-scale distributed frameworks for essential tasks like pioneer decision, synchronization, and failure recognition. This section displays an assessment of a ZooKeeper-like BFT benefit that depend on BFT-SMaRt, MinBFT, and CheapBFT for request dissemination, separately. ZooKeeper enables customers to store and recover (generally little) chunks of data in information hubs, which are overseen in a various leveled tree structure. We assess the three executions for various blends of read and compose tasks. In all cases, 1,000 customers more than once get to various information hubs, perusing and composing

information pieces of arbitrary sizes between one byte and two kilobytes. The outcomes demonstrate that with the execution stage(I. e., the ZooKeeper application) performing genuine work (and not simply sending answers), the effect of the assention convention on framework execution is diminished. In result, each of the three ZooKeeper usage give comparative throughput to compose overwhelming workloads. Be that as it may, the asset impressions fundamentally contrast between variations: in contrast with the MinBFT-based ZooKeeper, the reproductions in the CheapBFT-based variation spare 7-12% CPU and send 12-20% less information over the system. Contrasted with the BFT-SMaRt execution, the asset funds of the CheapBFT-based ZooKeeper signify 23-42% (CPU) and 27-43% (organize).

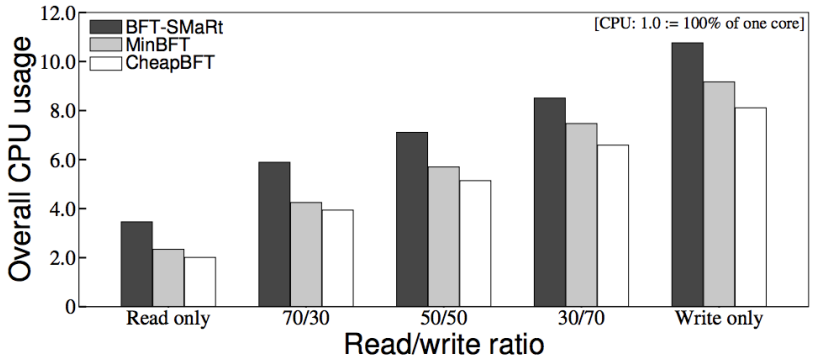
2.d. RAFT consensus algorithm

Raft is the consensus algorithm used in many NoSQL databases. We use NoSQL database to store all the metadata information of the crawling task. Below is the pseudo code for Leader Election for the RAFT consensus algorithm

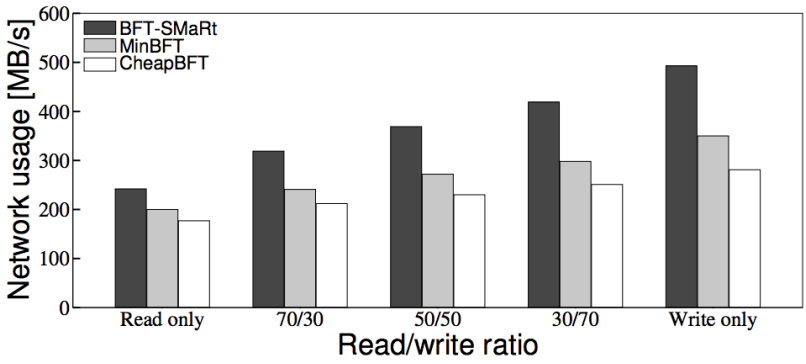
```
# follower not receiving heartbeats from leader
if election_timeout?
  increment_term()
  change_state_to_candidate()
  vote_myself()
  send_request_vote_to_all_partners()
  while candidate?
    wait_feedback(random_time)
    process_feedback()
    foreach entry in append_entries_received()
      if self.term() < entry.term()
        mark_current_election_as_lost()
      end
    end
    if i_won_election?
      change_state_to_leader()
      start_heartbeat_signals()
    else if partner_won_election?
      change_state_to_follower()
    else if no_one_won_election?
      increment_term()
    end
  end
end

# continue as leader or follower...
```



(a) Realized throughput for 1,000 clients.



(b) CPU usage per 10 Kreq/s normalized by throughput.



(c) Network transfer volume per 10 Kreq/s normalized by throughput.

Figure 6: Execution and resource utilization results for different BFT variants of our ZooKeeper service for workloads comprising distinctive mixes of read and write operations.

2.e. Decentralized data storage

We will use different decentralized data blockchain-based storage solution to store the crawled data. We will explain more about the storage in the section 3.1

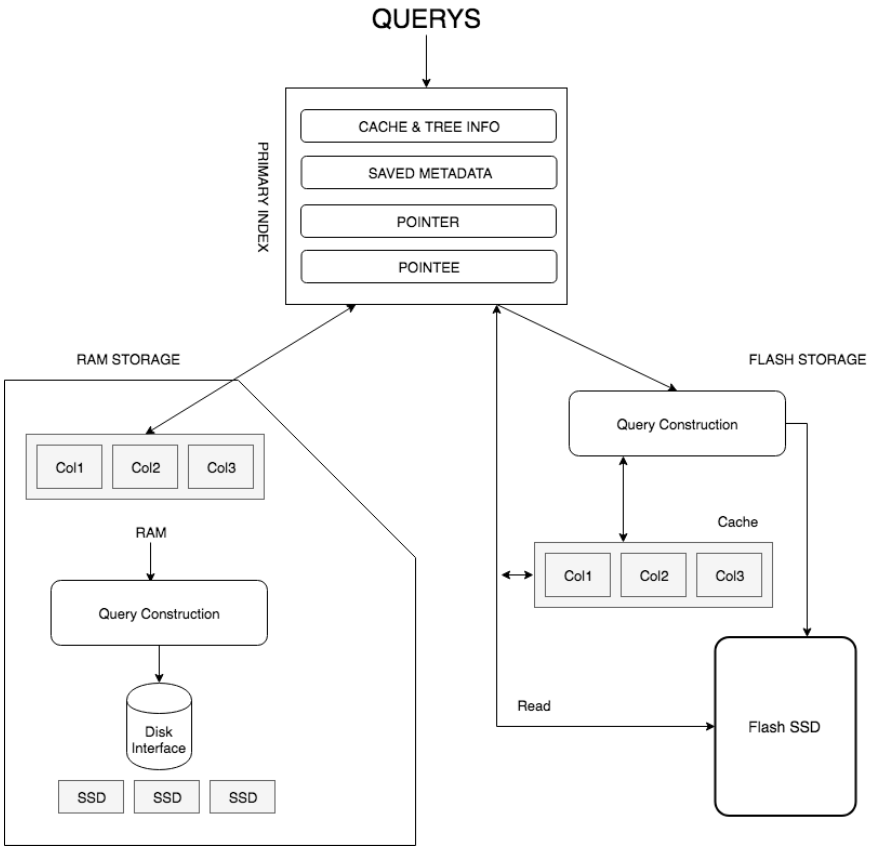


Figure 7: Distributted database storage

2.f. DFSM decoding path

When we prepare the supervised learning data, we simplify our model decoding path and ask the model DFSM a simpler question. Instead of asking the model to predict what is the phenom of this utterance, we also present the hypothesis text and ask the model to tell us wether we can align the text into the audio segment or not. Usually the Language model of speech recognition system is unsupervised-learning,

Command to generate the full Kaldi decoding graph.

```
make_lexicon_fst.pl lexicon_disambig.txt 0.5 sil '#'$ndisambig | \
   fstcompile --isymbols=lexgraphs/phones_disambig.txt \
    --osymbols=lmgraphs/words.txt \
    --keep_isymbols=false --keep_osymbols=false |\
   fstaddselfloops  $phone_disambig_symbol $word_disambig_symbol | \
   fstarcsort --sort_type=olabel \
   > lexgraphs/L_disambig.fst
```

Once acoustic models have been created, Kaldi can also perform force-align on audio accompanied by a word-level transcript

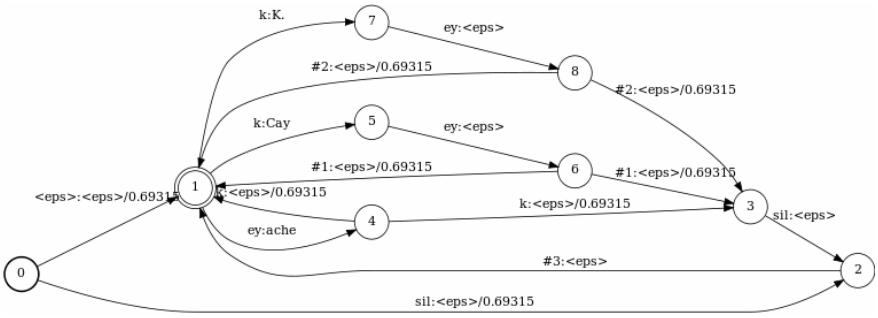


Figure 8: Lexicon full decoding graph

1. Create data/train files according to the new transcript and audio
2. Extract MFCC features
3. Align data
4. Obtain CTM output from alignment files
5. Concatenate CTM files
6. Convert time marks and phone IDs
7. Split final_ali.txt by file
8. Create word alignments from phone endings
9. Append header to each of the text files for Praat
10. Make Praat TextGrids of phone alignments from .txt files
11. Make Praat TextGrids for word alignments from word_alignment.txt
12. Stack phone and word TextGrids

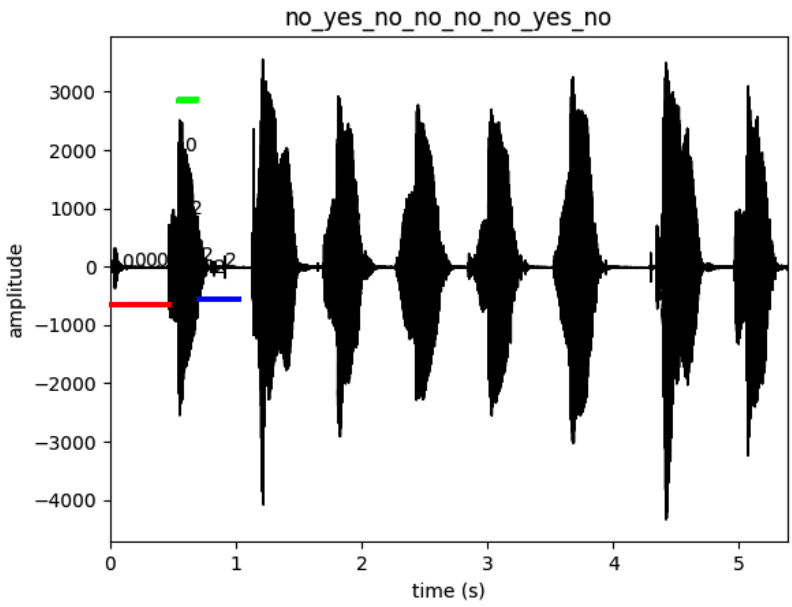


Figure 9: Simplified audio force-align flow

2.g. Transfer learning to update the model parameters

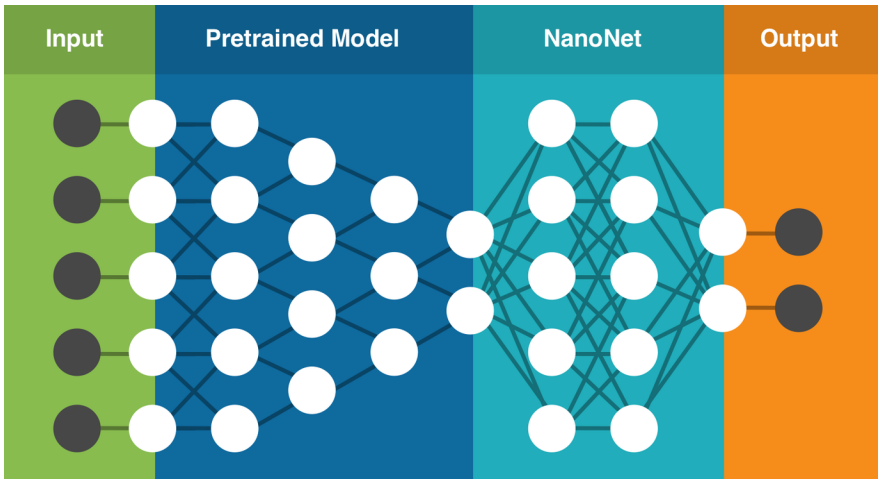


Figure 10: Transfer Learning diagram

(Transfer Learning) Given a source domain DS and learning task TS, a target domain DT and learning task TT , transfer learning aims to help improve the learning of the target predictive function fT (·) in DT using the knowledge in DS and TS, where DS != DT , or TS != TT . In the above definition, a domain is a pair D = {X , P(X)}. Thus the condition DS != DTimplies that either XS != XT or PS(X) != PT (X).

3. gPredict: Decentralized gradient learning framework

In order to improve the performance of AI algorithm, we need to increase the precision/recall, decrease the MSE. In this section, we present a set of ways to reduce the AI model training turn-around time on the blockchain.

Estimator: is just a simple function to model some meaningful characteristics of data sample. For example, $Y = g(S)$, $S = (x(1),,,,x(m))$, where x(i) is a random variable drawn from distribution D, i.e. $x(i) \sim D$. **Estimator Bias** measures how good our estimator is in estimating the real number. $Bias(Y_S, Y) = E_{S\sim D^m}[Y_S] - Y$ **Estimator Variance** measure how 'jumpy' our estimator is to sampling. $Var(Y) = Var_{S\sim D^m}[Y]$. **Bias-variance decomposition for estimators:** combines those two properties in one formula. $MSE = E[(Y_S - Y)^2] = Bias^2(Y_S, Y) + Var(Y_S)$ where the expectations are taken with respect to **S** random variable.

Formal proof for the above conclusion:

$$E[(Y_S - Y)^2] = E[Y_S^2] + Y^2 - 2E[Y_S]Y$$

$$Bias^2(Y_S, Y) = (E[Y_S] - Y)^2$$

$$= E^2[Y_S] + Y - 2E[Y_S]Y$$

$$Var(Y_S) = E[Y_S^2] - E^2[Y_S]$$

3.a. Training task as a transaction

```
contract TrainingTask {
    TrainedModel previousModel;
    function training();
    TrainedModel trainedModel;
}
```

3.b. Trained model as an asset

```
contract TrainedModel {
    string[][] configuration;
    double[] weights;
    state[] optimizerState;
}
```

3.c. Floating Point mapping

According to our study of the limited number precision within the low-precision fixed-point context, we find that by controlling the rounding scheme using only 16-bit representation when using stochastic gradient descent, we can train the model even faster without hurting any accuracy.

Limited Precision Arithmetic: Standard executions of back-propagation powered deep learning system commonly utilize 32-bit floating-point representation of real numbers for data storage and control. We abstract this problem by representing [IN.FR], where IN and FR relate to the integer and fractional part of the number, individually. The number of number bits(NB) plus the quantity of fractional bits(FB) yields the aggregate number of bits used to represent the whole number. The sum of NB + FB is referred as the word length WL.

Stochastic rounding: Given a number x and the target fixed-point representation <NB, FB>. We define $\lfloor x \rfloor$ as the largest integer multiple of $\theta \left(= 2^{-FB}\right)$ <= x. Stochastic rounding is an unbiased rounding scheme and possess the desirable property that the expected rounding error is zero, i.e. E(Round(x, <NB,FB>)) = x

So, P(rounding x to $\lfloor x \rfloor$) is relative to the approximation of x to $\lfloor x \rfloor$:

$$Round(x, < NB, FB >) = \begin{cases} \lfloor x \rfloor, & amp; \text{if } 1 - \dfrac{x - \lfloor x \rfloor}{\theta}, \\ \lfloor x \rfloor + \theta, & amp; \dfrac{x - \lfloor x \rfloor}{\theta}. \end{cases}$$

Then,

$$Convert(x, < NB, FB >) = \begin{cases} -2^{NB-1}, & amp; \text{if } x <= -2^{NB-1}, \\ 2^{NB-1} - 2^{-FB}, & amp; \text{if } x >= 2^{NB-1} - 2^{-FB}. \\ Round(x, < NB, FB >), & amp; otherwise. \end{cases}$$

Multiply and accumulate operation: Consider two dimensional vectors **d** and **h** such that every segment is represented in the settled fixed-point <NB,FB>, and define **e** = d.h as the inner product of d and h. Then we can split the computation into the following steps:

$$z = \sum_{i=1}^{d} a_i b_i$$

$$c_0 = Convert(z, < NB, FB >)$$

3.d. Hyper-Parameters routing

gPredict's activity is to locate the best estimation of a scalar-esteemed, perhaps stochastic function over an arrangement of possible arguments to that function. While numerous bundles will expect that these data sources are drawn from a vector space, gPredict is distinctive in that it urges developer to depict the search space in more detail. By giving more data about where the function is characterized, and where the best qualities are, gPredict can search the best parameter more efficiently.

3.e. Results reducing

Finally with the help of floating point mapping and hyper-parameters routing, we can get the results by reducing from blockchain type of distributed system.

Reducer algorithm

```
class REDUCER
    def REDUCE(string t, pairs [(s1,c1),(s2,c2)...])
        sum <-- 0
        cnt <-- 0
        for all pair (s,c) belongs to pairs:
            sum += s
            cnt += c
        rAvg = sum / cnt
        EMIT(string t, integer rAvg)
```

4. gCompute: Internet-scale AI dApps solution

4.a. AI model as an asset

Similar to the our definition of TrainedModel above, we can represent an AI model as an asset on the blockchain.

4.b. Scalable

Technical scalability: In order to compete with traditional internet giants like Google, Facebook, Alibaba, Tencent, our underlying platform blockchain technology has to horizontally scalable. If

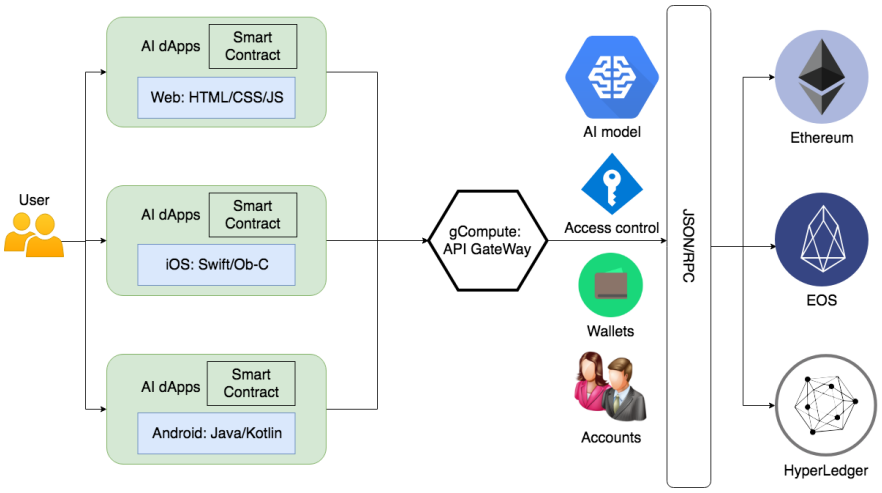


Figure 11: gCompute API GateWay

we have a AI dApp signup and use our platform for easy integration into all the existing blockchain backends, our system is able to handle tens of millions of daily active users. For this type of functional requirement, we will deploy multiple instances, abstract the core business logic into a docker and put load balancer in front of our instances to make sure every API/JSON calls went smoothly.

Business model scalability: Traditional internet company has proven to us that the best business model is to offer free service first and then offer a variety of revenue models (freemium, premium, subscription). So our underlying blockchain technology platform should have the flexibility to let application offer creative business revenue model. It is the free to use for users that causes the internet to gain more widespread adotion.

4.c. Controllable

Constant upgrades: Either big enterprises or small startups, even individual developers should have the flexibility to upgrade their application with new features constantly. Our platform will support both model software as well as smart contract upgrades. Even Google chrome, Facebook, Tencent WeChat, Apple iOS are subject to to different bugs or system failure, our platform and the underlying platform should be robust enough for bug fixes and automatic program call stack generation.

Continuous integration: When developers add new functionality to their existing softwares, they do not want to brake any old features. That is the where the continuous integration solution comes in. If certain commits break the old feature(can't pass certain testcases), the code hosting platform will reject that commit and generate a report about the error. Our system as well as the underlying blockchain platform will also support this kind of good practice.

Failure recovery: Worst case scenario: if we have network failure, system failure, underlying blockchain platform DDOS attack or any user key credential being compromised, our platform as well as underlying blockchain platform will have a failure recovery system. By versioning different

release of the AI models as well as the database state, we will be able to provide this kind of features for our end user as well as developers.

4.d. Difficult

PoW: PoW is the most adopted consensus algorithm for the unpermissioned public blockchain. Pioneered by bitcoin, PoW is the most known consensus algorithm on the blockchain ecosystem. PoW is hard to break and requires more than 50% attack to destroy the system. We will be able to safely integrates all PoW based public blockchain and finish most of our AI dApp business logic on the corresponding blockchain.

PoS: For the unpermionsoed public blockchain, PoS/BFT-DPOS/DPOS are the new challenger. PoS is proven to be more energy friendly. Under those algorithms, users who hold tokens on the corresponding blockchain platform are able to select block producers and produce blocks. Our AI dApp platform will be able to safely communicates with those public blockchain and implements all of our business logics.

SIEVE: For the permissioned private blockchain, there are numerous innovation around the consensus algorithm. One of the most interesting one is SIEVE. In short, SIEVE expands the first PBFT calculation by including theoretical execution and check stages to: 1) identify and sift through conceivable non-deterministic requests and set up the determinism of exchanges entering the PBFT 3-stage assertion protocol, and 2) enable agreement to be keep running on yield condition of validators, notwithstanding the accord on their info state offered by Classic PBFT. SIEVE is gotten from PBFT separately (motivated by thoughts portrayed in [Aublin et al., TOCS'15]) by reusing the PBFT see change convention to bring down its multifaceted nature and abstain from executing another accord convention sans preparation.By the nature of permissioned private blockchain, our system is able to securely connect those blockchain with access control implemented.

Quantum resistance: Post-quantum cryptography, otherwise called quantum-safe cryptography, can stand up to the assaults by quantum PCs. The improvement of such encryption innovation takes a more customary way, in view of troublesome issues in particular arithmetic fields. Through investigating and creating calculations, the post-quantum secure encryption innovation can be connected in the system, what's more, to give the most abnormal amount of information security. We will be looking forward to see exciting use case of such algorithm being put into practice on various public/private blockchain.

4.e. Maintainable

Extensibility: The power of modern software is built on top of each other. With vibrant support of third-party libraries and packages, developer is able to develop the software at much faster speed. We will provide a set of library dependency management tools to allow developers easily call and manage the already published smart contracts or import other contracts as a library dependency.

Readability: We will convert/parse all the transactions on the given blockchain and allow user to understand all our of systems AI dApps. User will be able to see who create the AI smart contract, the primary interactions of that smart contract, automatic smart contract abi generations, etc.

Explorbility: We should be able to explore the entire blockchain. We will provide insights into

the recent mined blocks on the blockchain. We will also provide insights into any transactions in any block that has already been mined and is currently attached to the blockchain network. User can also check the history of any public address and audit the balances, transaction history etc.

4.f. Responsive

Fast response: A great customer experience requests instantaneous feedback within couple seconds. User will get fractured if the application requires longer time to respond and the unresponsive AI dApp results in less competitive situation than traditional solutions. Our gCompute platform will try to optimize as much as we can to reduce the application response time.

Networking optimization: We will try to minimize the communication latency within the network. Network optimization ought to have the capacity to guarantee ideal use for system resources, enhance profitability and also effectiveness for the community. Network optimization takes a look at the individual workstation up to the server and the tools and connections related with it. Our system could consist of traffic shaping, redundant data elimination, data caching and data compression and streamlining of data protocols.

4.g. Concurrent

Synchronous execution: If the dApp itself requires sequentially independent steps, we should support fast sequential execution as well.

Asynchronous execution: Even though computers run in sequential mode, and uses the combination of locks & context switch to implement the pseudo asynchronous execution, modern applications all support concurrent execution by default. With the full internet-scale AI dApp solution in mind, we will try to optimize our code for the asynchronous execution as we can.